Introduction to **Threat Modelling** applied to **Connected and Autonomous Vehicles**

BY RICARDO M. CZEKSTER



Second Edition - October 2026

Self Publishers Worldwide

This page is intentionally left blank.

Introduction to Threat Modelling applied to Connected and Autonomous Vehicles

Modelling approaches for reasoning on vehicle cybersecurity

Second Edition – October/2026

Ricardo M. Czekster
Birmingham, United Kingdom



Self Publishers Worldwide

This book was typeset using LaTeX software (using OverLeaf – online).
Additionally, it was developed with the assistance of Large Language Models (LLM) technologies (using a mix of several tools, engines, and models).

Copyright © 2026 Ricardo M. Czekster
License: Creative Commons Attribution 4.0 International (CC BY 4.0)

Preface

Threat Modelling is eating the world.

Not quite, however, I hope that you, as a reader, gets my point.

The focus here is Connected and Autonomous Vehicle (CAV) and surrounding technologies. Their adoption will only increase in the near future, as everyone should pay particular interest in defending and protecting this infrastructure against any adversarial incursion.

This statement is even more true after the (not so gentle) introduction of Large Language Models (LLM)s to society, where users and analysts expect engines to "spit out" threat models readily useful for in-depth analysis, which is seldom the case.

I propose here a book to fill significant gaps for Threat Modelling CAVs, mostly because:

- No dedication to threat modelling: current books cover threat modelling as a topic among many cybersecurity concerns.
- Limited comprehensiveness: existing resources focus on specific methodologies (TARA) or general cybersecurity rather than comprehensive threat modelling approaches.
- Lack of CAV-related focus: most books address general automotive cybersecurity rather than the unique challenges of connected and autonomous vehicles.
- Academic vs. practical: research papers are highly technical but lack practical implementation guidance for industry professionals.

Having listed those reasons, we come to the conclusion that there is no comprehensive book dedicated specifically to threat modelling connected and autonomous vehicles.

It is my view that a comprehensive book on threat modelling for connected and autonomous vehicles must bridge theoretical frameworks with practical implementation while addressing the unique challenges of safety-critical, highly connected automotive systems. Success in such endeavours requires deep understanding of both cybersecurity principles and automotive engineering, combined with awareness of regulatory requirements and industry best practices. We sincerely hope that the resulting resource should serve as both an educational foundation and a practical reference for professionals working to secure the future of automotive mobility.

Table of Contents

List of Figures vii

List of Tables ix

Introduction 1

1 Foundations and frameworks 7
- 1.1 Usual attack vectors in CAV . . . 7
- 1.2 Overview of the attack surface of CAVs . . . 9
- 1.3 CAV-specific cybersecurity . . . 11
- 1.4 Threat Modelling Fundamentals . . . 12
  - 1.4.1 RM and RA approaches . . . 13
  - 1.4.2 A gentle introduction to TM mechanisms . . . 13
- 1.5 Regulatory Landscape and Standards . . . 14

2 Key RA methodologies for CAVs 17
- 2.1 Risk Assessment Methodologies & Techniques . . . 17
- 2.2 TARA: Threat Analysis and Risk Assessment . . . 21
- 2.3 STRIDE Framework for Automotive Systems . . . 24
- 2.4 Overview of techniques for TM . . . 26
  - 2.4.1 Data Flow Diagrams . . . 26
  - 2.4.2 DFD modelling in CAVs . . . 28
  - 2.4.3 Attack Trees . . . 29
  - 2.4.4 Attack Graphs . . . 32

3 Perspectives for threat analysis of CAVs 37
- 3.1 Future directions for CAV threat modelling . . . 38

Bibliography 43

List of Figures

1.1 Depiction of a reduced attack surface of CAVs. 9

2.1 Aggregating notions between ISO 31000:2018 and ISO/SAE 21434:2021. 18
2.2 V-model and selected clauses in ISO/SAE 21434:2021. 19
2.3 V-Model and DevOps comparison of tasks in modern SDLC. 20
2.4 Overview of TARA steps (slightly modified/appended). 22
2.5 Basic modelling primitives for DFD and DFD3. 26
2.6 Generic “Purchasing a product” model using DFD with inventory checks. 27
2.7 An AT and basic modelling primitives. 29
2.8 Working example of AT (nodes for describing methods are optional). 30
2.9 An AT that uses a SAND gate to represent ordered events. 32
2.10 Decorating leaves of an AT with qualitative modifiers. 33
2.11 An excerpt for an Attack-Defense Tree showing a defense node (MFA). 34
2.12 Examples of modelling systems with graphs. 35

List of Tables

1.1 Timeline of notable cyber-attacks and autonomy-related incidents against CAV and their impact on consumers. 8
1.2 Comparison of major RM|RA|TM frameworks. 15

2.1 Examples of attacks in CAVs using the STRIDE approach. 25

Acronyms

AG Attack Graph

AI Artificial Intelligence

AMT Attack Modelling Technique

API Application Programming Interface

APT Advanced Persistent Threat

AT Attack Tree

BCM Body Control Module

CAN Controller Area Network

CAV Connected and Autonomous Vehicle

CI/CD Continuous Integration/Continuous Delivery-Deployment

CIA Confidentiality, Integrity, and Availability

CPS Cyber-Physical System

CSMS Cyber Security Management System

CTI Cyber Threat Intelligence

CVSS Common Vulnerability Scoring System

DFD Data Flow Diagram

DORA DevOps Research and Assessment

DREAD Damage, Reproducibility, Exploitability, Affected users, Discoverability

E/E Electrical/Electronic

EBCM Electronic Brake Control Module

ECM Engine Control Module

ECU Electronic Control Unit

EV Electric Vehicle

FAIR Factor Analysis of Information Risk

hTMM Hybrid TM Method

HVAC Heating, Ventilation, and Air Conditioning

HW/SW Hardware & Software

ICS Industrial Control Systems

INCLUDES NO DIRT Identifiability, Non-Repudiation, Clinical Error, Linkability, Unlicensed Activity, Denial Of Service, Elevation of Privilege, Spoofing, Non-Compliant to Policy or Obligations, Overuse, Data Error, Information Disclosure, Repudiation, Tampering

IoT Internet-of-Things

IRM Inflatable Restraint Module

LINDDUN Linking, Identifying, Non-repudiation, Detecting, Data Disclosure, Unawareness, and Non-compliance

LLM Large Language Models

MFA Multi-Factor Authentication

ML Machine Learning

NCM Navigation Control Module

NFP Non-Functional Properties

OBD On-Board Diagnostics

OCTAVE Operationally Critical Threat, Asset, and Vulnerability Evaluation

OEM Original Equipment Manufacturer

OTA Over-the-Air

OWASP Open Worldwide Application Security Project

PASTA Process for Attack Simulation and Threat Analysis

QA Quality Assurance

RA Risk Assessment

RM Risk Management

RMF Risk Management Framework

SAE SAE International

SCA Software Composition Analysis

SDLC Software Development Life-Cycle

SDV Software Defined Vehicles

SoS System-of-System

SOTIF Safety of the Intended Functionality

SPARTA Security and Privacy Architecture Through Risk-Driven Threat Assessment

SPICE Software Process Improvement and Capability Determination

STRIDE Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege

TARA Threat Analysis and Risk Assessment

TCM Transmission Control Module

TM Threat Modelling

TMM Threat Modelling Method

USB Universal Serial Bus

V2B Vehicle-to-Building

V2C Vehicle-to-Cloud

V2D Vehicle-to-Device

V2F Vehicle-to-Farm

V2G Vehicle-to-Grid

V2H Vehicle-to-Home

V2I Vehicle-to-Infrastructure

V2M Vehicle-to-Motorcycle

V2N Vehicle-to-Network

V2P Vehicle-to-Pedestrian

V2V Vehicle-to-Vehicle

V2X Vehicle-to-Everything

VAST Visual, Agile, and Simple Threat

VVS Vehicle Vision System

Introduction

It is undoubtedly true that Connected and Autonomous Vehicles (CAV) will dominate future transport scenery in the years to come. Predominant are the intertwined notions of *connectivity*, *communication*, *remote management*, and *autonomy*. The idea of appending communication capabilities makes CAVs interact and connect with one another whilst maintaining the infrastructure for fixing defects or other issues. Specifically for autonomous driving, the SAE International (SAE) has created a classification for characterising and differentiating vehicles in the SAE J3016 document [1]:

- **Level 0:** No Driving Automation.
- **Level 1:** Driver Assistance.
- **Level 2:** Partial Driving Automation.
- **Level 3:** Conditional Driving Automation.
- **Level 4:** High Driving Automation.
- **Level 5:** Full Driving Automation.

CAVs are System-of-Systems (SoS) or Cyber-Physical Systems (CPS) as they consist of a plethora of systems and sub-systems equipped with communication capabilities through Internet-of-Things (IoT). The sheer number of Electrical/Electronic (E/E) components, ie, fundamental parts of any vehicle's complex network that controls its functions, including sensors, Electronic Control Unit (ECU)[1], actuators, connectors, wiring, mixed with other hardware and software elements.

All these elements within vehicles must be secured, preventing any serious vulnerabilities reaching road-drivers. At times, strict time-to-market constraints deploy unfinished products to the roads, lacking thorough testing rounds, and compliance checks. It is imperative to devise clever ways of checking overall software quality by means of automation, and, under security contexts, employing Threat Modelling (TM) approaches has proved useful and impactful.

The objective of this book is to detail how could TM be prescribed (mainly) for car Original Equipment Manufacturers (OEM), researchers, students, or other stakeholders as a *de facto* means of tackling security testing in CAVs, highlighting its features with real-world examples.

[1] In this book we distinguish between Electronic Control Unit (ECU) for electronic units and Engine Control Module (ECM) for engine related concepts (as ECU was originally termed "Engine Control Unit").

Given these concerns, coupled with adversaries' drive and relentless motivation to attack the infrastructure (or perform other malicious intents), steep time-to-market expectations (resulting in untested features being hastily deployed), and reasonable budgetary allocations by organisations, some key concepts in this domain promptly surfaces to help understanding the context. Although ISO/SAE 21434 [2] presented a good glossary for related terms and cybersecurity notions, we provide next academic related information based on sound research. For instance, in the seminal and time-tested[2] Avižienis et al. (2004) [3] work, there are interesting definitions that we append here:

- ***System***: "A system is an entity that interacts with other entities, ie, other systems, including hardware, software, humans, and physical world with its natural phenomena." The other systems are the environment of the given system.
- ***System Function***: what the system is intended to do.
- ***System Specification***: define the solution that the computer system has to implement to solve the problem. System function is described by the system specification.
- ***Correct Service***: correct service is delivered when the service implements the system function.

In terms of security, and related concepts, literature on the topic is vast and fruitful.

- ***Security***: broadly speaking, it entails preventing any harm, damage, or breach of assets. More precisely [3] (and better explained as follows), "Security is a composite of the attributes of confidentiality, integrity, and availability.".
- ***Cybersecurity***: consists on protecting digital assets, ie, hardware and software from cyber threats, eg, hacking attempts, disruptions, or data breaches, to cite a few concerns.
- ***Vulnerability***: a weakness in systems (software, hardware, network, configuration, etc.), that could be used to abuse, corrupt, exploit, or disrupt systems or impact functionalities, response times, or steal data, and so on.
- ***Threat***: anything (people/adversaries, events, etc.) with the potential of exploiting vulnerabilities.
- ***Threat Modelling***: the proactive and structured process of understanding potential threats affecting systems, listing their most-likely attack vectors, considering different designs that help securing systems further, and how to best approach tackling vulnerabilities.
  - It is a *Risk Assessment* technique (more to follow throughout the book), among many other that are designed to help understanding overall risks over systems.

Additionally, one might encounter the following 'notions' when working in this domain, that are also interesting to develop a bit further:

[2] "Time-tested" is an adjective meaning proven to be effective, valid, or useful over a long period of time. It describes methods, theories, or practices that have stood the test of time, demonstrating their reliability and workability through long-term use and experience.

- ***System Properties***: the system could be described in terms of its functional (about what a system does or how it performs) and non-functional (on how good, how fast, how secure, how usable, and so on it behaves when executed), or Non-Functional Properties (NFP) [3] properties, encompassing the broad term of *dependability* (listing in no particular order).
  - *Confidentiality* (C): the absence of unauthorised disclosure of information.
  - *Integrity* (I): absence of improper system alterations.
  - *Availability* (A): readiness for correct service.
  - *Reliability*: continuity of correct service.
  - *Privacy*: it is about the individual's right to control the extent and manner in which their personal information is shared across organisations and other stakeholders involved.
  - *Authenticity*: integrity of a message content and origin, and possibly of some other information, such as the time of emission.
  - *Accountability*: availability and integrity of the identity of the person who performed an operation.
  - *Safety*: absence of catastrophic consequences on the user(s) and the environment.
  - *Maintainability*: ability to undergo modifications and repairs.
  - *Resilience*: the ability of a system to retain operation under duress.
  - *Non-Repudiation*: availability and integrity of uniquely identifying the sender of a message (non-repudiation of the origin), or of the receiver (non-repudiation of reception).
- ***Trustworthiness***: according to NIST SP 800-12 Rev.1 [4], it is *"computer hardware, software and procedures that: 1) are reasonably secure from intrusion and misuse; 2) provide a reasonable level of availability, reliability, and correct operation; 3) are reasonably suited to performing their intended functions; and 4) adhere to generally accepted security procedures."*

Security is known to be a difficult problem, given the following six issues commented by Xu et al. (2021) [5] (cited here *ipsis litteris*, emphasis given by this book's author):

(i) The ***scale of cyberspace***[3], where billions of devices are interconnected and must be considered holistically;
(ii) The ***complexity of cyberspace*** in terms of the interdependence and interactions among its cyber-physical-human components;
(iii) The ***adversarial intelligence*** that attackers possess as skilled human beings which is on par with defenders;
(iv) The ***asymmetry*** between attackers who only need to identify and exploit a single vulnerability, and defenders who must seek to prevent or block each and every vulnerability;
(v) The ***difficulty in quantifying cybersecurity*** from a whole system (rather than building-blocks) perspective;
(vi) The ***multidisciplinary nature of cybersecurity***, technologically speaking, and especially the involvement of human beings as users, attackers, and defenders.

[3]According to NIST on-line glossary (`https://csrc.nist.gov/glossary/term/cyberspace`) that mentions NIST 800-30 Rev.1 [6], cyberspace is "a global domain within the information environment consisting of the interdependent network of information systems infrastructures including the Internet, telecommunications networks, computer systems, and embedded processors and controllers.".

Users sometimes expect security to be seamlessly present in systems. It is thus the job of people working on developing and generally working with system to actually go over and embed it throughout the solution facing end-users.

Finally, within our community's *parlance*/jargon, we often hear buzz words such as:

- ***Security Theatre***: to give the impression of being concerned about cybersecurity, without actually establishing controls or investing in system protections, also known as the 'illusion of cybersecurity'.

- ***Security Posture***: generally termed as an organisation's cybersecurity readiness to prevent, detect, respond, and recover from any cyber threats.

- Other words suitable for playing ***"corporate security bingo"***: let's go with 'security awareness', 'shift-left', 'kill chain', 'zero trust', 'zero-day' (or 0-day), 'APTs', 'AI security', 'holistic approach to cyber', 'paradigm shifting', and 'Digital Transformation' (there are many more, but we should stop now).

Stakeholders want systems that are resilient, trustworthy, 'fit for purpose', ready (ie, hardened) for market, and equipped with guarantees and protections that prevents any malicious incursions from happening. Under this "sea of definitions and concepts" it is imperative to effectively *focus* on model-based approaches that selects fundamental aspects to establish a sensible perimeter to defend. We, as a thriving cybersecurity community, should also be interested in learning and adapting defences proactively, proposing automated and high-coverage and comprehensive means of secure testing systems, and overall hardening CAV-based ecosystems (in terms of software).

Motivation

The vision for the future of mobility is where cars are safely deployed into roads, enacting congestion algorithms that balance the load over trajectories, and that pollute less and service society by ensuring protections (over the board, ie, data, etc) to passengers, drivers, and stakeholders. These ideas are in line with societal benefits in terms of safer vehicles (reducing crashes, increasing mobility, etc), boosting road-users productivity (more time to pursue other relevant goals), lower fuel consumption, and sustainability, to cite a few.

[4]Link: `https://upstream.auto/reports/global-automotive-cybersecurity-report/`, accessed on Oct/2026.

There is a sense in the community that cyber-attacks are increasing in sophistication, demanding equally increase in cybersecurity awareness and advanced ways of tackling surrounding issues altogether. Threat actors with advanced knowledge of systems and networks are conducting malicious incursions at scale, damaging the infrastructure, and affecting trust in CAVs. In terms of actual attacks, we list some examples of malicious activities next:

- Ransomware Attacks.
- Remote Hijacking and Keyless Theft.
- Cloud and back-end API Attacks.
- Electric Vehicle (EV) Charging Infrastructure Threats.
- Artificial Intelligence (AI) and Sensor Manipulation.
- Data Breaches and Supply Chain Exploits.
- Wireless updates via Over-the-Air (OTA) or physically in On-Board Diagnostics (OBD).
- Large-Scale Fleet Attacks.

These incursions, compounded with advanced phishing and social engineering practices that targets employees in attempts to steal credentials, access vehicle management systems, use insider information to promote attacks 'from within', and data exfiltration, to name a few, are happening at a frightening scale. More recently, we learned that adversaries can leverage Artificial Intelligence (AI) and promote API-driven botnet attacks on mobility apps and connected vehicle services, or perform zero-day exploits of onboard ECUs and safety systems. Unless everyone involved in cybersecurity seriously tackled the underlying problems (as the adage goes, "security is everyone's problem"), criminal activity will dominate this space, and society must prevent it from happening.

It is the objective of security experts and surrounding professions with an invested interest in cybersecurity to protect systems (broadly speaking) against threats. This book is about modelling potential threats that might arise in CAV contexts and how to enable teams' preparedness efforts to thwart, stymie, and mitigate cyber-attacks.

The CAV market has not yet realised its full potential, and the number of attacks is already unprecedented. One question that often surfaces by stakeholders in automotive circles is *"What are the **requirements** for achieving either UNECE R155/R156 or ISO/SAE 21434 **compliance**?"* The answer is multi-faceted, however, it could be summed up in having a ***good vulnerability management capability*** within the organisation. Vulnerabilities are introduced as early as the design phase of products, and one could readily employ Threat Modelling equally early to prevent defects from reaching end-customers, saving money and time.

Intended audience

Imagine for a moment that you are a security engineer working for an automotive OEM, and you have been tasked with performing threat modelling for a new CAV product that is about to be released into the market. Or, perhaps, you are a cybersecurity student looking to specialise in automotive security, and you want to understand how threat modelling applies to CAVs. Even more, you might be a researcher aiming to explore new methodologies for threat modelling in the context of connected and autonomous vehicles.

Within these roles, you might be wondering where to start, what methodologies to use, standards, and how to ensure that a set of threat models is comprehensive and effective. This book is designed to guide you through the process of threat modelling specifically for CAVs, providing you with practical insights, methodologies, and examples to help you succeed in your career.

Whether you are a security professional, a researcher, a student, or anyone interested in the field of automotive cybersecurity, this book aims to equip you with the knowledge and tools necessary to navigate the complex landscape of threat modelling for CAVs. By the end of this book, you will have a good understanding of the principles and practices of threat modelling in the context of CAVs, enabling you to contribute effectively to the security of these systems/sub-systems.

Enjoy!

Chapter 1

Foundations and frameworks

We establish the context by tackling cybersecurity underpinnings first, followed by addressing threat modelling in CAV.

Let's pose a few questions to drive our thinking next:

- Whenever a car operates oddly or some component malfunctions, does that configures an unremarkable event or something more serious, like the first step of a sophisticated cyber-attack?
- Are anomalies in logging, tracking, or data transfers something usual or advanced malicious data exfiltration?

These questions concern cybersecurity, privacy, and trust. First, let's understand how attacks take place in vehicular settings, so we map the attack surface (and assets) that needs protecting.

1.1 Usual attack vectors in CAV

Security is about protecting systems against damage, data tampering or theft, improper use, or unauthorised access, to name a few. It is about understanding how to increase trust and how malicious actors inflict damage in terms of Confidentiality, Integrity, Availability known in the field as "The CIA triad".

Before we proceed mapping the attack surface of CAVs, we discuss a brief timeline with examples of cyber-attacks on vehicles. The rate of malicious incursions jumped off the chart as cars became more electronic and digitised throughout the years. Manufacturers started deploying products with thousands of E/E components and sub-components, in a mix of software and hardware-based controls.

The idea was that modern transportation mandate advanced features for remote management capabilities, fast-paced communication among heterogeneous fleets, and autonomy, to name a

few advances. At the same time of these developments that made cars become smarter, the pace of cyber-attacks has grown accordingly. It is not the objective here to describe cyber-attacks in detail, as there is a host of literature on the topic [7, 8]. The focus here is to understand how to prevent and anticipate cyber-attacks with modelling approaches (namely TM), prioritising incursions with significant repercussion to customers and OEMs. We present next in Table 1.1 a non-exhaustive list of major incidents over the past 10 years.

Table 1.1: Timeline of notable cyber-attacks and autonomy-related incidents against CAV and their impact on consumers.

Date	Incident/Target	Attack Type/Vector	Consumer Impact
July 2015	Jeep Cherokee (Fiat Chrysler)	Remote exploit via infotainment	Hackers remotely controlled steering, braking, acceleration. Fiat Chrysler recalled 1.4 million vehicles. System firmware was updated for consumer safety.
March 2016	Multiple brands (Audi, BMW, Ford, Toyota, etc.)	Relay attack on keyless entry	Attackers unlocked cars using intercepted fob signals. affected. Enabled vehicle theft;
2016-17	Tesla Model S	Exploit chain via Wi-Fi and browser	Researchers gained remote control over braking, side mirrors, and locks. Led to over-the-air software patches, increased consumer scrutiny.
2018-2021	BMW (14 vulnerabilities found)	Local and remote exploits	Vulnerabilities affected telematics units and gateways. BMW issued urgent security updates to consumers, preventing remote takeover.
2023-2024	100+ incidents - Global CAV ecosystem	Ransomware, data breaches	Disrupted automotive and smart mobility services, including some EV charging networks. Consumers faced disruptions to vehicle functions, services, and fear about data exposure. Auto recalls and regulatory changes ensued.
June 2024	Waymo self-driving taxi, Phoenix/US	Partial system failure	Vehicle crashed into utility pole; All 672 vehicles recalled for urgent fixes. Public trust and confidence in autonomous vehicles affected.
2024	Ford BlueCruise	System misuse, not a targeted hack attempt	Two major fatal crashes while hands-free mode enabled; raised concerns about misuse and ability to handle edge cases and potential malicious triggers.

It is not enough to make individual components secure if the ecosystem is not hardened. Entry points could virtually be any, from infotainment to more elaborate ECUs or end-user interfaces.

We present a list of key ECUs [7] by level of importance in descending order:

1. Navigation Control Module (NCM).

2. Engine Control Module (ECM).
3. Electronic Brake Control Module (EBCM).
4. Transmission Control Module (TCM).
5. Telematics module with remote commanding.
6. Body Control Module (BCM).
7. Inflatable Restraint Module (IRM).
8. Vehicle Vision System (VVS).
9. Remote door lock receiver.
10. Heating, Ventilation, and Air Conditioning (HVAC).
11. Instrument panel module.
12. Radio and entertainment centre.

1.2 Overview of the attack surface of CAVs

Figure 1.1 shows a reduced representation of the potential attack surface of CAVs with various *interaction* models (V2X) and stakeholders.

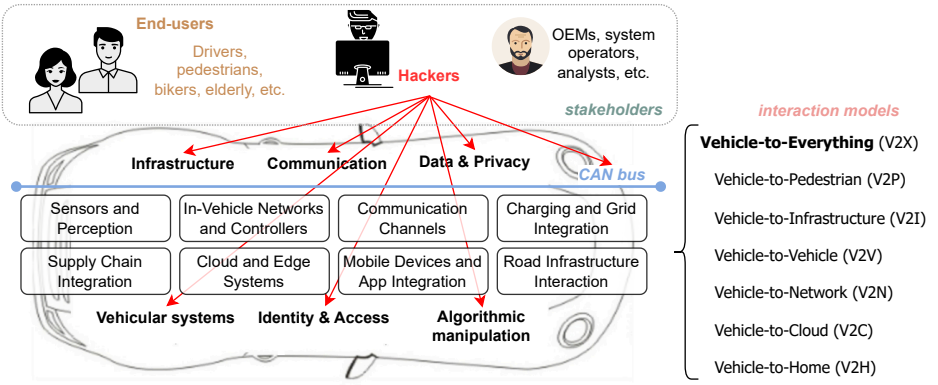


Figure 1.1: Depiction of a reduced attack surface of CAVs.

The figure also shows some dubbed "interaction models" of CAVs with the world:

- Vehicle-to-Everything (V2X)
  - Vehicle-to-Vehicle (V2V)
  - Vehicle-to-Infrastructure (V2I)
  - Vehicle-to-Pedestrian (V2P)
  - Vehicle-to-Network (V2N)
  - Vehicle-to-Cloud (V2C)
  - Vehicle-to-Device (V2D)
  - Vehicle-to-Grid (V2G)
  - Vehicle-to-Home (V2H)

  - Vehicle-to-Building (V2B)
  - Vehicle-to-Motorcycle (V2M)
  - Vehicle-to-Farm (V2F)

Unless it is relevant to the discussion, from now on we shall use only V2X to refer to any vehicle interactions required to explain concepts altogether.

Before we delve into the cybersecurity problem, there are a few things we must consider, listed next:

- **Massive and dynamic attack surface:** a 'normal' CAV has multiple systems, some researchers estimate lines of code for all ECUs, which is much larger than MS-Windows code, totalling about 40 million lines of code[1].
- **Malicious incursions:** there is a need to secure vehicles in terms of their cyber and physical domains, as they sit on customers/maintenance/elsewhere sites, being a 'security testing' venue for attempting to break or sometimes even incompetent use of technologies.
  - As an example, a lot of attacks focus on the Controller Area Network (CAN) bus in modern vehicles. Some adversaries find intricate ways of injecting malicious packets into the bus, thus influencing the car's behaviour and responses towards criminal aims. One example is known as Message Injection Attacks where threat actors leverage susceptible (without built-in encryption or authentication mechanisms) CAN buses to exploit vehicular systems and sub-systems, or perform 'Lateral Movement'[2].
- **Compliance to standards:** there is a plethora of regulations to adhere to in the vehicular domain (more to follow).
- **Intricate software ecosystems:** on cars, OEMs have ECUs operating jointly, ie, in Coupled ECU mode, which creates dependence on modules.
  - Software outsourcing of solutions compounds this problem even further, as multiple vendors, and multiple software teams tend to use different software versions, etc.
  - Vulnerabilities surface on specific software versions, and keeping track of everything is time-consuming. There are automated tools to keep tracking of these versions between deployments, for instance, Software Composition Analysis (SCA) tools that are currently used in DevOps practices.
  - Whenever patching systems gets required/mandated, these versions might influence each other, bringing forth other vulnerabilities.
- **Numerous stakeholders:** security issues affect OEMs, managers, risk and safety analysts, threat modellers, developers, testers (quality assurance), and stakeholders sharing roads and walkways, eg, pedestrians, road drivers, and so on.

Tackling cybersecurity is about convenience between manufacturers and end-users; on the one hand, it must be trustworthy, safe, and also reliable and on the other hand, end-users are to be the least annoyed as possible to fulfil basic requirements. Unless physical problems surface, remote management and OTA updates should handle important maintenance seamlessly. Unresponsive, rogue, and uncontrolled vehicles would be unsatisfactory properties to have, especially if caused by weak cybersecurity posture and ineffective controls.

[1] Lines of code is an *inadequate* metric in software, however, it gives a good indication of the sheer size of the problem.
[2]

Ideally, vehicles should return to basic operation (ie, disconnect it altogether) at the minimum confirmed evidence of wrongdoing, following the idea of a "circuit breaker" for cars. In reality, however, given the prevalence of interdependence, such a feature would be difficult to code.

1.3 CAV-specific cybersecurity

Given that CAV ecosystem is increasingly based on software, techniques for securing software, generally speaking, are to be embed in solutions. Next, a list of stress points in tackling vehicular security [9]:

- **Steep time-to-market deadlines:** This rushes deployment of incomplete/untested products to end-customers.
- **Lack of testing and Quality Assurance (QA):** Increased adoption of "let's do security now" mindset, often right before deployment, without it being a concern since the design phase of features. No systematic effort to tackle product quality not only in the OEM's codebase but throughout the software ecosystem including external libraries and Application Programming Interface (API).
- **Need of (near) real-time decision making:** As sensors collect data, internal systems and sub-systems in vehicles must make decisions to consistently act in a safe and secure manner, protecting road-users, pedestrians, and the infrastructure. Adversaries may attempt to trick cameras or influence sensors in ways that they start operating wrong or yielding improper actions given the (erroneous) data they were fed.
- **Extended connectivity:** CAVs are notoriously highly interconnected (as a feature), demanding to interact with the infrastructure, other vehicles, and so on, constantly. Despite being a strength, it increases the attack surface, and may lead to faulty operational statuses, when cybersecurity gets neglected.
- **Data and privacy:** internal car sensors and systems/sub-systems generate massive data that could be exfiltrated by malicious third-parties.
- **Plethora of systems and sub-systems:** These Hardware & Software (HW/SW) elements operate interdependently, ie, they depend on different systems to enable the whole vehicle's features. Systems operating wrongly (due to many reasons, from hardware faults to malicious interventions) reduce trust and may even be unsafe for end-users, or cause data leaks (impacting privacy), and so on.

Modern techniques help stakeholders creating (cyber)secure and safe vehicular solutions, eg:

- **Shifting-left:** The idea is to think about security since the design of a feature.
- **Early TM:** Analysing threats since early designs are proven to add value to projects, as teams fix issues quickly before it gets compounded and more difficult to discover/patch.
- **Use of secure Continuous Integration/Continuous Delivery-Deployment (CI/CD):** Appending security tools and automated testing in the pipeline before the deployment of features to end-users.
- **Track full data life-cycles:** Observing how data traverse systems is fundamental for TM, as some models try to understand the modifications applied to data to determine whether or not a cyber-attack took place or if any system is operating anomalously. Currently, there

are some initiative considering what was dubbed 'observability', ie, a way of continuous monitoring systems to understand behaviours.
- **Use of simulated environments to test ideas:** The objective is to use Software Defined Vehicles (SDV) application suits to virtually deploy software and inspect behaviour before it goes to operational settings.

These measures and countermeasures, however, stretch resources to their limit.

1.4 Threat Modelling Fundamentals

TM is nowadays a *loaded* term (unfortunately) that gets confused with different notions. For example, Shostack (2014) [10] considers it an umbrella term with multiple associations, suggesting that it is "the key to a focused defense". Authors Uzunov and Fernandez (2014) [11], state that TM is *"a process that can be used to analyze potential attacks or threats, and can also be supported by threat libraries or attack taxonomies"*. Focusing on software development, Tarandach and Coles (2020) [12] consider TM to be *"the process of analyzing a system to look for weaknesses that come from less-desirable design choices"*. The Open Worldwide Application Security Project (OWASP) on its page about TM[3] comments that it *"works to identify, communicate, and understand threats and mitigations within the context of protecting something of value"*.

The Threat Modeling Manifesto[4] suggests that *any* TM process should answer the following four questions:

1. What are we working on?
2. What can go wrong?
3. What are we going to do about it?
4. Did we do a good enough job?

Mind that the same questions are readily applicable for analysing **any** risk to **any** system.

Also, bear to the fact that ***no system will be 100% immune to attacks*** after strictly following all major guidelines, recommendations, and regulations. They will most likely be deployed with vulnerabilities and exploit opportunities for adversaries to poke around and find security gaps in the perimeter. The only sure-proof system 100% secure that exists is one that is isolated, operating without any dependence, closed, and without external access (it may still be prone to hardware faults). As far as the stakeholders are concerned, they did the best possible job to secure and harden the system against malicious incursions.

What is the difference between 'threats' and 'risks'? (let's not even mention 'hazard', for the sake of clarity). Answering that demands considering this scenario: *"I am walking on a dense forest alone."*

- **Threat:** I wonder that I might spot an aggressive grizzly bear (ie, a source of potential harm).
- **Risk:** I ponder over the uncertainty whether or not I might get bitten by a grizzly bear, the likelihood, impact, and severity for me ('are there even bears here?', going to hospital, lose a leg, involved costs).

[3] Link: `https://owasp.org/www-community/Threat_Modeling`.
[4] Link: `https://www.threatmodelingmanifesto.org/`.

This example in particular does highlight the fact that an 'agent' (a *threat actor*) cause 'something' to 'someone'. And that is not always the case as, for example, a threat might be 'rain', which may affect the interaction altogether. Additionally, another threat might be the fact that one might get lost in the forest (no sun, no compass), accidents throughout the route (break an ankle), becoming ill (by eating a poisoned root), trees falling along the path, and so on.

According to Tarandach and Coles [12], TM means to 'look for weaknesses" in interacting assets. For the authors, central is the notion of interacting components, actors, and potential weaknesses. Then, for the previous example, the (threat) actor is the grizzly bear, the interacting components are us two, and the weakness is my lack of preparedness to stay alive.

For Shostack, *"A threat is a possible future problem; a risk is a quantified threat. That quantification often applies to likelihood or impact."*[5] For the author, TM is about tackling easily addressable things in systems, from a technical standpoint, and use RM for trickier situations that arise.

Effectively, those are broad (risk-inspired) security requirements to establish protections of systems (generally) to a host of stakeholders. More specifically, TM is one of many Risk Assessment (RA) techniques belonging to an overall Risk Management (RM) capability.

1.4.1 RM and RA approaches

RM is a broad activity in organisations that involves governance, leadership, commitment, and improvement, ie, those elements forming many managerial aspects for tackling risks. There are several methodologies for managing risk, for example ISO 31000:2018 [13], OCTAVE [14], and Factor Analysis of Information Risk (FAIR) that tackles quantitative information risk management (broadly used by industry[6]), to mention a few.

ISO 31000:2018 defines *risk* as the effect (ie, deviation from expected) of *uncertainty* on objectives. It develops more concepts around three pillars, namely: 1. Framework (governance, leadership, commitment, improvement); 2. Principles (value creation & protection); and 3. Process (context, risk communication, risk assessment – risk identification, analysis, evaluation – risk treatment, and reporting/monitoring risk continuously. The document also defines other risk-related concepts such as risk sources, events, consequences, likelihood, impact, and controls.

RA is a task within a broader RM effort [16, 15]. Across different approaches [17], it entails three phases: 1. Risk Identification; 2. Risk Analysis; and 3. Risk Evaluation. As mentioned, TM is a RA technique used to tackle risk in projects, being more direct and usually operating within teams developing solutions.

1.4.2 A gentle introduction to TM mechanisms

In terms of TM approaches and methodologies we mention Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege (STRIDE) [10], Process for Attack Simulation and Threat Analysis (PASTA) [18], Linking, Identifying, Non-repudiation, Detecting, Data Disclosure, Unawareness, and Non-compliance (LINDDUN) [19], Attack Tree (AT) [20, 21] or Attack Graph (AG) [22, 23], Persona non Grata, Security Cards, Hybrid TM Method (hTMM),

[5] Link: `https://shostack.org/blog/risk-management-and-threat-modeling/`.

[6] Visit `https://www.fairinstitute.org/` for more information.

Damage, Reproducibility, Exploitability, Affected users, Discoverability (DREAD), Quantitative TMM, TRIKE, Visual, Agile, and Simple Threat (VAST), INCLUDES NO DIRT[7], Security and Privacy Architecture Through Risk-Driven Threat Assessment (SPARTA), and CORAS [24]. These techniques were better explored by Xiong et al. (2019) [25] and also in Tarandach and Coles (2020) [12]. It is out of the scope of this book to detail any of those particular methods.

In this book, we shall comment on the ones that are the simplest approaches and have the most impact in vehicular research, namely Data Flow Diagram (DFD), STRIDE, and AT/AG:

- **DFD:** It employs the "follow the data" mindset, tracking how it traverses through components. Some TM techniques adopt this visual notation to describe threats, eg, STRIDE, TRIKE, and PASTA.
- **STRIDE:** Used to represent the most likely aspects and preoccupations of cybersecurity to observe when designing and coding systems. For each aspect, it maps a CIA related concern.
- **AT:** Useful for understanding major risk areas to focus attention, and also on communicating risk to multiple background stakeholders. It provides a hierarchical view of most-likely attacks and depicts tasks that attacks must accomplish for abusing systems.
- **AG:**

As mentioned, tackling cybersecurity entails addressing CIA concerns, which are encapsulated by the methods just mentioned.

1.5 Regulatory Landscape and Standards

Previous research outcome [26] discussed standards and approaches for cybersecurity, safety, and privacy issues in CAV.

We highlight the document ISO/SAE 21434:2021 [2, 27] focused on cybersecurity for the automotive industry that replaced SAE J3061 [28, 29]. In 2021, two United Nations (UN) regulations about vehicle cybersecurity (UNECE WP.29/R155) [30] and software updates (UNECE WP.29/R156) [31] provided binding regulations for CAVs. They require **type approval** and document **lifecycle security management** principles to guide OEMs and mandates Cyber Security Management System (CSMS) to be operational for vehicular deployment in roads. Also important was the publication of ISO 21448:2022 [32] on safety of road vehicles to ensure the Safety of the Intended Functionality (SOTIF).

[7] Full acronym is: Identifiability, Non-Repudiation, Clinical Error, Linkability, Unlicensed Activity, Denial Of Service, Elevation of Privilege, Spoofing, Non-Compliant to Policy or Obligations, Overuse, Data Error, Information Disclosure, Repudiation, Tampering.

Table 1.2: Comparison of major RM|RA|TM frameworks.

Type	Framework	Focus/Approach	Typical Use	Key Features, Strengths, Weaknesses
TM	STRIDE	Threat categories (DFDs)	Software, IT systems	Six threat types, DFD-driven. Simple and comprehensive, easy for beginners. May lack nuanced risk prioritization.
	PASTA	Risk/business-centric	Large, complex organizations	Seven-step, simulation-based, risk-aligned. Very thorough. Complex and time-intensive.
TM	LINDDUN	Privacy-focused (DFDs)	Privacy-critical systems	Privacy threat mapping and systematic analysis. Effective for privacy, but broader threats are less covered.
RM	OCTAVE	Asset/risk-based	Organizational IT	High-level risk analysis, good for management. Limited technical depth.
RA	DREAD	Quantitative risk scoring	Supplemental, combined usage	Five-factor risk scoring for prioritization. Effective for ranking, but subjective and less commonly used alone today.
	VAST	Scalable, DevOps-ready	Large enterprises	Supports automation and process-driven analysis, scalable for enterprise/DevOps. Limited documentation/adoption.
RA	FAIR	Large industrial support	Medium-Large enterprises	Quantitative approach for handling risk. Allow to rank risks, analysing occurrence frequency and impact.

Type: Risk Management (RM), Risk Assessment (RA), or Threat Modelling (TM).

- Executive management leadership and commitment to the effort of engineering cybersecurity in measure with expectations.
- Standardised roles and responsibilities with special attention to cybersecurity and privacy across the supply chain.
- Adoption of industry-wide terminology for cybersecurity processes.
- Promote good practices for continuous monitoring and incident response capabilities.
- Documentation and traceability requirements.

In this book, we will detail the TARA approach in CAV context, and explain how teams are supposed to adopt the approach for increased value and cybersecurity protections.

Chapter 2

Key RA methodologies for CAVs

Why? Because systems are prone to vulnerabilities, regardless the amount of testing performed by QA teams. They invariably present defects and it the job of risk analysts and threat modellers to identify and mitigate these shortcomings before they reach end-users. There is a need for a systematic approach when handling these issues, and this chapter will focus on key methodologies directly applicable to CAV contexts.

2.1 Risk Assessment Methodologies & Techniques

Risk Assessment (RA) operates within a broader Risk Management (RM) methodology, which is broader in scope, and tackles other risk-related domains that organisations must address. It is out of the scope of this book to detail existing RM methodologies, the focus here is directly on RA and its repercussions in CAV's threat modelling concerns. As mentioned, interested readers should find extensive literature on this topic, for example, OCTAVE (and variants), ISO 31000:2018 and IEC 31010:2019 (this one about RA techniques), NIST RMF, or FAIR, to cite a few.

Figure 2.1 shows a comparison between ISO 31000:2018 and ISO/SAE 21434:2021. There is a clear overlap among tasks in both standards. Perhaps one major difference is that ISO 31000:2018 is a generic document applicable to any risk (or to any uncertainty) arising in any systems whereas ISO/SAE 21434:2021 (describing TARA) is focused on assets, threats, attacks, impact, and treatment that are specific to CAVs. The figure shows how ISO/SAE 21434:2021 break down tasks on what is basically the "Risk Assessment" as described in ISO 31000:2018 plus the "Risk Treatment" task that is common to both approaches.

The most important phases for tackling uncertainty (ie, cybersecurity protections) are the combined effort of Risk Assessment with Threat Modelling:

- **Risk Identification:** Employ techniques for eliciting risk views such as brainstorming, Delphi technique, interviews, surveys.
  - Asset identification (Clause 15.3): Focus on information assets, physical assets, human assets, and intangible assets. Work towards awareness and stakeholder engagement or performing physical audits and inspections, automated discovery tools (networks and connectivity, and intra-system as well).

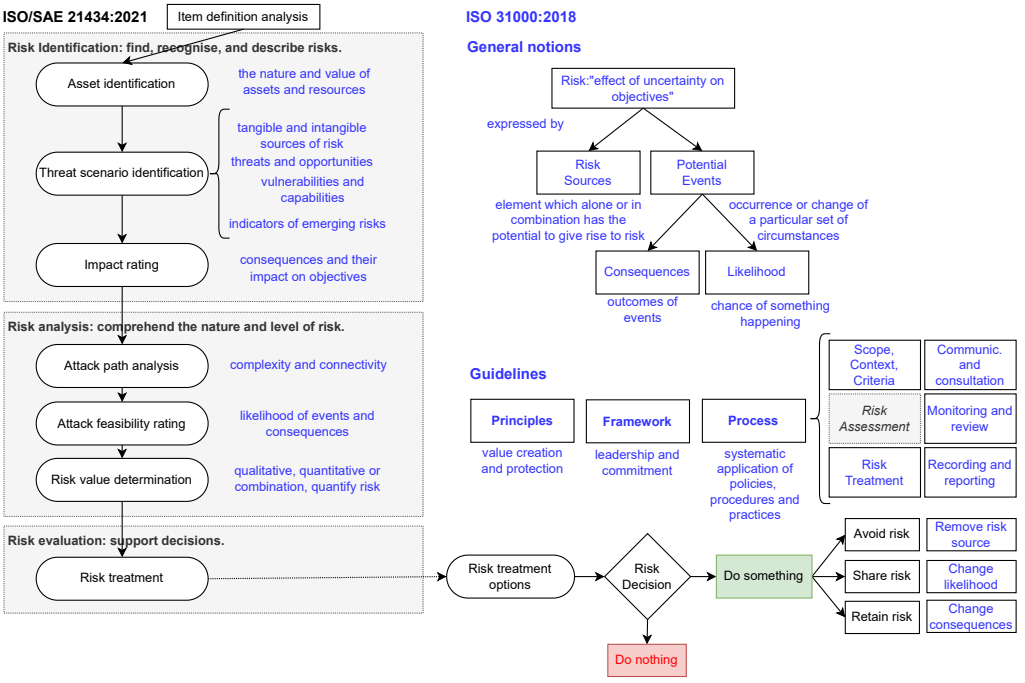


Figure 2.1: Aggregating notions between ISO 31000:2018 and ISO/SAE 21434:2021.

  - Vulnerability analysis (Clause 8.5) and management (Clause 8.6): Identify weaknesses in systems/sub-systems.
  - Identify major threats and threat actors (some call this 'Threat Scenarios').
- **Risk Analysis (Clause 15):**
  - Attack path analysis (Clause 15.6) and
- **Risk Evaluation:**
  - Risk treatment (Clause 15.9): There is a host of actions to take, for example, avoid, share, retain, remove risk sources, change likelihood, change consequences.

ISO/SAE 21434:2021 goes a bit further on suggesting the use of the V-model, depicted as an excerpt in Figure 2.2, showing the processes of 'designing items', 'product development', and how it interacts with validation tasks. It stresses the importance of designing 'items' and considering 'validation' at the same level. In Clause 10 it suggests: i) definition of cybersecurity specifications; ii) verification of cybersecurity specification across architectural levels; iii) identify weaknesses in components; and iv) provide evidence that the results of the implementation and integration of components conform to the cybersecurity specifications.

In terms of verification, it lists two ways for performing this task:

1. It could include review, analysis, simulation, and/or prototyping.
2. Additionally, it could employ requirements-based test, interface test, resource usage evaluation, verification of the control flow and data flow, static and/or dynamic analysis.

For the process of validation, it recommends: i) functional testing, ii) vulnerability scanning, iii) fuzz testing, and/or iv) penetration testing.

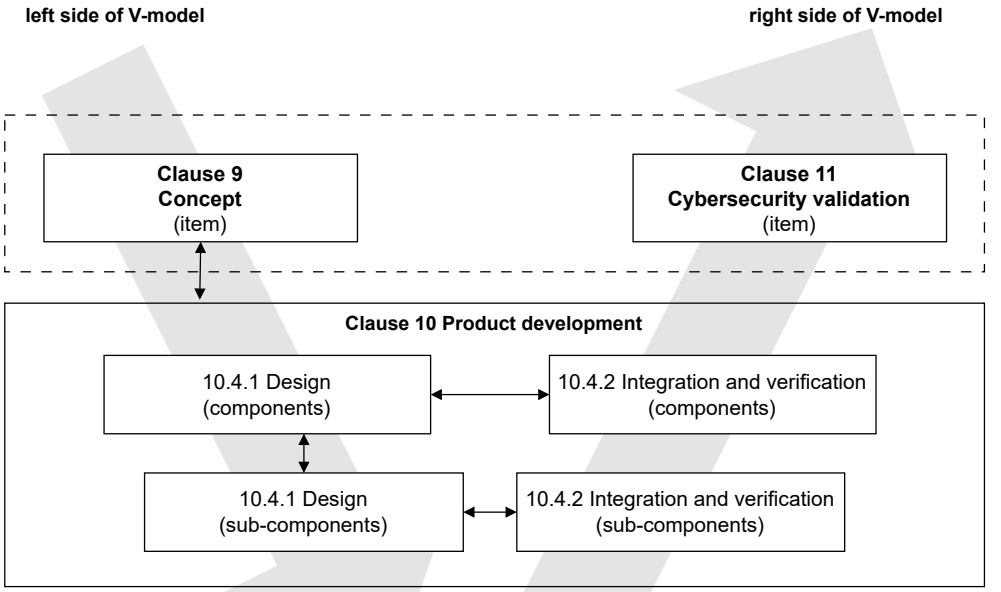


Figure 2.2: V-model and selected clauses in ISO/SAE 21434:2021.

In retrospect, CAV product development (software items), in terms of quality assurance and reducing defects reaching end-customers, is not that different from 'normal' software development. Both activities require specifications and verification/validation tasks to ensure that products meet objectives.

Figure 2.3 depicts the 'classic' V-Model and contrasts with modern Software Development Life-Cycle (SDLC) using DevOps (Development + Operations) embed with security tasks. The idea of the latter approach is to shorten design from deployment in a continuous approach that entails planning/design, coding, building solutions, testing, releasing, deploying, operating, and monitoring. It is highly useful for systems involving software teams as they need to frequent integrate their code so they have the latest version to work with at (almost) any point.

DevOps does not replace the V-Model in automotive contexts; rather, the two are increasingly combined. Safety and security standards such as ISO 26262, ISO/SAE 21434, and Auto-

motive Software Process Improvement and Capability Determination (SPICE) remain structured around the V-Model, as it provides the traceability between requirements, design, and verification that type approval and audits demand. DevOps practices complement this by shortening feedback loops within each stage, allowing software issues to be detected and fixed rapidly as they surface (and before deployment), while preserving V-Model's verification and validation evidence. Additionally, DevOps enables a faster response whenever a vulnerability is found, as patches can be built, tested, and delivered via OTA updates; note, however, that in vehicles rollback to earlier versions is typically restricted by anti-rollback protections and UNECE R156 software update management requirements, so remediation usually means *patching forward* rather than *reverting*.

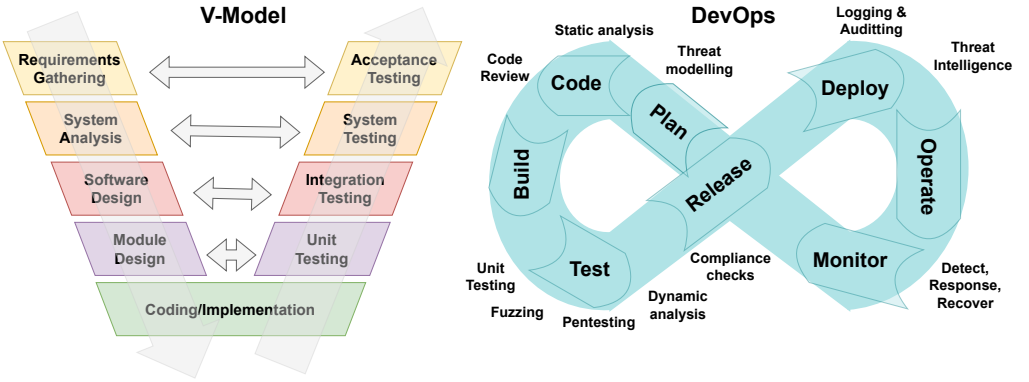


Figure 2.3: V-Model and DevOps comparison of tasks in modern SDLC.

The V-Model belongs to a period where organisations conceived entire products for releasing them to the public whereas modern approaches break down the concept into a system being composed by features, that eventually gets deployed to end-users. It still considers the SDLC phases, ie, plan/design, code/build, test, and release/deploy, however, it does that to one or a set of features. Additionally, after it reaches user-facing environments, it gets tracked, monitored, and it could be decommissioned as analysts, security officers, and quality managers see fit, depending on performance, security assurances (cyber-attacks, CIA violations), and misbehaviours/anomalies get detected.

Notice that DevOps in the figure embeds security related tasks throughout the SDLC. Some authors call this DevSecOps, however, for the sake of simplicity, we will adopt just DevOps here. Security is a fundamental requirement to any system facing end-users, ensuring protections are in place for safeguarding data, privacy, and customers from harm (safety). If teams are considering the security underpinnings on the feature-level, the idea is that after testing an ensemble of features within each release, it might help securing the system as a whole, incurring in less development time, and producing lower amounts of defects ending in production environments. It is out of the scope here to discuss DevOps any further, we point the users towards the extensive literature on the subject, as well as the DevOps Research and Assessment (DORA) metrics (from Google), to cite a few materials.

2.2 TARA: Threat Analysis and Risk Assessment

Threat Analysis and Risk Assessment (TARA) [2] is a cybersecurity methodology

, offering a systematic approach to risk evaluation focused on identifying, analysing, and mitigating cybersecurity risks in CAVs. The need for such approach has emerged in recent years as modern vehicles become increasingly software-driven and Internet-connected. TARA serves as a foundational mechanism for ensuring acceptable levels of cybersecurity risk throughout vehicle's lifecycle, from initial design to decommissioning.

In terms of literature on the subject, Benyahya et al. (2023) [33] investigated about RA for CAVs by summarising ISO/SAE 21434:2021 main concepts. A recent approach dubbed TARA 2.0 [34] that addresses higher automation requirements for vehicles offers an enhanced framework for addressing cybersecurity, privacy, and expert subjectivity.

In ISO/SAE 21434:2021's Clause 15, TARA follows a structured seven-step methodology given by Figure 2.4.

The idea of TARA's methodology is to investigate potential venues of vulnerabilities in CAVs, understanding threat actors, and addressing impact and likelihood to the organisation (as any RA methodology also does). Expanding on the main concepts of the figure, it describes:

Step 1: **Asset Identification:** begins with enumerating all assets and characterising them in terms of their CIA components. Examples of assets, in this context, are (non-exhaustively):

- Hardware components: ECUs, sensors, actuators, or controllers.
- Software modules: Operating Systems, applications, or firmware.
- Communication interfaces: CAN bus, Ethernet, wireless connections, V2X interfaces.
- Data: User information, vehicle telemetry, or cryptographic keys.
- Functions: Autonomous driving capabilities, braking systems, or steering control.

Step 2: **Threat Scenarios:** employing TM, analysts probe most likely cybersecurity threats using frameworks such as STRIDE, DFD, and so on.

Step 3: **Rating impact:** the idea here it to take each scenario and evaluate them in terms of safety (harm/damage to passengers, drivers), financial (repair or recall costs, lawsuits, reputational damage), operational (consequences to vehicle usability, loss of control, improper operation), or privacy (user data exposure or leakage of sensitive information) impact.

Note: Because this task creates a numerical rate for impact, each dimension is typically rated on a scale such as 1 (negligible), 2 (moderate), 3 (major), and 4 (very high/severe). Higher impacts get prioritised.

Step 4: **Attack path analysis:** it involves a detailed analysis of the sequences of actions that threat actors require to realise a threat scenario. Analysts could employ Attack Tree or Attack Graph to accomplish this task. It identifies tasks leading to the root causes of attacks over assets. It helps communicating the ways adversaries may exploit vulnerabilities and how dependent is the system behaving between attack steps.

Step 5: **Attack Feasibility Rating:** This task assesses how likely it would be for attackers to successfully exploit identified threats. Five attributes are then evaluated (next it shows an example of scales that could be used, suitable ranges should be adapted by experts):

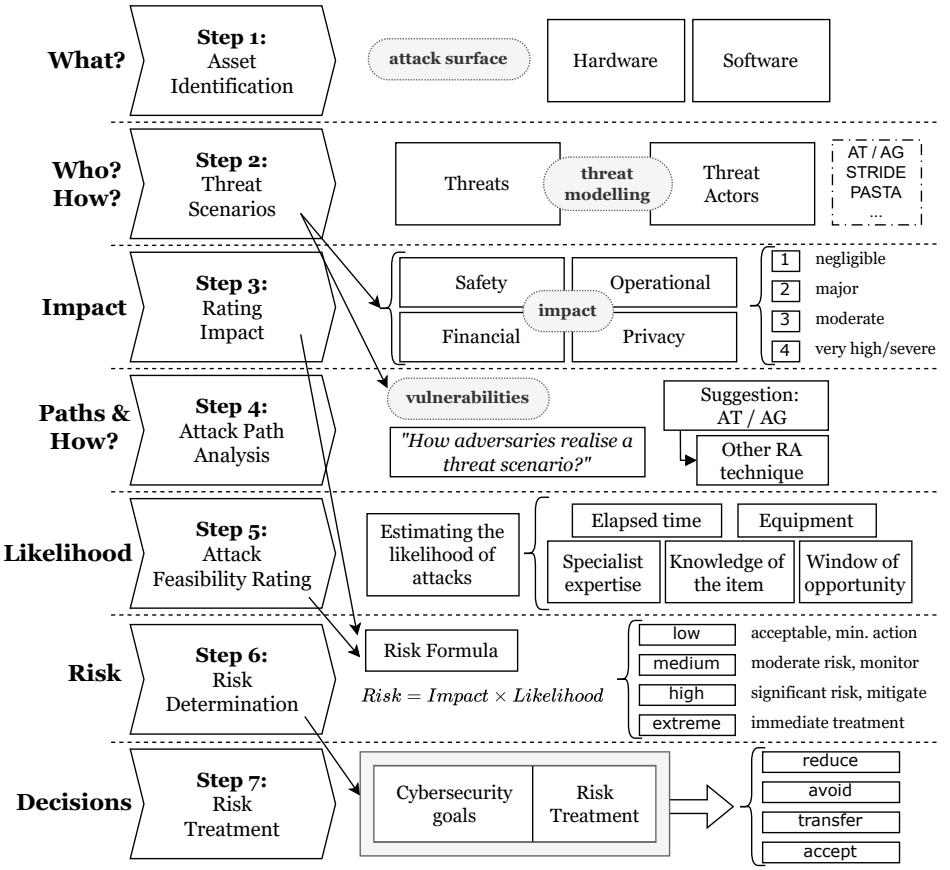


Figure 2.4: Overview of TARA steps (slightly modified/appended).

- Elapsed Time: Duration required to complete the attack ($\leq$1 day, $\leq$1 week, $\leq$1 month, $\leq$6 months, >6 months).
- Specialist Expertise: Required attacker knowledge and skills (layman, proficient, expert, multiple experts).
- Knowledge of the Item: Understanding of the target system required (public, restricted, confidential, strictly confidential).
- Window of Opportunity: Time and access availability to execute the attack (unlimited, easy, moderate, difficult/none).
- Equipment: Tools and resources needed (standard to specialised/bespoke).

Step 6: **Risk Determination:** It suggests employing fundamental risk formulas (more formulas were discussed in Gritzalis et al. (2018) [17]) given by `Risk = Impact ×` . Results are then mapped onto a risk matrix (commonly 4×4)

- Low: Acceptable risks requiring minimal action.
- Medium: Moderate risks requiring monitoring.
- High: Significant risks requiring mitigation.
- Extreme/Critical: Unacceptable risks requiring immediate treatment.

Step 7: **Risk Treatment and Cybersecurity goals:** Organisations align their goals with identified (and quantified) risks, determining proper risk treatment strategies ranging from:

- Risk Reduction: Implementing security controls and countermeasures.
- Risk Avoidance: Redesigning systems to eliminate threats.
- Risk Transfer: Sharing risk through insurance or third parties.
- Risk Acceptance: Formally accepting residual risks with justification.

Cybersecurity goals are carefully considered within the approach where specific security requirements guide system design, development, and testing. It is sensible to document these requirements and allocate appropriate resources for handling specific components deemed non-secure or unsafe, hardening systems throughout the development lifecycle.

We next list some of the difficulties of the ISO/SAE 21434:2021 approach:

- **Prioritisation:** the massive attack surface of vehicles (about 1,000-3,000 microchips and up to 150 ECUs operated by up to 150 million lines of code) makes difficult to cover a wide range of attacks over a multitude of possibilities.
- **Interpretation:** The standard allows different teams to make different interpretations to the same items, adding to compliance issues and ambiguity.
- **Weak integration:** It covers and focuses on vehicles, however, it misses out other interacting elements in the infrastructure such as buildings, signs, smart components, user devices, and so on.
- **Actual path analysis:** Uncovering the progression of attacks is tricky as threat actors may have effaced their presence by tampering logs or destroying evidence of their actions. Additionally, they may have acted on someone else's behalf by stealing their identity.
- **Quantification:** The standard requires the assigning of numerical scales for impact (Step 3 – Rating Impact) and severity, attack feasibility (Step 5), and risk calculation using formulas (Step 6). If analysts employ inadequate ranges it might affect treatment and mitigation efforts.
- **Over-simplicity of risk formula:** For determining risk (Step 6) the formula only considers impact and likelihood. There are more nuanced formulas [17] that employ other dimensions such as cost, average loss, and distinguishes assets altogether ('normal', 'critical', and so on).
- **Extensive documentation:** the published document has 88 pages with addenda. It is quite fragmented in its clauses, pinpointing activities that requires interpretation and domain expertise to add value to teams.
- **Standardisation:** It would be beneficial to have the two ISO documents integrated (ISO 31000:2018 and ISO/SAE 21434:2021), ie, where analysts might see how risk gets converged on the approaches. Additionally, the purchase costs for the standards could deter organisations from adopting them.

As mentioned, securing systems is a complex task that involves multiple teams operating at different production levels (hardware and software, electronic components, threat modellers, security officers and teams working in implementation/testing, and so on). The recommendations and guidance are useful for stakeholders to focus efforts and resources towards cybersecurity across OEMs. Next, we direct attention to STRIDE as a valuable TM technique.

2.3 STRIDE Framework for Automotive Systems

This framework is a TM approach that guides analysts/managers/security officers towards the key aspects to observe in systems amenable to cyber-attacks. The premise of using it prompts stakeholders to 'think like attackers', considering more or less the same venues they might employ to get hold of systems. It was proposed by Microsoft in 1999 within the security software development community. Threat Modeling Tool (TMT), a DFD-like notation to explain/convey threats in systems (the tool is still maintained, with some updates throughout the years).

The acronym collapses CIA into the following aspects (described each aspect within brackets below):

- **Spoofing:** [authentication] Impersonating or forging legitimate entities' identities (organisations, business partners, etc) to gain access to systems, data, or spreading malware.
- **Tampering:** [integrity] It involves the malicious or unauthorised modification of data, code, configuration files/properties, system logs, or hardware.
- **Repudiation:** [non-repudiation] Allowing adversaries to perform malicious actions without accountability (allowing plausible deniability), causing flawed/incomplete audit trails, or lacking traceability.
- **Information Disclosure:** [confidentiality] Unauthorised access or exposure of sensitive data by malicious actors.
- **Denial of Service (DoS):** [availability] Forcing systems to overload by abusing calls, abusing, denying, or degrading service to users, or rendering them unavailable by overwhelming resources.
- **Elevation of Privilege:** [authorisation] The act of bypassing security controls for gaining/raising unauthorised access, or by escalating control levels.

In ISO/SAE 21434:2021, STRIDE has a single mention when it lists TM approaches. Table 2.1 shows how to use STRIDE in automotive contexts, with some examples. Bear in mind that the table shows one attack example for each letter (of STRIDE), and that multiple categories could be defined by modellers.

STRIDE is useful in CAV contexts as it presents modellers with a simple to use framework that highlights, in a structured way, key areas for concern when addressing cybersecurity. It is mentioned in

As advantages, it can detect threats and threat actors in early stages of the SDLC, and it can help stakeholders identify areas of interest, and focus efforts. It has a large community of adopters, and there is a host of literature on the topic readily available.

Table 2.1: Examples of attacks in CAVs using the STRIDE approach.

Attack	STRIDE category	Scenario high-level description.
CAN bus abuse	**S**poofing	**Asset:** ECU and hardware (bus).
		Description: Malicious attacker finds a way to inject spurious (fake) CAN messages directly to the braking ECU, causing the vehicle to infer wrong brake requests.
		Impact: [safety] Severe (potential collisions), [financial] High (liabilities)
		Mitigation: Implement CAN message authentication using secure mechanisms and cryptographic message authentication codes.
Modify mileage & user data	**T**ampering	**Asset:** .
		Description: Attacker access to CAV's internal network to modify odometer.
		Impact: [operational] Medium, [financial] High (car can be sold by wrong market value)
		Mitigation: Secure in-vehicle network and sensor authentication coupled with advanced analytics to detect fraud.
Sensor data repudiation	**R**epud.	**Asset:** sensor data logs.
		Description: Driver or third party manipulates or denies sensor logs after accident, undermining forensic analysis and accountability analysis over vehicle's system logs.
		Impact: [financial] High (insurance disputes), [privacy] Moderate.
		Mitigation: Cryptographically signed logs and store using tamper-protection mechanisms.
Break into data telematics	**I**nf. Discl.	**Asset:** data and storage systems.
		Description: Threat actor telematics data using malware or compromised mobile app, or even by intercepted wireless communications.
		Impact: [privacy] Severe, [financial] Moderate
		Mitigation: Employ end-to-end encryption and strong authentication. Track data traversing systems.
Jamming signals	**D**oS	**Asset:** .
		Description:
		Impact: [operational] , [safety] High (car may be involved in accident).
		Mitigation:
Exploit Infotainment	**E**lev. of Priv.	**Asset:** Infotainment ECU.
		Description: Use of remote connection to compromise a vehicle's infotainment system allowing attacker to gain root access privileges.
		Impact: [operational] Medium (perimeter breach), [financial] High (further attacks will take place).
		Mitigation: Segmentation and isolation of critical systems with access to the infotainment module and intrusion detection.

However, in terms of downsides, STRIDE is not a 'silver bullet'; it was originally used for information technology and it extensively focuses on data flows and information security proper-

ties. Additionally, it does not provide a risk scoring to qualify the model and allow prioritisation to occur after analysis. The framework is not easy to automate for a range of potential defects in systems, given its high-level nature of handling and presenting cybersecurity issues altogether. It is thus recommended the approach to be used for specific objectives within a broader TM analysis.

2.4 Overview of techniques for TM

Threat modellers use Attack Modelling Techniques (AMT) to understand system's shortcomings. This active area of research in the cybersecurity community [35] helps stakeholders address cyber-attack progression and perceive malicious tasks performed by adversaries as they abuse systems.

2.4.1 Data Flow Diagrams

DFD is a modelling formalism that draws insights from Graph Theory and was initially conceived in the middle of 1970 as a structured analysis and design technique. One uses simple modelling primitives (DFD Components) such as 'processes', 'flows', 'data stores', and 'terminators' to convey how data traverse systems. Threat modellers employ DFD widely as it depicts most important items deemed for analysis. For instance, they use simple primitives and visual elements to represent situations showing data (or any type of communication) passing through systems. Simple diagrams readily convey ideas for both technical and non-technical team members.

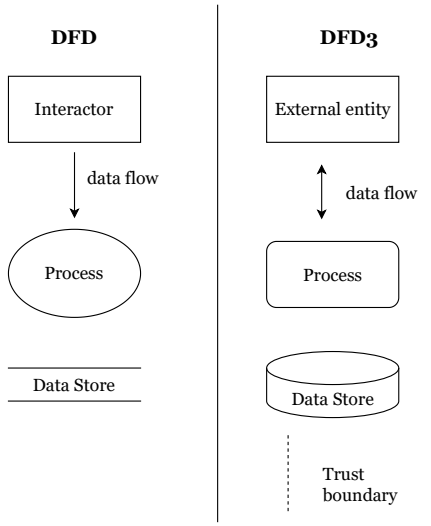


Figure 2.5: Basic modelling primitives for DFD and DFD3.

In the context of cybersecurity, academic and industry practitioners concerned with common sense and expressiveness changed the original DFD formalism main entities and added the notion of "Trust boundaries" in a proposition named DFD3[1], used for TM. Simplicity and objectivity are in the heart of DFD3 design, where rectangles, squared rectangles, drums, arrows (with decorated data items), and dashed lines are sufficient to represent a wide variety of reasonable situations for modellers.

Figure 2.5 shows a DFD and basic primitives[2], ie, "Interactors" or "External entities" (initiators of processes), processes, data flows, and data stores.

As an example of the descriptive and expressive power of DFDs, Figure 2.6 shows a customer purchasing an item on an on-line store that checks inventory.

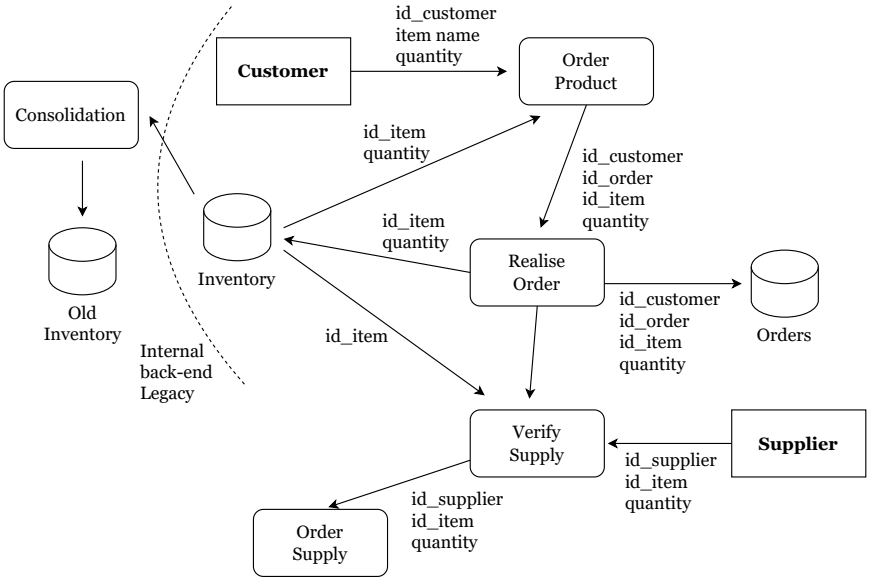


Figure 2.6: Generic "Purchasing a product" model using DFD with inventory checks.

The model allows stakeholders to understand major elements belonging to a system, important processes, how data (and which data) traverses the processes, and how they end up or gets retrieved from data stores. Mind that this is an excerpt of a broader setting, ie, it assumes that customer is logged in, and already selected items to purchase. The model captures the essence of modelling the overall desired process altogether. Understanding how data 'travels' within components allow security analysts to make sure data in-transit and data in-rest assurances, also for inspecting whether or not configuration files could be changed by malicious, incompetent, or 'normal' users (administrators).

[1]Link: `https://github.com/adamshostack/DFD3`.

[2]For generating the figure, modellers could use Diagrams.net (link: `https://app.diagrams.net`) that offers Threat Modelling options. It is a free on-line tool with interesting features.

2.4.2 DFD modelling in CAVs

As mentioned, this formalism's reasoning concerns

. For example, data flows between internal elements (such as users and vehicle systems) and external elements (like sensors and cameras), travelling through the vehicle's communication buses (eg,) to and from the ECUs.

DFDs describe how data moves between external entities, processes, and storage. For CAVs, they are useful to expose interfaces and integration points (sensors, ECUs, cloud, etc.), identify trust and safety boundaries (on-board buses vs. external networks), capture non-functional constraints per flow (bandwidth, latency, reliability), and highlight potential security/attack surfaces (unauthenticated telemetry, OTA channels).

Focusing on modelling efforts, it is appropriate to remind some Upstream's statistics mentioned earlier, stating that

Under these attacks, it is reasonable to think that one should focus on communication issues, authorisation, and data privacy. And security analysts should overview the DFD models under the prism of CIA triad, where for instance communication issues may arise when availability or integrity attacks are perpetrated by adversaries.

Typical DFD elements comprise:

- **External entities:** Communication, external (cloud) services, other vehicles (V2V), infrastructure (V2I), remote operators/OEMs.
  - **Communication Interfaces:** V2X communications, CAN bus systems, Wireless connections, Bluetooth interfaces, Cellular networks.
  - **Software Systems:** present in ECUs, infotainment systems, telematics modules, or OTA.
  - **Sensors and External Systems:** vehicle sensors, cameras, navigation systems, charging infrastructure.
  - **Physical Access Points:** OBD port, USB ports, Infotainment systems, Remote door lock systems, Key fob signals .
- **Processes:** Data collection (from sensors, cameras, ECUs, and so on), data processing, sensor fusion mechanisms, modules responsible for decisions (braking, avoiding obstacles, steering, etc), perception, planning, control, OTA manager, or where actions get carried out.
- **Data stores:** sensor/customer/vehicle specialised databases, internal repositories, on-vehicle logs and metadata catalogues, training data stores, local hard disk maps, or operational databases.
- **Data flows:** Raw sensor streams, diagnostics, OTA updates, event logging, vehicle's statuses update, data synchronisation, alerts and notifications, fused perception messages, telemetry, or control commands.

Example flows (context-level):

- Sensors (LiDAR/camera) → Sensor Preprocessing → Sensor Fusion → Perception.
- Perception → Trajectory Planner → Control → Actuators.

- Vehicle Diagnostics → Telemetry Upload ↔ Cloud Analytics; Cloud → OTA Update → Vehicle Update Manager.

Modelling guidance:

- Use layered diagrams (context → level-1 → level-2) so each diagram stays readable.
- Annotate flows with bandwidth/latency and security requirements.
- Explicitly show trust boundaries and cryptographic/authentication checkpoints.
- Avoid mixing safety-critical control loops with non-real-time data paths without noting timing constraints.

2.4.3 Attack Trees

ATs were first mentioned by Bruce Schneier, in 1999, in Dr. Dobbs Journal (a computing magazine) [20]. It consists of creating a hierarchical view of the tasks required to attack a system. The attack is the 'Root Node' and the ways of realising attacks are in the 'Leaf Nodes'. Figure 2.7 shows the building blocks of a basic Attack Tree and its modelling primitives.

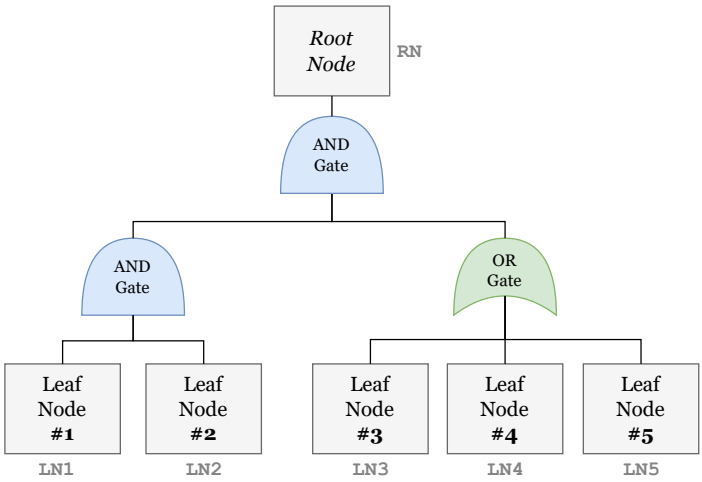


Figure 2.7: An AT and basic modelling primitives.

A basic tree has the following components:

- **Root Node:** this is normally related to the issue under study, eg, "DoS Attack in CAV", or "Steal vehicle credentials".
- **Logic Gates:** in the basic version it could be i) AND Gate; or ii) OR Gate, each one representing which nodes to be true. For AND, all underlying Leaf Nodes must be true whereas for OR Gate, only one Leaf Node must be true.
- **Transitions:** connects nodes to logic gates.
- **Leaf Nodes:** used to represent actions in the model.

For the specific example in the figure, one could work out a logic formula that states:

```
RN = (LN1 AND LN2) AND (LN3 OR LN4 OR LN5)
```

So, for the example, in order for the Root Node to 'be true', LN1 and LN2 must also be true, and either LN3, LN4, or LN5 could be true. A lot of different reasoning could be derived by modellers using these basic primitives, which is the strength of this kind of modelling. ATs are useful for understanding 'what might go wrong?' and also, it visually depicts attack paths. Additionally, stakeholders could easily understand steps that adversaries may take and then create or devise protections to prevent those tasks from happening. For instance, if a security officer creates cybersecurity controls to prevent LN1 from happening, it will render it 'false', thus collapsing the attack altogether. Obviously that preventing LN1 is not a trivial task to ensure, as threat actors might find different mechanisms to circumvent it.

Figure 2.8 shows an example of AT to 'Learn Root Password' of a system:

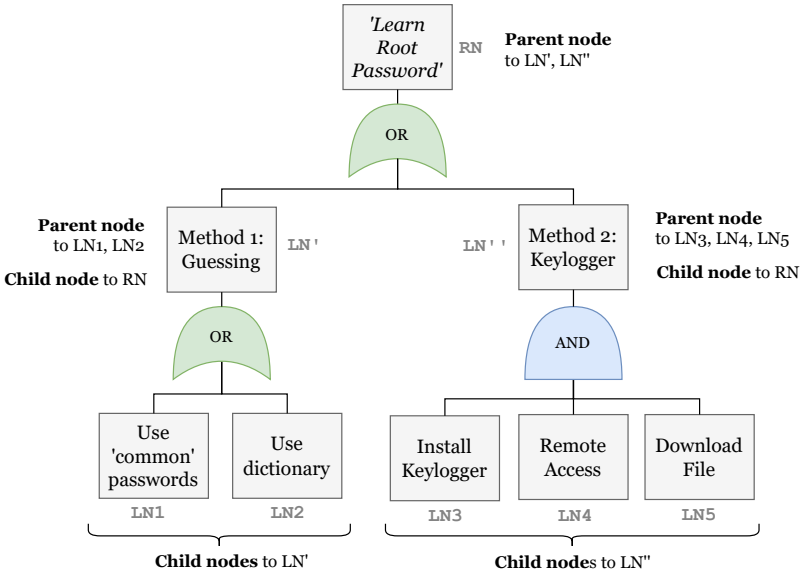


Figure 2.8: Working example of AT (nodes for describing methods are optional).

The figure also depicts the hierarchy, explaining how *Parent nodes* (a node with leaves, that belongs to a higher hierarchical level) and *Child nodes* (a node that has a Parent node) work in the model. For this example, the logical formula is:

$$\text{RN} = (\underbrace{\text{LN1 OR LN2}}_{\text{Method 1 (LN')}}) \text{ OR } (\underbrace{\text{LN3 AND LN4 AND LN5}}_{\text{Method 2 (LN'')}})$$

For this example, RN = `Method 1 OR Method 2`, ie, an adversary may either guess the password or install a keylogger. The formula is representing the fact that an adversary is either

successful by guessing it (a simple task) or by using a method that installs, access, and download a password, a more difficult task. A security officer making sure that software installing should be hardened, it collapses the whole branch of "Method 2".

Over time, modellers started thinking about some shortcomings of ATs, for instance, the need for have some kind of ordering among tasks, or decorating leaves with other information such as 'effort' or 'cost', to name a few. Also, it would be interesting to consider variants into the formalism. For example, incorporating time, allowing other logic gates to be used (eg, XOR, etc), or to have 'Attack' and 'Defense' tasks altogether, or mapping countermeasures and mitigations. These additions offer superior analysis methods for modellers that allow them to create representations for more complex and nuanced cyber-attack possibilities. We shall describe these modifications next.

Ordered logic gates: the idea is to represent order in tasks. Let's briefly recap how logic gates work (up until now):

- **OR gate:** only one leaf must be realised (successful) to achieve the parent node.
- **AND gate:** all leaves connected to the Parent node must be successful for the Parent node to be successful as well.

In ordered reasoning, we shall accept the following new possibility:

- **Sequential AND (SAND):** All Child nodes must be successful, however, they must occur in a specific order (usually left-to-right).

Figure 2.9 shows how SAND gate works in a model excerpt. The leaf nodes connected to the SAND gate must occur in the specific order (left-to-right) of 1. Connect to customer's network; followed by 2. Listen to connection.

This primitive enriches the modelling possibilities for ATs, allowing modellers to model intricate adversarial behaviours. The logical formula is:

$$\text{RN} = (\overrightarrow{\text{LN1 AND LN2}}) \text{ OR } (\dots)$$

Decorating leaf nodes: in this modelling primitive, leaves can assign quantitative (ie, numerical values) or qualitative (eg, scales such as 'Low', 'Medium', or 'High') indicators to qualify "Effort", "Difficulty" (likelihood), "Cost", "Skill", or "Impact to organisation".

Figure 2.10 represents an AT with decorated leaves. Note that for this example all modifiers are qualitative where the key are: { Cost, Skill, Feasibility }.

The logical formula for this model is:

$$\text{RN} = (\underbrace{\text{LN3 OR LN4 OR LN5}}_{\text{LN1}}) \text{ OR } (\underbrace{\overrightarrow{\text{LN6 AND LN7 AND LN8}}}_{\text{LN2}})$$

Associating a defense task to an attack: At the same time that leaves represent attacks, one could decorate the borders of such task in 'Red' whereas 'Green' borders would represent defense mechanisms to prevent or avoid malicious activities. Figure 2.11 shows an excerpt of an

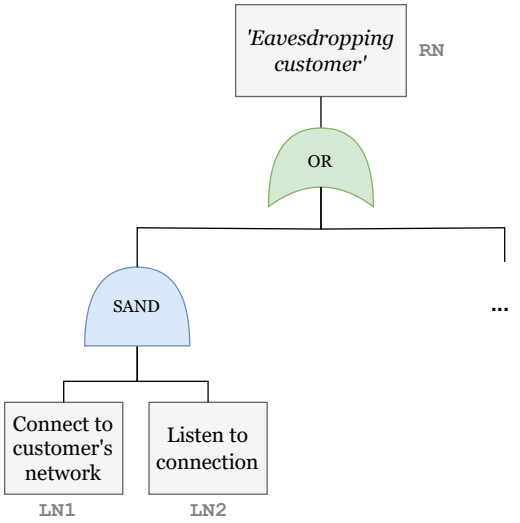


Figure 2.9: An AT that uses a SAND gate to represent ordered events.

Attack-Defense Tree showing attacks (in red) and a defense mechanism (in green) that employs Multi-Factor Authentication (MFA).

There are several other modifications that could be appended to AT that were not mentioned here. We urge readers to investigate about such representations that captures other interesting adversarial behaviours.

2.4.4 Attack Graphs

A graph is a structure that represents relationships between modelled objects. A bit more formally, a graph G consists of the following formula G = (V, E) where:

- V: a set of vertices (or nodes);
- E: a set of edges, that connect pairs of vertices, ie, $E \subseteq \{(u, v) \mid u, v \in V\}$.

Graphs can be one of many types, for example, *Simple Graphs* have no loops or multiple edges, *Directed Graphs (Digraphs)* are structures where all edges have a specific direction from one vertex to another, or even *Weighted Graphs*, a graph where edges have weights (decorated with numerical values representing among other possibilities, cost, or length.

They are used in cybersecurity to represent cyber-attacks, adversaries, and vulnerable assets. As an example, we shall comment on the Stuxnet attack (discovered in 2010) which caused severe damage to the Iranian nuclear program targeting Industrial Control Systems (ICS). A host of work was discussed in previous literature outlining how AGs can be used to model the attack and how they can be used to investigate incidents.

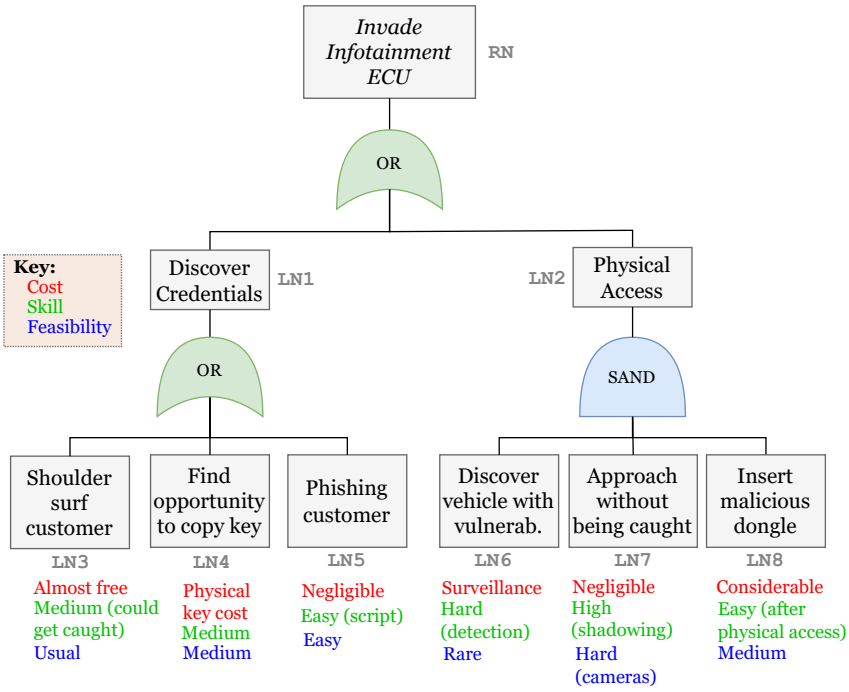


Figure 2.10: Decorating leaves of an AT with qualitative modifiers.

Figure 2.12 shows two examples of using graphs to model systems: Figure 2.12a shows the Stuxnet worm venues of spreading modelled as a graph [36] whereas Figure 2.12b represents a generic configuration of dispersed assets over user, corporate, and legacy networks interacting.

The Stuxnet attack had multiple ways of compromising assets, eg, through removable devices or by injecting malware at printers, allowing the worm to quickly spread and reach its final target, which was an air-gapped network.

There is a myriad of interesting materials about the Stuxnet attack and how to model it using AGs available in recent literature. We urge readers to investigate further about it as it has been extensively scrutinised by the cybersecurity community throughout the years, with interesting insights and lessons learned.

One could model potential adversaries attempting to access different systems facing the Internet, and then how they might move laterally by gaining access to internal resources and

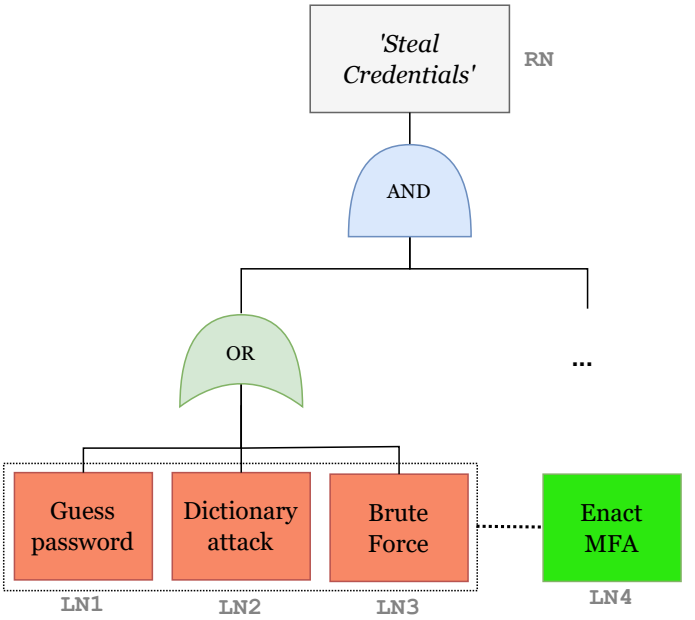
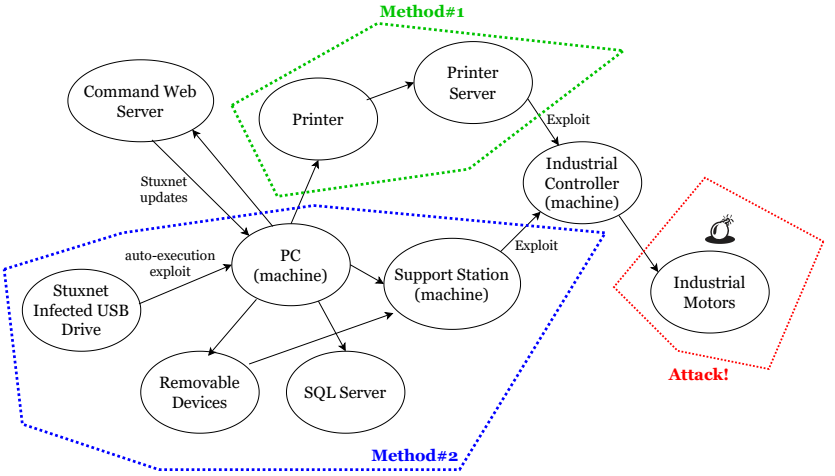
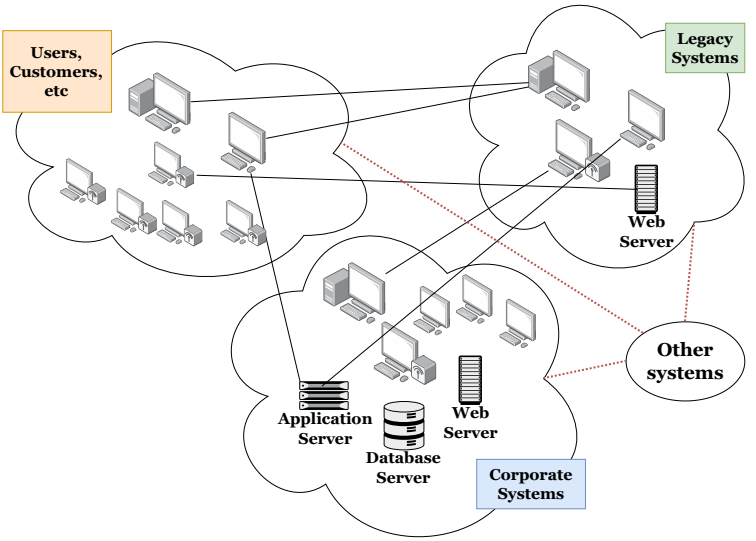


Figure 2.11: An excerpt for an Attack-Defense Tree showing a defense node (MFA).

air-gapped systems.



(a) Representing the Stuxnet attack as a graph.



(b) Using a graph to model a system with multiple assets and networks.

Figure 2.12: Examples of modelling systems with graphs.

Chapter 3

Perspectives for threat analysis of CAVs

The threat landscape for CAVs continuously evolves, requiring analysts to alter the ways in how they employ threat modelling methodologies for the full vehicle's life-cycle. As vehicles transition from isolated mechanical systems to complex, interconnected cyber-physical entities, they must enhance traditional approaches to account for multi-layered architectures spanning onboard sensors, edge computing nodes, cloud infrastructure, and V2X communication networks. Future TM frameworks must adopt a holistic perspective that simultaneously addresses vulnerabilities across sensor systems, ECUs, communication protocols, and artificial intelligence-driven decision-making processes, recognising that a breach at any single point can cascade through the entire system with potentially catastrophic safety consequences.

, we have highlighted some useful existing TM approaches that can be adapted for CAV. It served as a starting point for understanding how to approach the complex security challenges posed by autonomous vehicles. Given the sheer number of components belonging to the attack surface of vehicles, significant gaps remain in current TM practices when applied to the unique characteristics of CAVs. to open the discussion for future research and development in this area, encouraging the community to explore novel methodologies that can better capture the intricacies of these systems altogether.

- Security underpinnings of CAVs and the need for dynamic and adaptive TM approaches that can evolve alongside the vehicle's software and hardware configurations, with the possibility (yet to be explored) incorporating real-time CTI to address emerging vulnerabilities.
- A comparison between different standards related to CAV.
- The importance of integrating safety and security considerations within threat models, recognising that attacks on CAVs can have direct implications on passenger safety and public trust in autonomous technologies.
- The potential for leveraging novel TM techniques, approaches, and tools, for instance, AI/ML that can be used to automate the modelling process and enabling more efficient

identification and prioritisation of risks in complex vehicle systems.
- The necessity for collaborative frameworks that involve stakeholders from automotive manufacturers, cybersecurity experts, regulatory bodies, and end-users to ensure comprehensive threat assessments that reflect diverse perspectives and expertise.
- The value of continuous monitoring and feedback loops in threat modelling processes, allowing for ongoing refinement and adaptation of security measures as new threats emerge and vehicle technologies evolve.

3.1 Future directions for CAV threat modelling

There is a huge discrepancy in how TM was conceived to operate and how it should operate nowadays, taking into consideration the demands and the inter-connectedness of modern vehicle fleets. For example, traditional frameworks like STRIDE were not designed to address these characteristics. As vehicles increasingly rely on a plethora of different sets of models for perception, planning, and control decisions, threat modellers must develop equally sophisticated methods to assess adversarial attacks on a myriad of systems having multiple points of entry and vulnerabilities. For instance, data poisoning threats during model training might become a true battleground for assessing how complex algorithms are coping with threats. On the other hand, prompt injection attacks against voice assistance systems might trick a CAV system to underperform or yield incorrect outcomes.

The greatest emerging security threat perhaps lies in data poisoning, where malicious or corrupted training data compromises the integrity of AI models, potentially causing vehicles to misinterpret environmental cues or make unsafe decisions under specific conditions. Future approaches will have to incorporate data provenance and tracking altogether, fostering automated validation of software system combined with anomaly detection modules to ensure that the training data underpinning vehicle autonomy remains trustworthy and hard to easily corrupt.

Another point that we raise here is the need for having a convergence of standardisation, regulatory frameworks, and advanced TM tools. As regulatory bodies increasingly mandate cybersecurity standards—such as those outlined in frameworks like ISO/SAE 21434 and emerging WP.29 (from UNECE/UN) regulations, that state that TM approaches must be embedded as a mandatory component of the vehicle development lifecycle sooner rather than later in the process.

As vehicle architectures become increasingly distributed, with critical processing occurring both at the edge (within the vehicle) and in the cloud, threat models must evolve to capture latency-sensitive dependencies and the unique risks posed by OTA updates, EV charging infrastructure integration, and software-defined vehicles. The vehicles of the future will demand TM practitioners that offers hybrid expertise spanning automotive systems engineering, cybersecurity, artificial intelligence, and data science, ensuring that security by design principles permeate every layer of autonomous vehicle architecture from conception through deployment and

throughout the operational lifecycle.

About the author



Ricardo M. Czekster works with dependable and secure computing systems. He is a Lecturer (Associate Professor) at the School of Computer Science and Digital Technologies at Aston University, at Birmingham/UK. He received his BSc. in Computer Science from Pontifical Catholic University of Rio Grande do Sul (PUCRS), Brazil, in 2002, followed by a MSc. in Computer Science, and a PhD. in Computer Science from the same institution in 2006 and 2010, respectively. Before joining Aston University in 2022, he was a Research Fellow at Newcastle University, UK, working on projects related to cybersecurity for Active Buildings.

His research interests include formal methods, security and privacy, cyber-physical systems, and Internet-of-Things, to mention a few. He is a prolific researcher publishing in high impact international conferences and journals.

This book was a product of insomnia, curiosity, and the desire to contribute to the community. Those wee hours spent writing and editing the content were rewarding and fulfilling, and I hope you find the material useful in your journey towards understanding TM for CAVs.

Ricardo M. Czekster
Birmingham, October 2026 (Second Edition)

This page is intentionally left blank.

Bibliography

[1] SAE International, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," SAE International, Tech. Rep. J3016_202104, Apr. 2021. [Online]. Available: https://www.sae.org/standards/content/j3016_202104/

[2] ISO, SAE, "ISO/SAE 21434: Road vehicles – Cybersecurity engineering," 2021.

[3] A. Avižienis, J. C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, 2004.

[4] National Institute of Standards and Technology. An Introduction to Information Security. [Online]. Available: https://csrc.nist.gov/pubs/sp/800/12/r1/final

[5] S. Xu, M. Yung, and J. Wang, "Seeking foundations for the science of cyber security: Editorial for special issue of information systems frontiers," *Information Systems Frontiers*, vol. 23, no. 2, pp. 263–267, 2021.

[6] National Institute of Standards and Technology, "Guide for Conducting Risk Assessments," Gaithersburg, MD: National Institute of Standards and Technology, Tech. Rep. NIST.SP.800-30r1, September 2012. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-30r1

[7] S. Parkinson, P. Ward, K. Wilson, and J. Miller, "Cyber threats facing autonomous and connected vehicles: Future challenges," *IEEE transactions on intelligent transportation systems*, vol. 18, no. 11, pp. 2898–2915, 2017.

[8] X. Sun, F. R. Yu, and P. Zhang, "A survey on cyber-security of connected and autonomous vehicles (cavs)," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6240–6259, 2021.

[9] R. Abreu, F. Branco, M. J. Reis, and C. Serôdio, "Cybersecurity in Connected and Autonomous Vehicles: A Systematic Review of Automotive Security," *IEEE Access*, 2025.

[10] A. Shostack, *Threat modeling: Designing for security.* John Wiley & Sons, 2014.

[11] A. V. Uzunov and E. B. Fernandez, "An extensible pattern-based library and taxonomy of security threats for distributed systems," *Computer Standards & Interfaces*, vol. 36, no. 4, pp. 734–747, 2014.

[12] I. Tarandach and M. J. Coles, *Threat Modeling*. O'Reilly Media, Inc., 2020.

[13] International Organization for Standardization. (2018) Risk management – guidelines. [Online]. Available: https://www.iso.org/standard/65694.html

[14] C. J. Alberts and A. J. Dorofee, *Managing information security risks: the OCTAVE approach*. Addison-Wesley Professional, 2003.

[15] R. S. Ross, "Guide for conducting risk assessments (NIST. SP 800-30rev1)," *Joint Task Force Transformation Initiative, The National Institute of Standards and Technology (NIST), Gaithersburg*, 2012.

[16] IEC 31010:2019, "Risk management – Risk assessment techniques," https://www.iso.org/standard/72140.html, 2019.

[17] D. Gritzalis, G. Iseppi, A. Mylonas, and V. Stavrou, "Exiting the risk assessment maze: A meta-survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 1, pp. 1–30, 2018.

[18] T. UcedaVelez and M. M. Morana, *Risk Centric Threat Modeling: process for attack simulation and threat analysis*. : John Wiley & Sons, 2015.

[19] K. Wuyts, L. Sion, and W. Joosen, "LINDDUN GO: A Lightweight Approach to Privacy Threat Modeling," in *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 2020, pp. 302–309.

[20] B. Schneier, "Attack trees," *Dr. Dobb's journal*, vol. 24, no. 12, pp. 21–29, 1999.

[21] V. Saini, Q. Duan, and V. Paruchuri, "Threat modeling using attack trees," *Journal of Computing Sciences in Colleges*, vol. 23, no. 4, pp. 124–131, 2008.

[22] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. M. Wing, "Automated generation and analysis of attack graphs," in *Proceedings 2002 IEEE Symposium on Security and Privacy*. IEEE, 2002, pp. 273–284.

[23] R. P. Lippmann and K. W. Ingols, "An annotated review of past papers on attack graphs," MIT Lincoln Laboratory, Tech. Rep., 2005.

[24] M. S. Lund, B. Solhaug, and K. Stølen, *Model-driven risk analysis: the CORAS approach*. Springer Science & Business Media, 2010.

[25] W. Xiong and R. Lagerström, "Threat modeling–A systematic literature review," *Computers & security*, vol. 84, pp. 53–69, 2019.

[26] R. M. Czekster, "Showcasing standards and approaches for cybersecurity, safety, and privacy issues in connected and autonomous vehicles," *arXiv preprint arXiv:2508.01207*, 2025.

[27] G. Macher, C. Schmittner, O. Veledar, and E. Brenner, "ISO/SAE DIS 21434 automotive cybersecurity standard – in a nutshell," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2020, pp. 123–135.

[28] SAE J3061, "Cybersecurity Guidebook for Cyber-Physical Vehicle Systems," https://www.sae.org/standards/content/j3061_201601/, 2016.

[29] C. Schmittner, Z. Ma, C. Reyes, O. Dillinger, and P. Puschner, "Using SAE J3061 for automotive security requirement engineering," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2016, pp. 157–170.

[30] UNECE WP.29/R155, "Uniform provisions concerning the approval of vehicles with regards to cyber security and cyber security management system," UN Regulation, Tech. Rep., 2021.

[31] UNECE WP.29/R156, "Uniform provisions concerning the approval of vehicles with regards to software update and software updates management system," UN Regulation, Tech. Rep., 2021.

[32] ISO 21448:2022, "Road vehicles – Safety of the intended functionality," https://www.iso.org/standard/77490.html, 2022.

[33] M. Benyahya, T. Lenard, A. Collen, and N. A. Nijdam, "A systematic review of threat analysis and risk assessment methodologies for connected and automated vehicles," in *Proceedings of the 18th international conference on availability, reliability and security*, 2023, pp. 1–10.

[34] M. Benyahya, A. Collen, T. Lenard, and N. A. Nijdam, "TARA 2.0 for Connected and Automated Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 2025.

[35] H. S. Lallie, K. Debattista, and J. Bal, "A review of attack graph and attack tree visual syntax in cyber security," *Computer Science Review*, vol. 35, p. 100219, 2020.

[36] J. T. Seo, "Towards the advanced security architecture for microgrid systems and applications," *The Journal of Supercomputing*, vol. 72, no. 9, pp. 3535–3548, 2016.

[37] Auto-ISAC, "Automotive threat matrix (ATM)," Online, 2023, accessed on Oct/2026. [Online]. Available: https://atm.automotiveisac.com/



Yet another venture by *the lazy panda collection*
URL: `https://pixabay.com/photos/mammal-wildlife-animal-zoo-panda-3074618/`
Simplified Pixabay License

Introduction to Threat Modelling applied to Connected and Autonomous Vehicles
Copyright © 2026 Ricardo M. Czekster
License: Creative Commons Attribution 4.0 International (CC BY 4.0)

What people are saying?

- “Incredible array of words and ideas!”, Bob Woodkirk.
- Bill Raymul has commented: “This book sits on my night stand as I keenly enjoy reading it every day.”