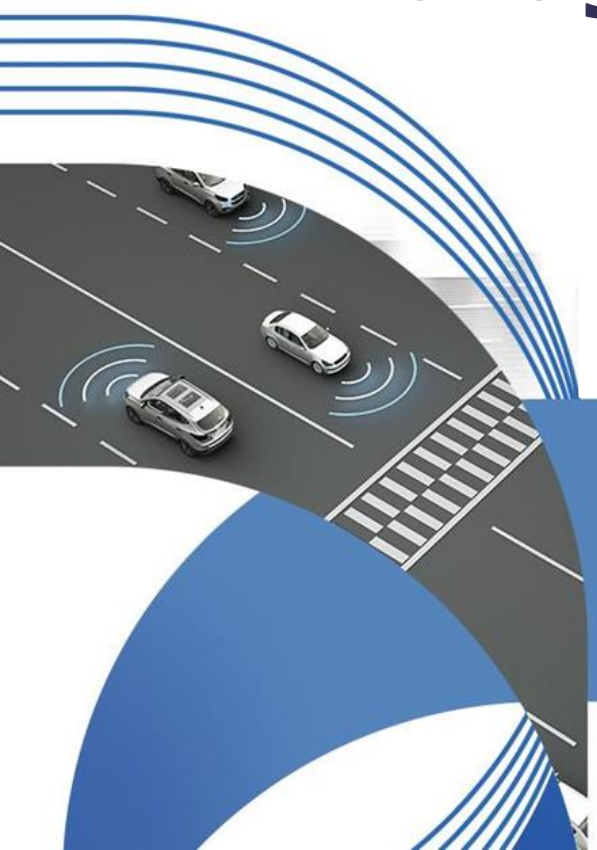
|Inaugural Symposium on Vehicle Security and Privacy (VehicleSec 2023)|

Formally Verified Software Update Management System in Automotive



Jaewan Seo, Jiwon Kwak, Seungjoo Kim†

† Corresponding Author

This work was partly supported by Institute of Information communications Technology Planning Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2018-0-00532, Development of High-Assurance(EAL6) Secure Microkernel, 50) and Korea Agency for Infrastructure Technology Advancement(KAIA) grant funded by the Ministry of Land, Infrastructure and Transport (Grant 23AMDP-C162334-03, Research and Development on Integrated Automotive Cybersecurity Assurance Technologies, 50)

고려대학교 KOREA UNIVERSITY

Who are we ?

Who are we ?

ABOUT US

Jaewan Seo
Korea University
bace@korea.ac.kr

Jaewan Seo is a graduate student(Master Course) at SANE Lab in Korea University. His research interest is automotive security, security engineering, threat modeling, security assessment

Jiwon Kwak
Korea University
jwkwak4031@korea.ac.kr

Jiwon Kwak is a graduate student(Ph.D. Course) at SANE Lab in Korea University. His research interest is security engineering, high-assurance system, threat modeling, formal method

Who are we ?

ABOUT US

Seungjoo Kim

(Corresponding Author)

Korea University

skim71@korea.ac.kr

Seungjoo (Gabriel) Kim has been a professor at the School of Cybersecurity in Korea University from 2011. For the past 7 years he was an associate professor in Sungkyunkwan University and had 5 years of background as a team leader of KISA(Korea Internet & Security Agency).

In addition to being a professor, he is a director of AR^{2}C(Army RMF Research Center), a director of CHAOS(Center for High-Assurance Operating Systems), a head of SANE(Security Assessment aNd Engineering) Lab, an adviser of undergraduate hacking club CyKor(DEFCON CTF 2015 & 2018 winner) at Korea University, and a founder/advisory director of an international security & hacking conference SECUINSIDE. Since 2018, he has been a review board member of Black Hat Asia.

His research interests lie primarily in building "inherently secure, high-assurance, and provably secure systems and architectures" & "composable and scalable secure systems".

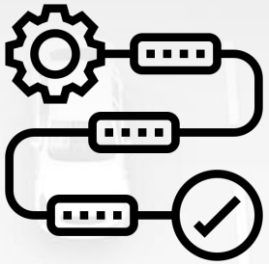
Topic

01

Motivation
&
Our research

The reason why we conducted the Research
&
Explain how we do that

02



Threat Modeling

Explain the process and result of threat modeling

03

Formal Method

Explain the formal model of our proposal
&
Verification result

04

Conclusion

Briefly conclude our research
&
Future research

Motivation

Automotive Cybersecurity

Fiat Chrysler recall 1.4m vehicles !

2015

BlackHat USA 2015
Remote Exploitation of an Unaltered Passenger Vehicle

2016

Steelcon 2016
Hacking the Nissan Leaf

Tesla recalls 135,000 Model X, Model S for safety issues !

2017

BlackHat USA 2017
Over-The-Air: How We Remotely Compromised the Gateway, BCM, and Autopilot ECUs of Tesla Cars

2018

BlackHat Europe 2018
Tool(PASTA: Portable Automotive Security Testbed) for researchers and budding car hacking experts is suggested.

2019

Autosec 2019
Modern vehicle hacking

2020

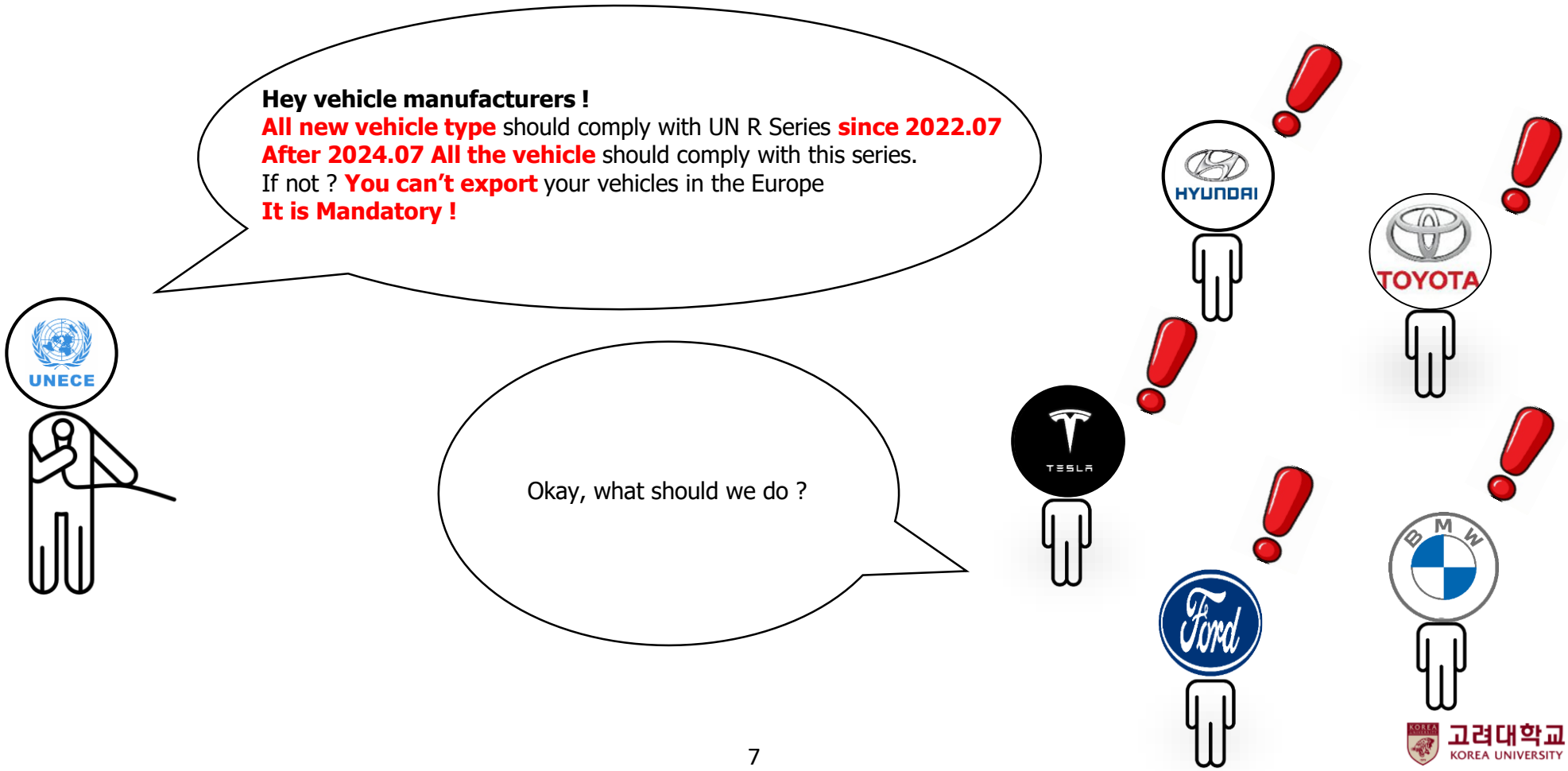
The consumer group “Which?” exposes security flaws that range from remotely exposing private, customer information in Ford, VW.

UN Regulation is announced !

Motivation

UN Regulation

- To ensure cyber security for vehicles, UNECE has announced that OEMs must comply with a series of UN Regulations.



Motivation

UN Regulation

- It is too difficult for OEMs to develop vehicles following UN Regulation series

According to UN R series

⋮

the vehicle manufacturer shall demonstrate:
The process they will use to ensure that software updates will be protected to reasonably prevent **manipulation** before the update process is initiated

⋮

Which location does **manipulation** occur ?

Wireless network ? CAN ?
In ECU ?

⋮

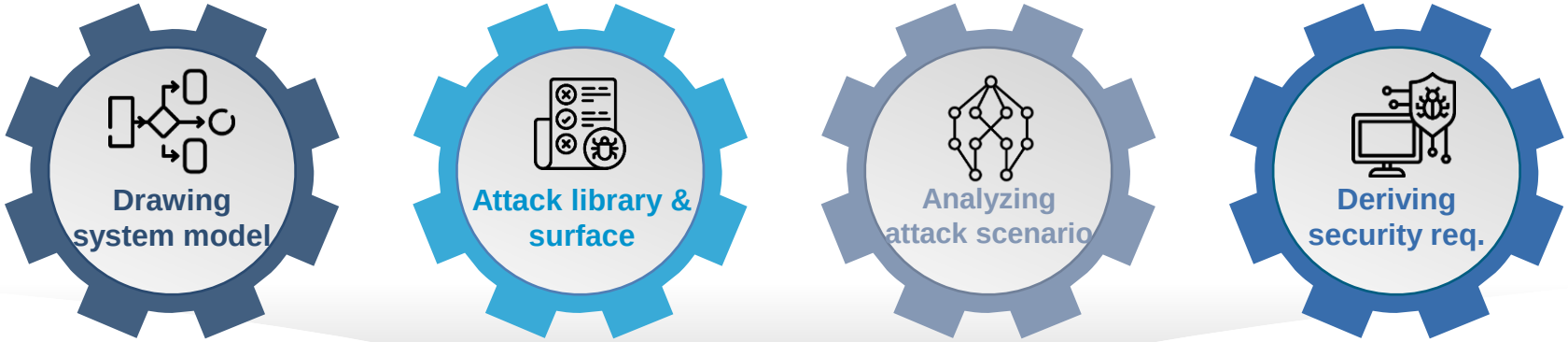
When should we apply these countermeasures to a product in the development lifecycle? In design ? Implementation phase ?

⋮

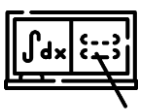
Which countermeasure are suitable for mitigating **manipulation** in acceptable level ? Is it **sufficient** ?

Our Research

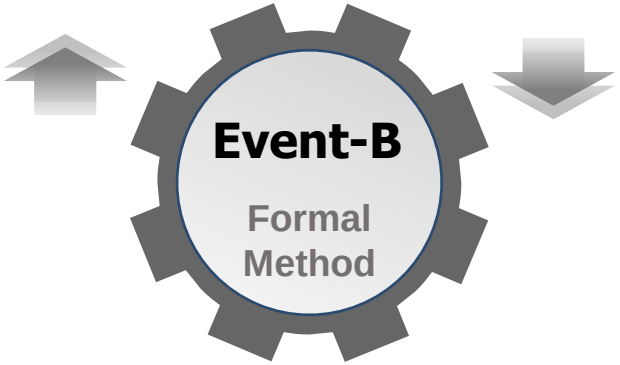
Overview



Designing Secure SUMS Architecture



Formal Specification

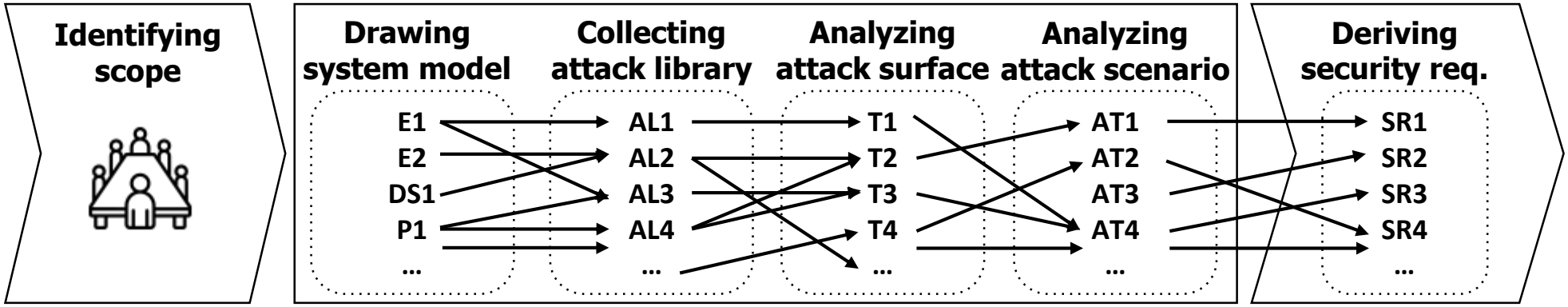


Formal Verification

Threat Modeling

What is Threat Modeling?

- Structured process to identify known vulnerabilities, threats and derive security requirements systemically



Stage	Description
Identifying scope	Identifying assets in the target system (**What should be protected** from malicious users?)
Drawing system model	Simplifying the structure of the target system through modeling techniques based on data flow (**How does** your target system work?)
Collecting attack library	Collecting known vulnerabilities, weaknesses, and attacks related to target system (**What happened** in the target system before?)
Analyzing attack surface(STRIDE)	Identifying attack surface for target system component (**What could happen**?)
Analyzing attack scenario	Analyzing how identified threats are systemized to damage the target system (**How does a hacker attack** target systems??)
Deriving security requirements	Deriving security requirements to mitigate attack scenarios in the target system (**How does a security expert** keep target systems safe from hackers?)

※ **E**: Entity, **DS**: Data Store, **P**: Process, **AL**: Attack Library, **T**: Threat, **AT**: Attack Tree, **SR**: Security Functional Requirement

Threat Modeling

Identifying scope

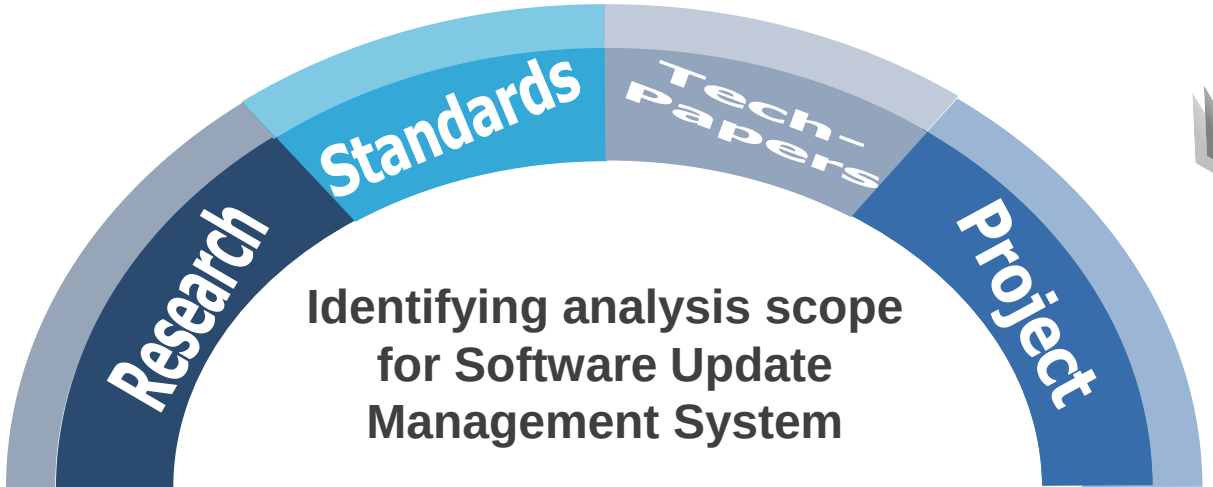
- Identifying components based on 24 documents that are published in ITU-T, SAE, COVESA, UN R156, related papers, etc.

Standards
- ITU-T
- ISO/IEC

Technical Papers
- SAE J3061
- PAS 1885

Research Papers
- BlackHat/DEFCON
- ESCAR
- Autosec (Vehiclesec)

Project Deliverables
- COVESA(Connected Vehicle Systems Alliance)
- Automotive Grade Linux



Threat Modeling

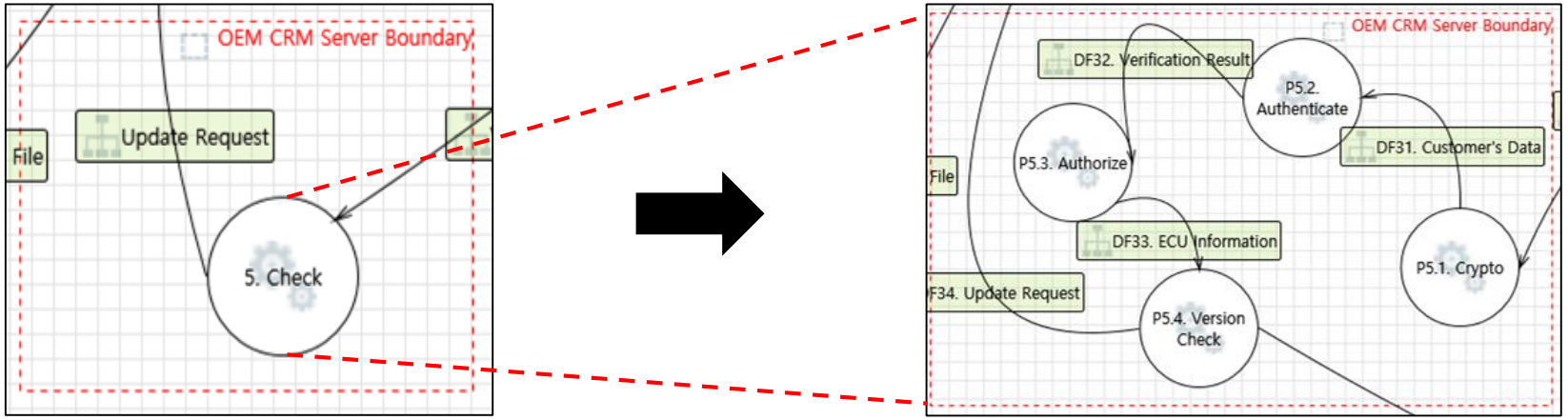
Drawing system model

- Abstraction is to remove the rest only with the essential information related to (vulnerabilities) analysis on the design map, which can improve the concentration and efficiency of the analysis

Threat Modeling

Drawing system model

- Context Level DFD: Expresses the entire system based on **roles** such as the OEM, client, and auto repair shop
- Level 0 DFD: Decomposes its role based on the **objective** of its components, such as development, updates, and customer management
- Level 1 DFD: Decomposes its components based on their **functionalities** such as encryption and signing
- Level 2 DFD: Decomposes its functionalities based on further **criteria**(API in source code)

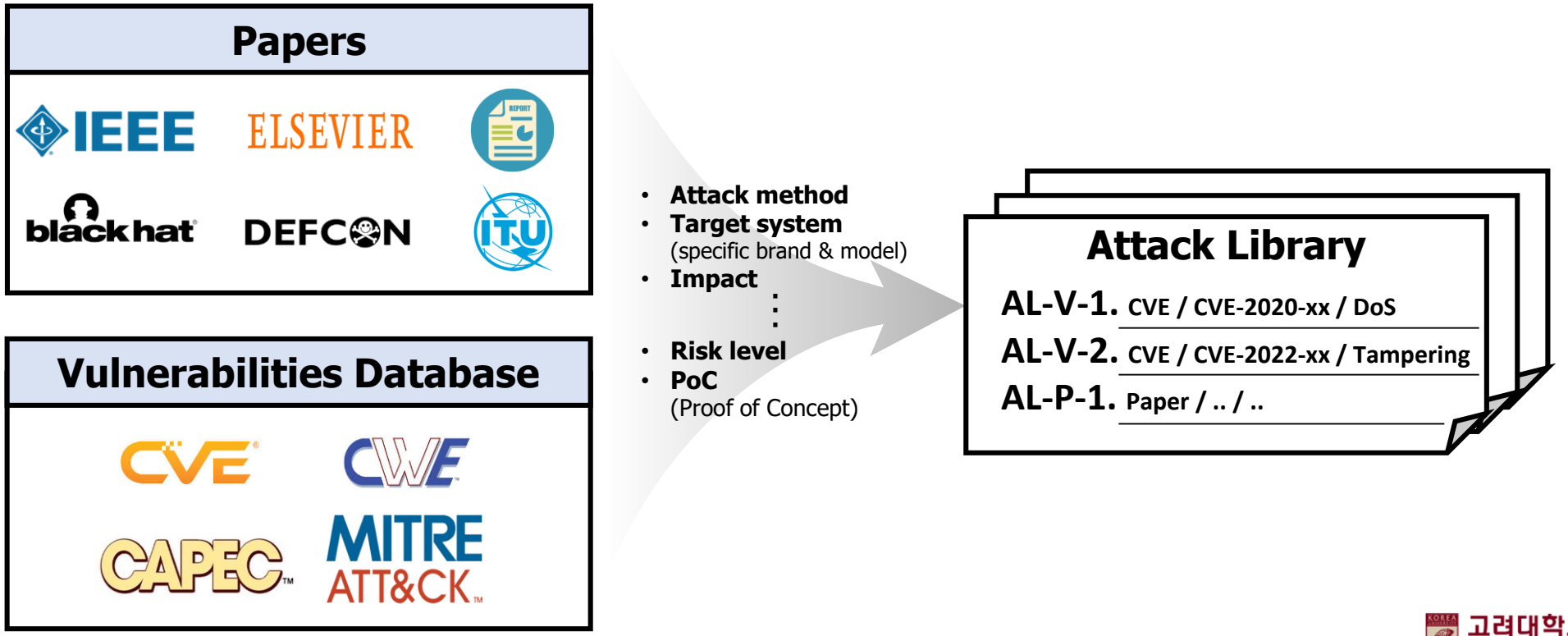


Context Level DFD for SUMS **Level 0 DFD for SUMS**

Threat Modeling

Collecting attack library

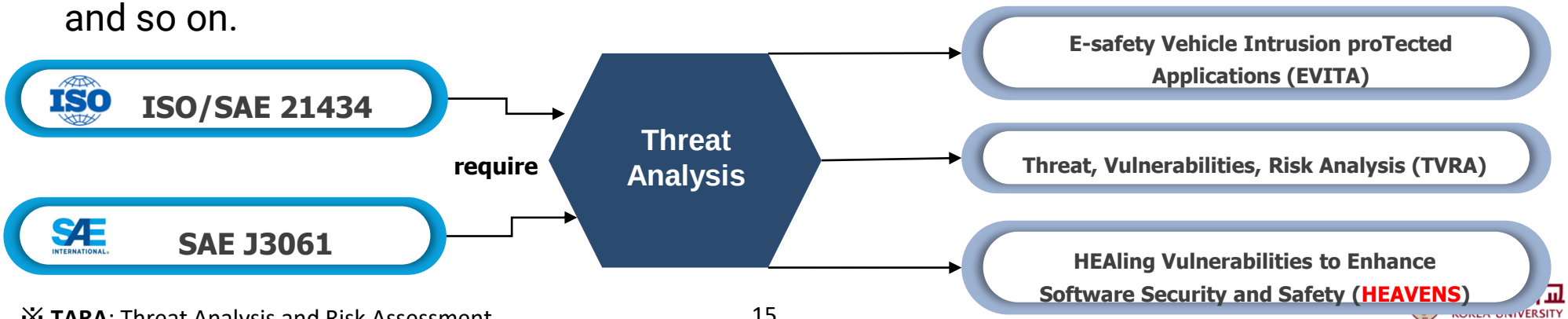
- Collects information on vulnerabilities known so far in relation to the analysis target and makes it into a database
  - **41** vulnerabilities & weaknesses (Source: CVE/CWE)
  - **29** vulnerabilities & threats (Source: Paper/Standard/Technical Report)



Threat Modeling

Analyzing attack surface (STRIDE)

- Standards for cybersecurity in the automotive domain recommend the usage of TARA when conducting threat analysis and risk assessment
  - TVRA(2011) analyzes actual attack scenarios that can occur in the automotive & evaluate their risk based on attack potential in ISO 18045
  - EVITA(2011) analyzes threat scenarios and evaluates risks, but it mainly focuses on whether scenarios and risks affected to performance of the entire system
  - **HEAVENS(2012)** analyze threats in vehicles based on Microsoft's STRIDE and evaluates the risk level by calculating “Threat Level”, and “Impact Level”
- As the number of software installed in vehicles has significantly increased, the STRIDE-based threat modeling techniques, are widely used in the OEMs such as BMW, Hyundai, and so on.



※ **TARA**: Threat Analysis and Risk Assessment

Threat Modeling

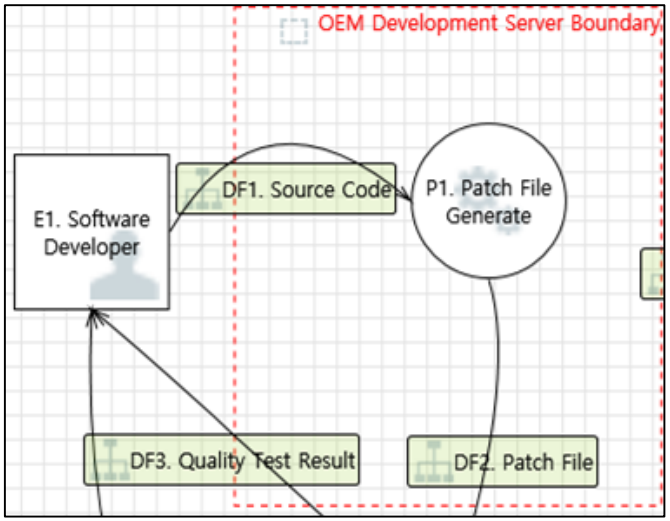
Analyzing attack surface (STRIDE)

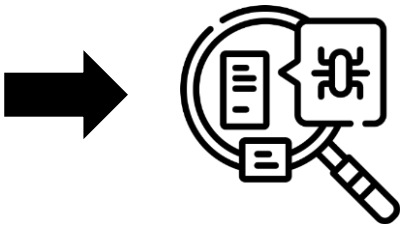
- Based on the rules in STRIDE 1007 threats are identified in our system model
  - STRIDE-per-Interaction
  - STRIDE-per-Element

STRIDE-per-Interaction

	S	T	R	I	D	E
Entity	V		V			
Process	V	V	V	V	V	V
Data Store		V	V	V	V	
Data Flow		V		V	V	

STRIDE-per-Element





1007 Threats !

Threat Modeling

Analyzing attack scenario

- Constructing attack scenarios related to 4 attack purpose by organizing threats/ vulnerabilities that identified in the previous step

Threats

T1. E1 / S / Attacker can obtain information through **spoofing**.

T2. ..

T3. ..

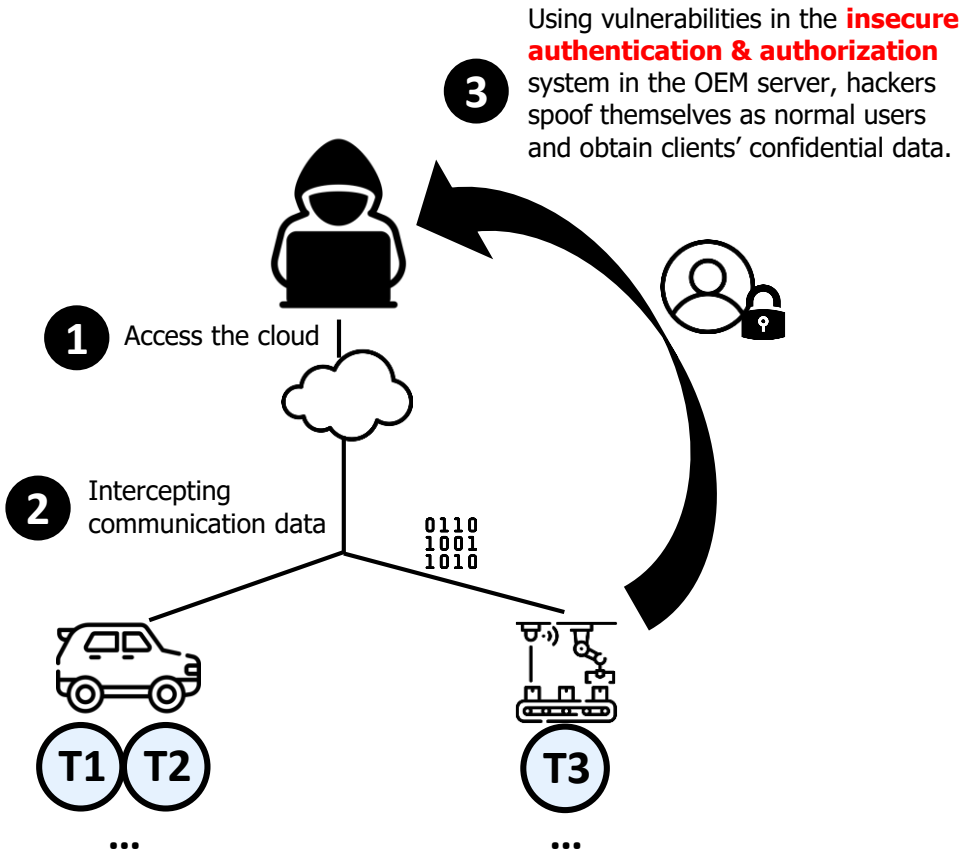
...

Attack Library

AL-P-1. Attacker can **acquire data by guessing** the admin's ID & PW.

AL-P-2. ...

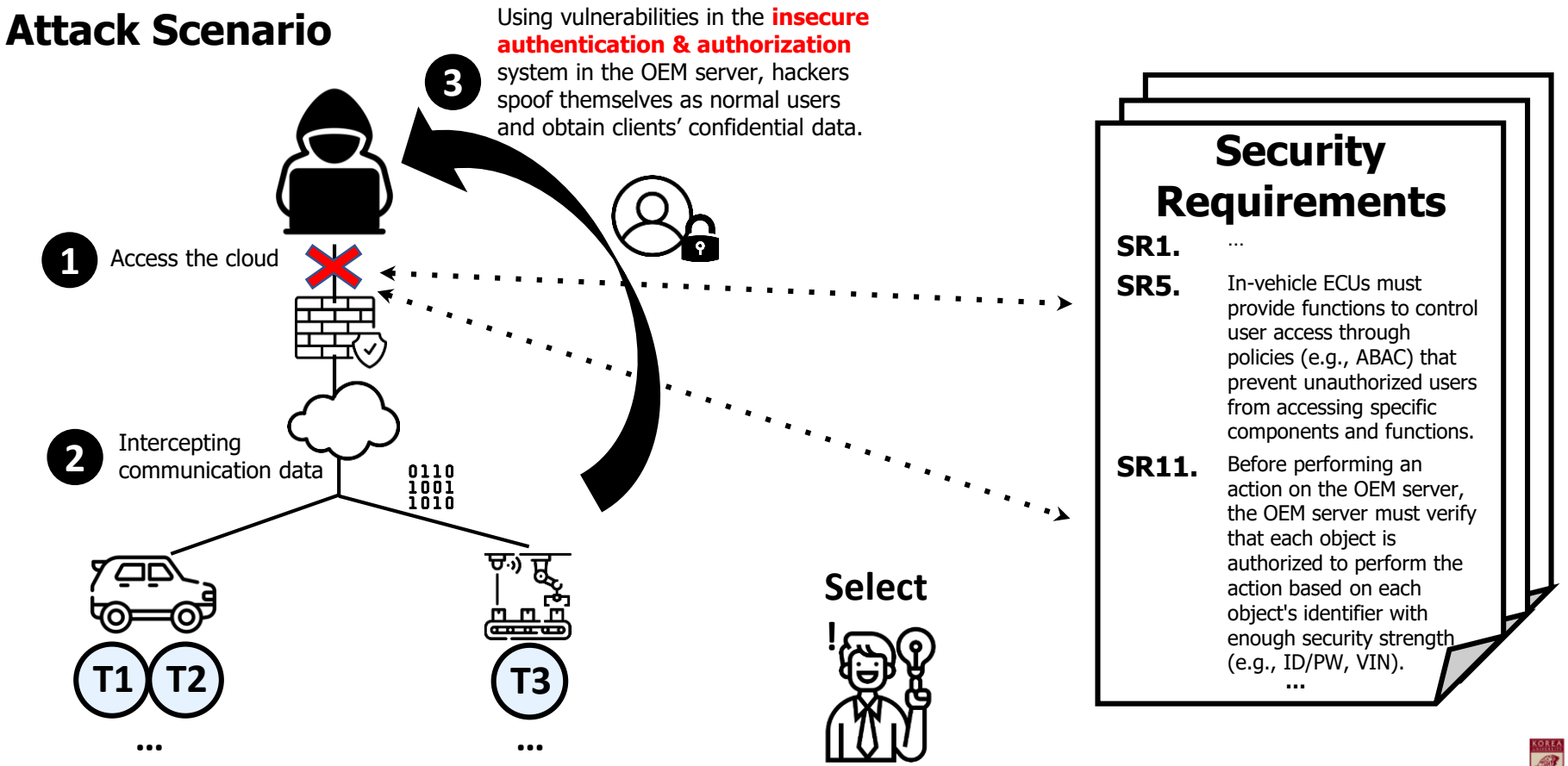
...



Threat Modeling

Deriving security requirements

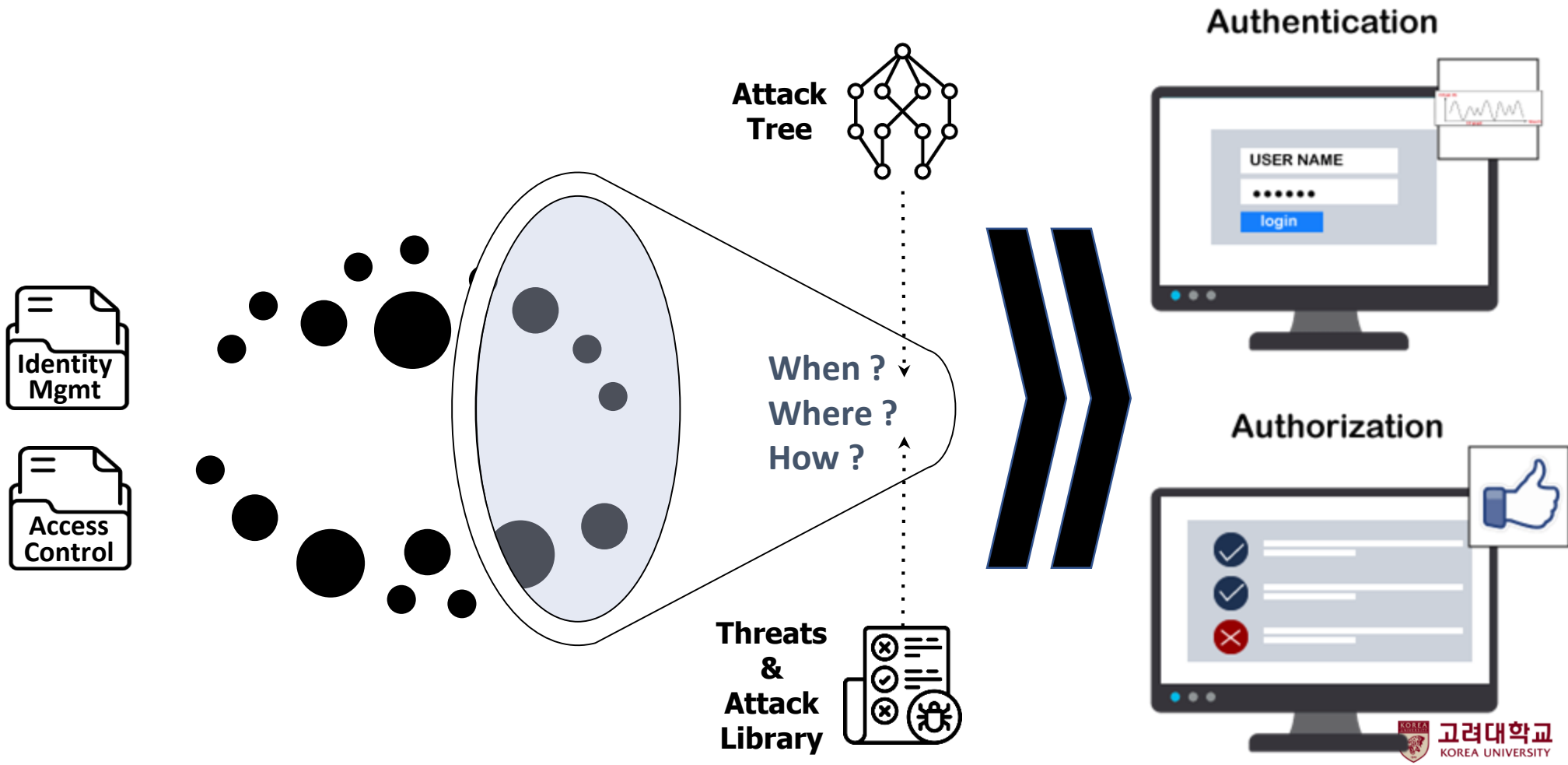
- To mitigate several attacks/threats in the scenario, we proposed 38 security requirements



Design

UML(Unified Modeling Language)

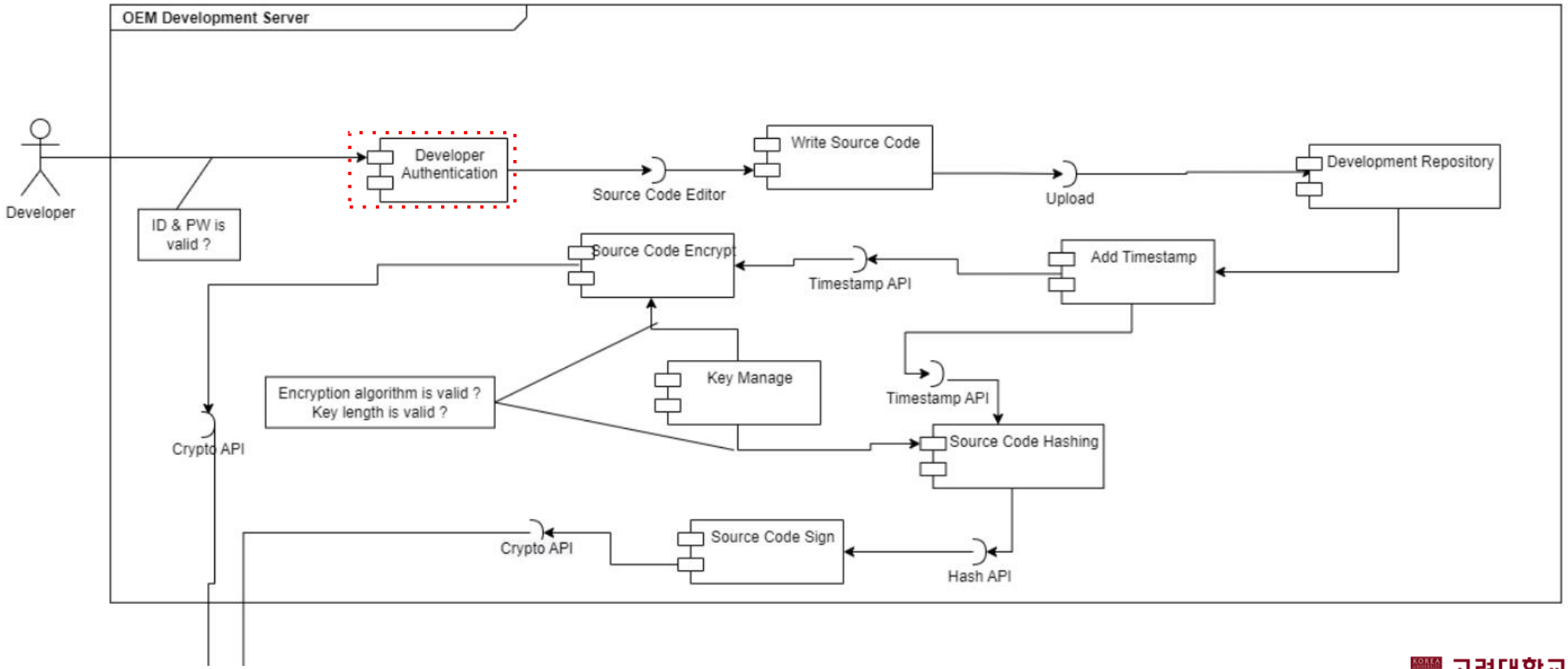
- Design security function based on multiple security function requirements in UML



Design

UML(Unified Modeling Language)

- Design security function based on multiple security function requirements in UML



Formal Method

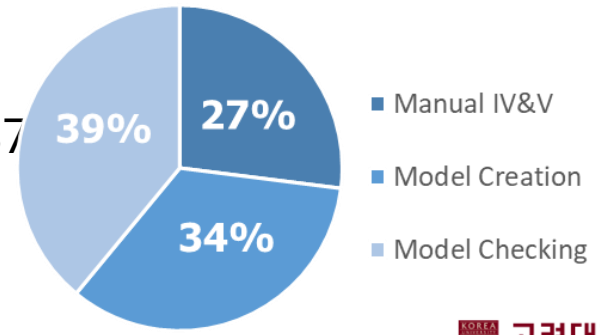
What is formal method ?

- Formal methods are mathematically based techniques for specification & verification of systems, both hardware and software.
- The use of formal methods approaches can help to eliminate errors early in the design
- process.

  Practitioners have also recognized that they can make searching for reusable components more effective by having formal specifications of components

Benefits of formal method (Case study in NASA)

- Using formal method found 144 defects their traditional IV&V would miss (73% of all defects found)
  - Estimating it would cost approximately 3500 men per hour at $100 per man hour to fix the 144 defects later in the lifecycle
  - Early defect removal savings is $350K
  - The cost to perform formal methods analysis: -$137
  - Net savings of $213K or 5% of the total project



[Origin of Defect Detection in NASA]

Formal Method

Event-B

- Formal method for system-level modeling and analysis which uses
    - set theory as a modeling notation
    - refinement to represent systems at different abstraction levels
    - models are organized into 2 components
        - contexts: (static) parameters of a formal model and their properties
        - machines: (dynamic) transition system with the state specified by variables and guarded events

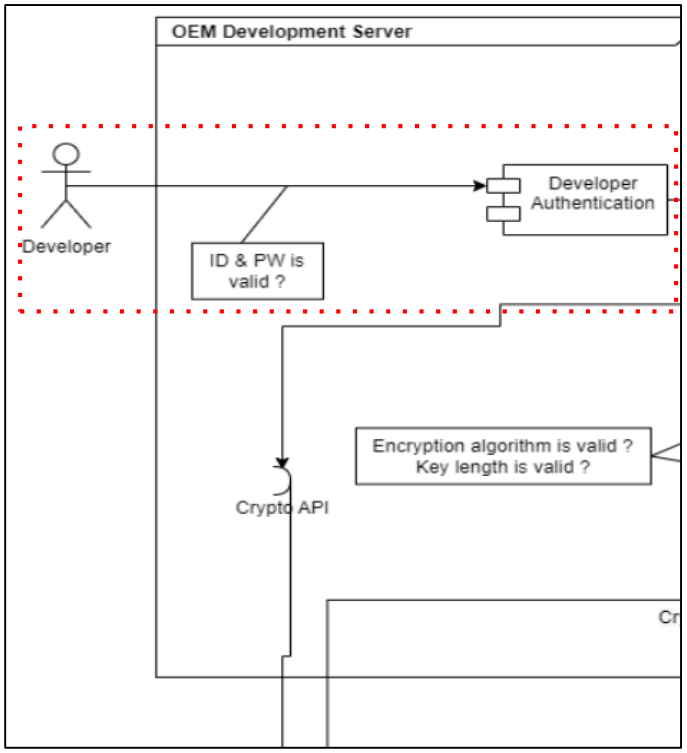
Rodin Platform

- Eclipse-based IDE for Event-B
    - provides support for refinement and mathematical proof
        - by Atelier B engine, its formal can be formally verified
    - refinement to represent systems at different abstraction levels
    - mathematical proof to verify consistency between refinement levels

Formal Method

Formal specification

- Convert security functions into the formal form using Event-b




```
EVENTS
  INITIALISATION:    not extended ordinary
  THEN
    act1:   Permission :∈ ∅
    act2:   Entity := ∅
    act3:   OEM_Teamleader_LogIn_Try := 0
    act4:   Access_Agent :∈ ∅
  END
  OEM_TeamLeader_Access_Granted:    not extended ordinary
  ANY
    Input_ID
    Input_Password
    Input_Agent
  WHERE
    grd1:   Input_ID = {OEM_TeamLeader_ID} ∧ Input_Password = {OEM_TeamLeader_Password} not theorem
    grd2:   OEM_Teamleader_LogIn_Try ≤ 5 not theorem
    grd3:   Input_Agent ∈ {Developer_PC} not theorem
  THEN
    act1:   Access_Agent := Input_Agent
    act2:   Permission := {Upload, Fix, Transfer, Download, Approve}
    act3:   OEM_Teamleader_LogIn_Try := 0
  END
  OEM_TeamLeader_Access_Denied:    not extended ordinary
  ANY
    Input_ID
    Input_Password
    Input_Agent
  WHERE
    grd1:   Input_ID ∉ {OEM_TeamLeader_ID} ∨ Input_Password ≠ {OEM_TeamLeader_Password} not theorem
    grd2:   OEM_Teamleader_LogIn_Try ≤ 5 not theorem
    grd3:   Input_Agent ∉ {Developer_PC} not theorem
  THEN
    act1:   OEM_Teamleader_LogIn_Try := OEM_Teamleader_LogIn_Try + 1
  END
```

① At first, its trial value is set to 0

② Check ID/Password & Login_Trial number

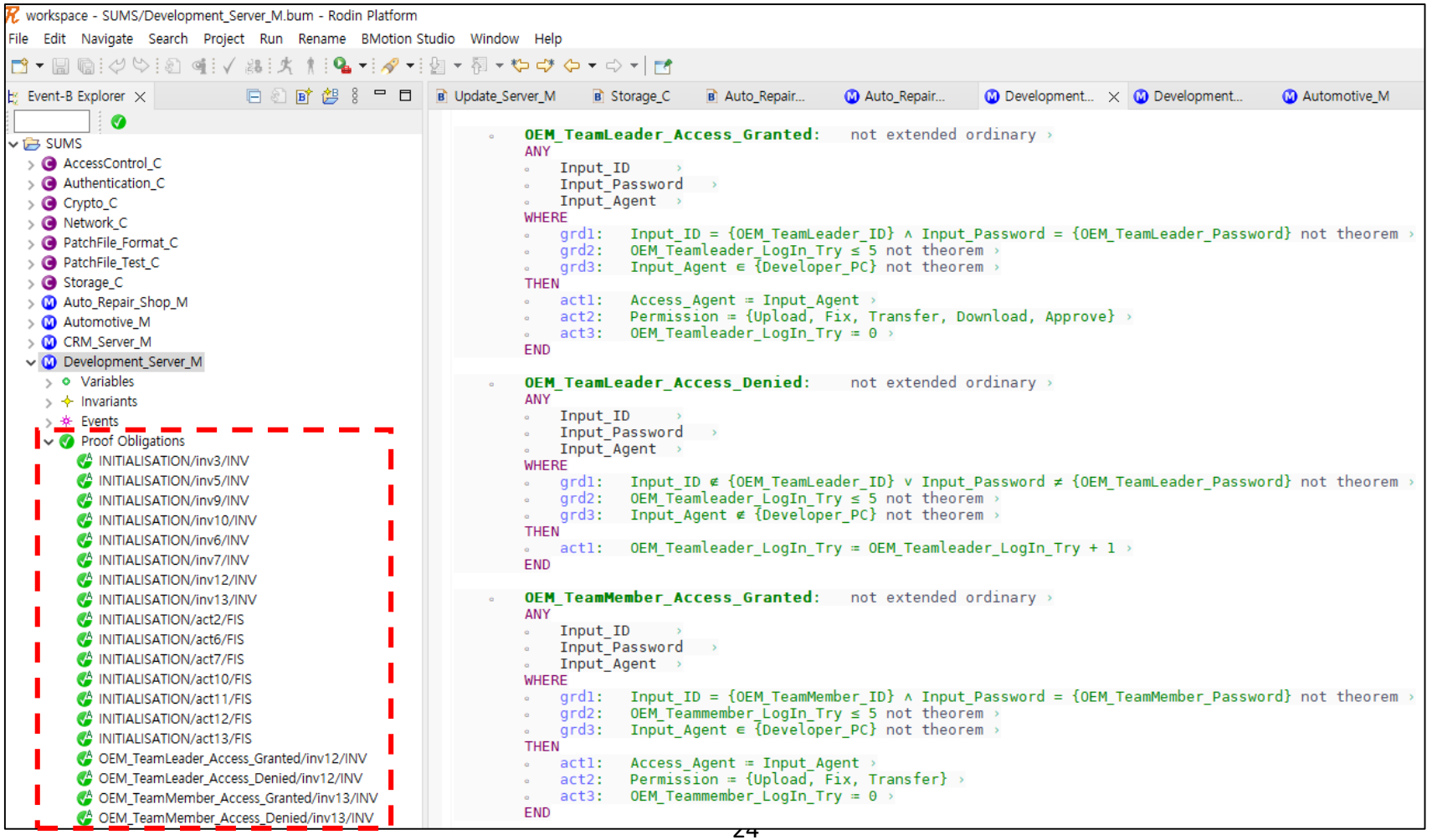
③ If Log-in succeed, its trial value is reset

③ If Log-in fail, its trial value is increase

Formal Method

Formal verification

- Verification of formal specifications using Atelier B which is loaded on Rodin



Conclusion

Conclusion

- Vehicles become vulnerable because extra software and hardware, which are necessary for providing the functionality, become new attack surfaces
- In this study, we conducted threat modeling for SUMS
  - Identifying all known threats & vulnerabilities that may occur in SUMS by STRIDE
  - Proposing security requirements and designed a secure SUMS architecture
- Finally, we formally verified that our secure SUMS architecture reflects the security requirements derived from threat modeling using Event-B

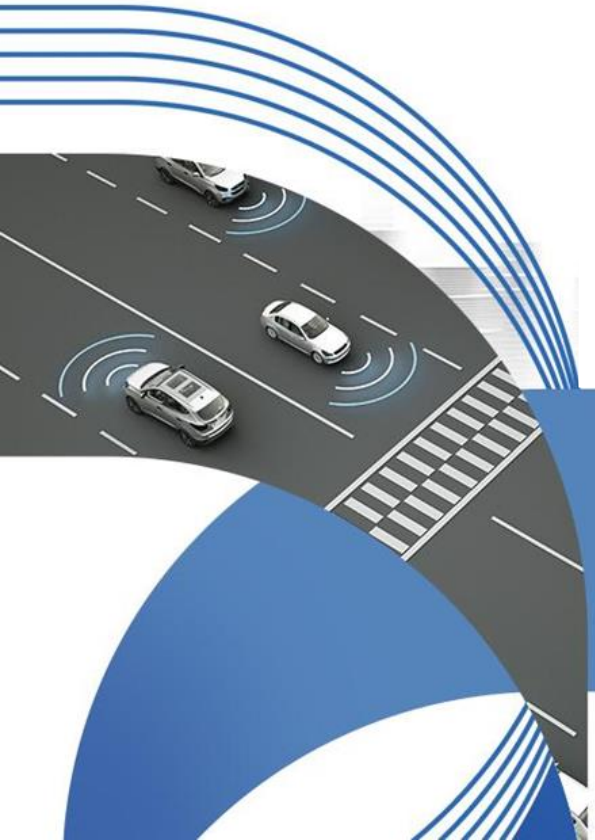
Future works

- Based on the formal model, we generate source code for SUMS by using Atelier B
  - In Atelier B, the additional implementational model is required to generate source
- code

  We are trying to generate source code by using deliverables only in the design phases

Any Questions ?

Thank you



고려대학교
KOREA UNIVERSITY