

Une école de l'IMT

SysML Model Transformation for Safety and Security

Florian Lugou, Rabéa Ameur-Boulifa, Ludovic APVRILLE ludovic.apvrille@telecom-paristech.fr



ISSA'2018 - Barcelona

Case study

Conclusion



Context: Security for Embedded Systems Embedded systems

SysML-Sec Method SysML-Sec

Case study Case Study

Conclusion

Conclusion, future work and references



SysML-Sec

Case study

Conclusion

Examples of Threats

Transport systems

- Use of exploits in Flight Management System (FMS) to control ADS-B/ACARS [Teso 2013]
- Remote control of a car through Wifi [Miller 2015] [Tecent 2017]

Medical appliances

 Infusion pump vulnerability, April 2015. http://www.scip.ch/en/?vuldb.75158



(C) Wired - ABC News



(C) Hospira



SvsML-Sec

Case study

Conclusion

How to Identify Vulnerabilities?

Investigations

- Testing ports (JTAG interface, UART, ...)
- Firmware analysis
- Memory dump
- Side-channel analysis (e.g. power consumption, electromagnetic waves)
- Fault injection

Institut Mines-Telecom

. . .

Sept. 2018

4/33

Secure your systems! Develop your system with security in mind from the very beginning Our solution: SysML-Sec, supported by TTool SysML-Sec

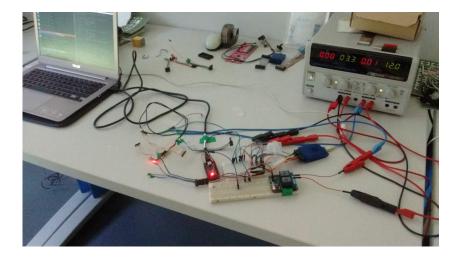


SysML-Sec

Case study

Conclusion

Firmware Dumping

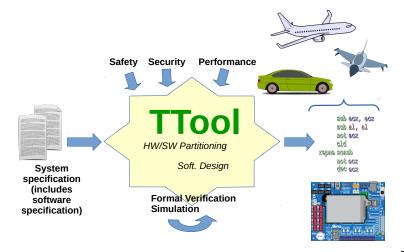




 Case study

Conclusion

Goal: Designing Safe and Secure Embedded Systems





SysML-Sec

Case study

Conclusion

TTool: Key Features



- Model-Driven Engineering tool
- Free and Open-Source
 - Plug-in can be used to insert private/commercial features
- Easy to use
- Focus on safety, security and performance
- Formal verification at the push of a button



SysML-Sec

Case study

Conclusion



Common issues (addressed by SysML-Sec):

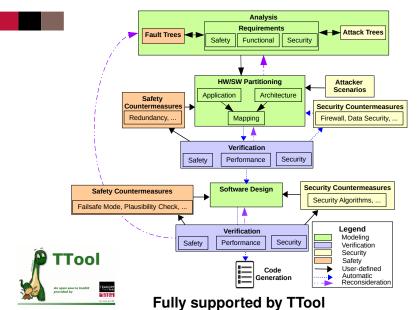
- Adverse effects of security over safety/real-time/performance properties
 - Commonly: only the design of security mechanisms
- Hardware/Software partitioning
 - Commonly: no support for this in tools/approaches in MDE and security approaches



SysML-Sec

Case study

Conclusion





SysML-Sec

Case study

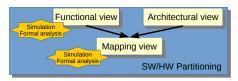
Conclusion





Before mapping

 Security mechanisms can be captured but not verified



After mapping

- Verify security (confidentiality, authenticity) according to attacker capabilities
 - Whether different HW elements are or not on the same die
 - Where are stored the cryptographic materials (keys)
 - Where are performed encrypt/decrypt operations
- Impact of security mechanisms on performance and safety
 - e.g. increased latency when inserting security mechanisms



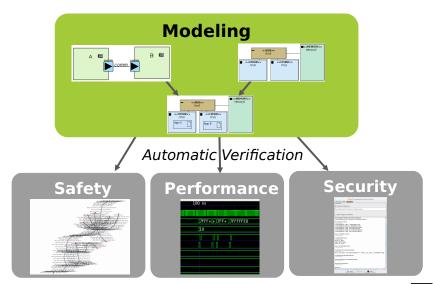


SysML-Sec

Case study

Conclusion

Partitioning Verification





SysML-Sec

Case study

Conclusion

Security Verification

SysML Model **ProVerif Specification** Results Bus1 Generated ProVerFspecFication*) (* Quesies Secret *) query attacked/set Alice_secretD ata) Courses and Distant (* Symmetric key csytography *) fus i wrcsyte (bit tring, bits ting): bit tring, reduc farail i: bitsting, k bitsting, i decsyst CPU1 CPU2 Translation int Alice_0 (resciend): bits sing) = is (ch Cannol, cfC annolDate, bits ting): let (m esciend), rcal_Alice_0, Alice_receD ata_2; kitelica that may see Task1 Task2 101.1.200 DC DC PS S If new cession D: bistring/if System_0(session10) Task1 Task2 Bob_0(sessionID))](Alice_0(sessionID) nen Alco_sk_dats: bits ting Laborations (Cardina) Task1 Verification Attacker Task1.kev sencrypt(Task1.secretData, Task1.key)

Backtracing



Case study

Conclusion

Automated Proverif Specification Generation

- Main idea
 - Decompose SysML-Sec behaviors into a set of basic blocks
 - Generate Proverif code
- The semantic function for generating the code:
 - Processes generation

 $\llbracket.\rrbracket_{\mathcal{E}}^{p}:\textit{Basic_block} \rightarrow \textit{Proverif_process}$

Main process generation

 $[\![.]\!]_{\mathcal{E}}: \textit{SysML_components} \rightarrow \textit{Proverif}$



Sept. 2018

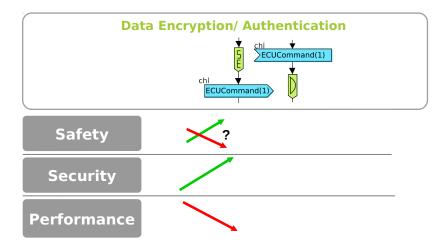
14/33

SysML-Sec

Case study

Conclusion

Safety and Security Mechanisms

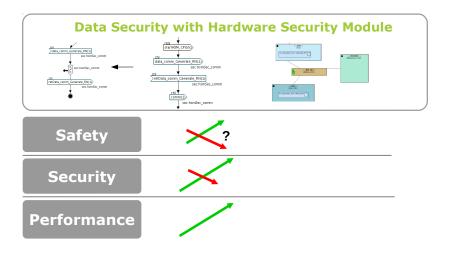




Case study

Conclusion

Safety and Security Mechanisms (Cont.)



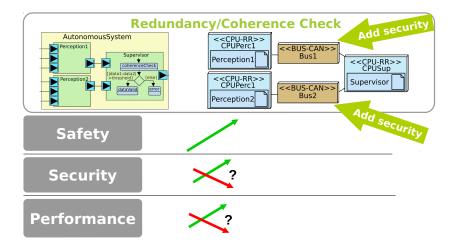
				TELECOM	- -	
15/33	Sept. 2018	Institut Mines-Telecom	SysML-Sec	ParisTech ■ 密調問	ė	

SysML-Sec

Case study

Conclusion

Safety and Security Mechanisms (Cont.)

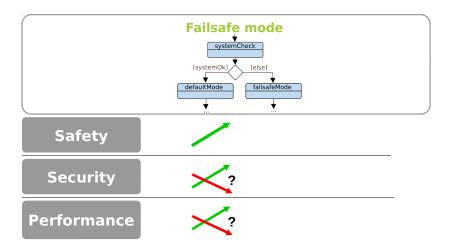


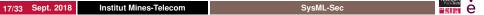


Case study

Conclusion

Safety and Security Mechanisms



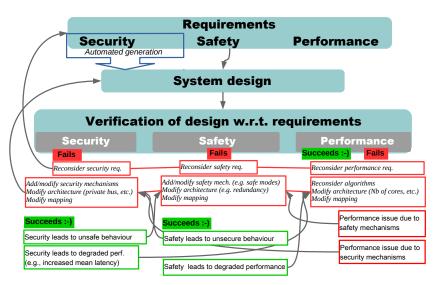


SysML-Sec

Case study

Conclusion

Safety/Security/Performance



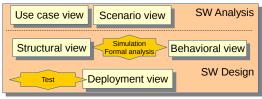


Case study

Conclusion



SysML-Sec: SW Design



- Precise model of security mechanisms (security protocols)
- Proof of security properties : confidentiality, authenticity
- Channels between software blocks can be defined as private or public
 - This should be defined according to the hardware support defined during the partitioning phase



Case study • 0000000000 Conclusion

Case Studies

Cyber security of connected vehicles

- Safety/Security/Performance
- EVITA FP7 Partners: Continental, BMW, Bosch, ...
- VEDECOM

H2020 AQUAS

- Automated train sub-systems (ClearSy): Safety/Security/Performance
- Industrial Drives (Siemens): Safety/Security/Performance

Nokia

Digital architectures for 5G networks (Safety/Performance)

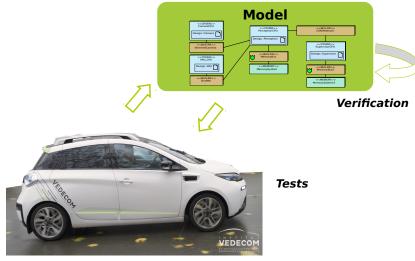


SysML-Sec

Case study

Conclusion

Case Study: VEDECOM Autonomous Vehicle





SysML-Sec

Case study

Conclusion



Standard: ISO26262

- SOTIF: Safety Of The Intended Function
- Security: impact of potential attacks on safety

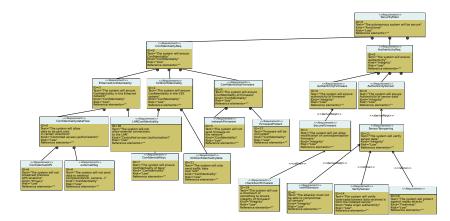


SysML-Sec

Case study

Conclusion

Requirements

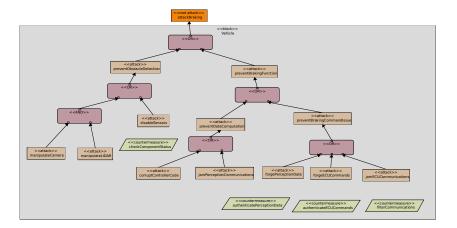


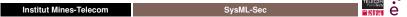


SysML-Sec

Case study Conclusion

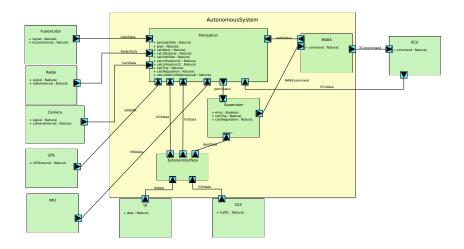






Case study

Functional View

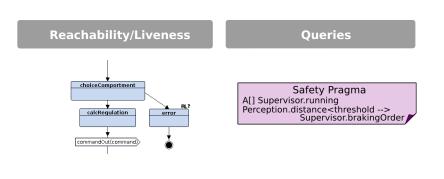




Case study

Conclusion

Safety Verification (Before Mapping)



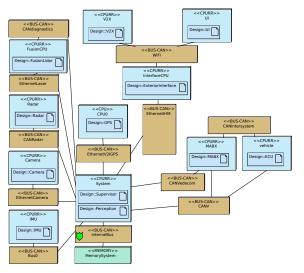


SysML-Sec

Case study

Conclusion

Architecture and Mapping Views



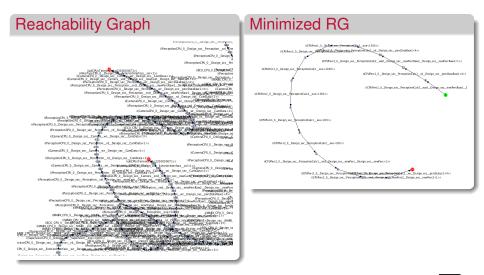


SysML-Sec

Case study

Conclusion

Safety Verification (After Mapping)





SysML-Sec

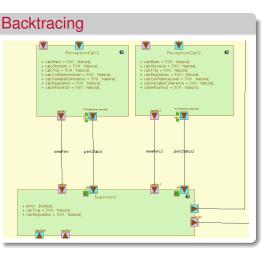
Case study

Conclusion

Security Verification

Dialog window

Verification options Generale ProVerifi ode III: [Users/hdovicapvfile/TTool/provent/] [optioprestrates reachability: # all		mated Security		
Execute ProVerif as: /copt/proverif/proverif Compute state reachability: Bill >selected no Allow message duplication in private channels: ¥ Yes No I Generate typed Pf calculus No	Company Bushington da Inc.	Discound of the second statement	- 181	
Compute state reachability: ● all ○ selected ○ no Allow message duplication in private channels: ● Yes ○ No ☑ Generate typed Pi calculus			eni/	
Allow message duplication in private channels: Yes No Generate typed Pi calculus				O no
Generate typed Pi calculus		private channels: Yes		
	V2X.percData2			
Satisfied Weak Authenticity:				
Satisfied Weak Authenticity: PerceptionCalc1_encrypt_percData1percData1 ==> Supervisor.decrypt_percData1_dummy PerceptionCalc2_encrypt_percData2.percData2 ==> Supervisor.decrypt_percData2_dummy	Satisfied Weak Authenticity: PerceptionCalc1encrypt_per			
PerceptionCalc1_encrypt_percData1.percData1 ==> Supervisor.decrypt_percData1_dummy; PerceptionCalc2_encrypt_percData2.percData2 ==> Supervisor.decrypt_percData2_dummy;	Satisfied Weak Authenticity: PerceptionCalc1encrypt_per PerceptionCalc2encrypt_per			
PerceptionCalc1_encrypt_percData1_percData1 ==> Supervisor.decrypt_percData1_dummy PerceptionCalc2_encrypt_percData2.percData2 ==> Supervisor.decrypt_percData2_dummy Non Satisfied Authenticity:	Satisfied Weak Authenticity: PerceptionCalc1encrypt_per PerceptionCalc2encrypt_per Non Satisfied Authenticity:	Data2.percData2 ==> Superviso	r.decrypt_percData2_d	ummy
PerceptionCalc1_encryt_percData1perCbata1 ==> Supervisor.decryt_percData1_dummy PerceptionCalc2_encryt_percData2percData2 ==> Supervisor.decryt_percData2_dummy Non Satisfied Authenticity: PerceptionCalc1.signalistate_writechannel_Design_sec_percStatus2percStatus2,chData ==> 1 PerceptionCalc2.signalistate_writechannel_Design_sec_percStatus2percStatus2,chData => 1	Satisfied Weak Authenticity: PerceptionCalc1_encrypt_per PerceptionCalc2_encrypt_per Non Satisfied Authenticity: PerceptionCalc1.signalstate_ PerceptionCalc2.signalstate_	Data2.percData2 ==> Superviso	r.decrypt_percData2_d	ummy: 1 ==> ! ata ==
PerceptionCalcL_encrypt_percDatal_percDatal_a = > Supervisordercrypt_percDatal_dummy PerceptionCalc2_encrypt_percDataZpercData2 ==> Supervisordercypt_percData2_dummy nos Satisfied Authenticity: PerceptionCalc2.signalstate_writechannel_Design_scepercStatus2percStatus2chData ==>	Satisfied Weak Authenticity: PerceptionCalc1_encrypt_per PerceptionCalc2_encrypt_per Non Satisfied Authenticity: PerceptionCalc1.signalstate_ PerceptionCalc2.signalstate_	Data2.percData2 ==> Superviso	r.decrypt_percData2_d	ummy 1 ==> : ata ==
PerceptionCalcl.encypt.percDatal.percDatal ==> Supervisor.decrypt.percDatal.dummy PerceptionCalc2_encypt.percData2percData2 ==> Supervisor.decrypt_percData2_dummy ion Satisfied Authenticity: PerceptionCalc2.signalistae_writechannel_Design_sec_percStatus2perClatus2.ndbata ==>>	Satisfied Weak Authenticity: PerceptionCalc1_encrypt_per PerceptionCalc2_encrypt_per Non Satisfied Authenticity: PerceptionCalc1.signalstate_v PerceptionCalc2.signalstate_v	Data2.percData2 ==> Superviso vritechannel_Design_sec_percSta vritechannel_Design_sec_percSta	r.decrypt_percData2_d itus.percStatus_chData itus2.percStatus2_chD	ummy: 1 ==> ! ata ==

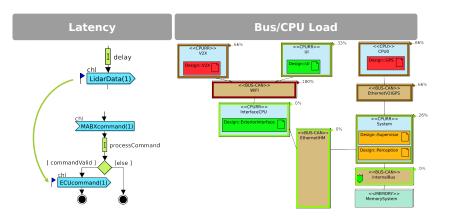




Case study

Conclusion

Performance Verification





Case study

Conclusion

SW Design, Code generation, Test

- First SW model from mapping models
- SW model refinement
- SW model verification (safety, security)
- Code generation
 - (Virtual) Prototyping, test





Case study

Conclusion ●○

Conclusion and Future Work

Achievements: SysML-Sec

- Methodology for designing safe and secure embedded systems
- Fully supported by TTool
- Applied to different domains, e.g., automotive systems, IoTs, malware

Future work

- Security risk assistance and backtracing
- Assistance to handle conflicts between security/safety/performance
 - Design space exploration



SysML-Sec

Case study

Conclusion ○●

To Go Further ...

Web sites

- https://sysml-sec.telecom-paristech.fr
- https://ttool.telecom-paristech.fr



References

- Ludovic Apvrille, Yves Roudier, "SysML-Sec: A SysML Environment for the Design and Development of Secure Embedded Systems", Proceedings of the INCOSE/APCOSEC 2013 Conference on system engineering, Yokohama, Japan, September 8-11, 2013.
- Ludovic Apvrille, Yves Roudier, "Designing Safe and Secure Embedded and Cyber-Physical Systems with SysML-Sec", Chapter in Model-Driven Engineering and Software Development, p293–308, Springer International Publishing, 2015

