Modeling the trade-off between security and performance to support the product life cycle

Radek Fujdiak*†, Petr Blazek*, Ludovic Apvrille‡, Zdenek Martinasek*, Petr Mlynek*†, Renaud Pacalet‡, David Smekal* Pavel Mrnustik†, Maros Barabas* and Maysam Zoor‡
*Brno University of Technology, Technicka St. 12 Brno 61600, Czech Republic
Email: {fujdiak, mlynek, maros.barabas, martinasek}@vutbr.cz,
{blazekpetr, smekal}@phd.feec.vutbr.cz
†Trustport, Purkynova St. 2845/101 Brno 612 00, Czech republic
Email: {Pavel.Mrnustik}@trustport.com
‡Telecom ParisTech, Campus SophiaTech, 450 route des Chappes st., BIOT Sophia-Antipolis 06410, France
Email: {ludovic.apvrille, renaud.pacalet, maysam.zoor}@telecom-paristech.fr

***Abstract*—Nowadays, the development of products for modern cyber-physical systems consists of many stages defined by the product life cycle (PLC). However, many manufacturers are not paying full attention - if any at all - to each PLC stage. This, among others, is causing growth of development costs. Therefore, the first stage of PLC becomes crucial. Moreover, a significant part of the development costs might be saved via testing the required parameters in this early stage, e.g., via modeling tools, simulation tools or emulators. Considering among others the current cyber-warfare and everyday growing number of threats, security is becoming one of the most critical topics in PLC. However, the security aspects come with significant trade-offs with performance. This paper focuses on methodology for dealing with these trade-offs via simulation in the early stage of PLC, where basic requirements are settled. To establish security requirements, an extensive Secure Software Development Life Cycle catalog is used together with an advanced modeling framework TTool based on UML/SysML-Sec for performance trade-off analysis. This combination creates a powerful approach for establishing the balance between security and performance requirements. As an example, a particular security requirement is selected. Namely, confidentiality, fulfilled by the encryption algorithm AES. This introduces the methodology and approach to the co-engineering issue in the PLC stages, where two different development teams with also different goals (security, performance) are dealing together with the single combined issue. Our results should help to understand the importance of the early PLC stage and show one possible approach on how to deal with these issues.**

I. Introduction

Nowadays, the product life cycle (PLC) and the product life cycle management (PLM) belong to the most crucial stages of manufacturing development processes. These stages help to manage the company's product all the way throughout its whole life-cycle [1]. The PLM considers several product life-cycle stages [2]: (i) conceive (specification, and concept design stage), (ii) design (detailed design, validation and analysis stage containing simulation, and tool design), (iii) realisation (plan for manufacturing, manufacturing, building/assembling, and testing together with quality control), and (iv) service (sell and deliver, use, maintain and support, and dispose). However, many manufacturers are still not paying sufficient attention to the PLC processes [3], although, optimization in this area may result in significant saving of energy, costs, workforce and much more [4].

The design of modern cyber-physical systems (CPSs) is currently facing many challenges including the implementation of safety, security and also performance requirements. Moreover, the safety, security and performance are mostly interdependent and the target is to find a reasonable balance or trade-off [5]. To do so, the companies are investing significant amounts of their resources into the design and realization stages. In the past decades, researchers and developers have created abstractions, which are helping in terms of product design [6]. These might help in terms of co-engineering and modeling during the design stage and provide a significant reduction of invested resources during this specific phase, as well as during the whole product life cycle [7].

During the development stage, many challenges might occur, which could throw the developers back to the beginning of their work. Modeling is a powerful method, which helps to avoid these mistakes and discover hidden relations or impacts on the monitored parameters. There are several projects which are dealing with these issues, such as enCOMPASS (H2020) [8], CRYSTAL (FP7) [9], ITEA2 MERgE [10], AMASS (H2020) [11], ASSERT (FP6) [12], INDENICA (FP7) [13], AQUAS (H2020) [14] and many others. The recent projects, AQUAS and AMASS, consider the interplay and trade-offs of all mentioned main parameters - security, safety and performance. This paper focuses on one selected part of the H2020 AQUAS project, namely the trade-offs between the security and performance parameters. The catalog tool Secure Software Development Life Cycle (SSDLC) was considered for defining the security parameters and security level based on the most current norms, standards, risk analysis, and best practice. Together with SSDLC, the simulation, modeling and verification framework TTool toolkit was used to help discover possible interconnections between the security requirements (security) and its impact on the final system response (performance) during the development stage, which should result in significant savings of resources in the later stages of the PLC.

II. Background

The continuous digitalization brings new trends such as the fourth industrial revolution, known as Industry 4.0, in which the old paradigm of isolated systems is no longer valid [7], [15], [16], [17]. It consists of a future vision of industrial development to the smart factory, including reliance on Cyber-Physical Systems (CPS) and construction of Cyber-Physical Production Systems (CPPS) [18]. These changes foster automation, productivity, reliability and completely change the business model of the current industry. However, these modern cyber-physical systems increasingly constitute a target for cyber-attacks [19] for example the 2014 steel mill attack in Germany [20] or in general attacks such as ICS Insider, IT Insider, Common Ransomware, Target Ransomware, Zero-Day Ransomware, Ukraine Attack, Sophisticated Ukraine Attack, Market Manipulation, and many others [21]. Therefore, ensuring the security and protecting the system are among the most crucial tasks for nowadays CPS [22]. The aspects of the cyber-physical systems that are ensuring security are very often negatively impacting the performance parameters such as system response, memory load, CPU load and more [5]. The negative correlation between performance and security creates many challenges over the whole PLC. Selection of right algorithms with considering the possible negative impact on each other is one of the essential steps before the development, which might help the companies to save significant resources in the following stages of the PLC [7]. The current approach of benchmark testing might be generalized and used for future projects as well.

This paper summarizes parts of the work that was completed in the first year of the AQUAS H2020 project [14] for industrial use-case led by Siemens company, and is dealing with co-engineering (development teams' cooperation in the different stages of PLC), tooling (tool interaction in the different stages of PLC), methodology (methodological work to generalize the cooperation processes through the PLC processes), and required trade-offs of safety, security and performance (everyday issues in the current PLC processes of modern CPS).

The main contribution of this paper is the introduction of the open tools, which might be used during the early stage of the PLC - TTool and SSDLC. Moreover, complex analyses of the mathematical background for the selected algorithm and security requirement are also provided. The novelty of this paper relies on a different approach towards the complexity and trade-off issue between security and performance. The implementation approach is always costly from the development point of view. Therefore, an example of a possible solution for an earlier stage of PLC is provided as well. The introduced methodology may be used across the security requirements and different use-cases because it consists of independent hardware and software implementation. Thus it is also portable. It brings a simple-solve solution for questions such as which security algorithm might be appropriate, which security level should be selected, which security recommendation might be still followed and others.

III. Experimental Environment

The experimental environment includes the two main tools SSDLC and TTool, which are closely described in the following sub-chapters A and B. The experimental environment is displayed in Fig. 1 (SL stands for security level and t represents the system response time).

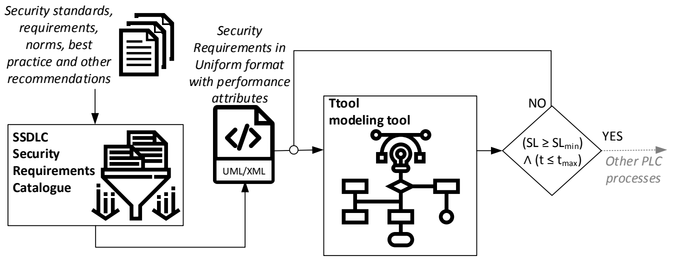
Security standards, requirements, norms, best practice and other recommendations
SSDLC Security Requirements Catalogue
Security Requirements in Uniform format with performance attributes
UML/XML
Ttool modeling tool
NO
(SL ≥ SL_min) ∧ (t ≤ t_max)
YES
Other PLC processes

Fig. 1. The experimental environment used in the modeling.

The SSDLC software processes the current trends, standards, security requirements, norms and best practice experiences and based on this information creates a well-arranged catalog of security requirements with a clear division into different security levels. These requirements are provided via UML/XML notation. The UML/XML is a generalized language, which provides common ground for exchanging data between different environments. In this case, UML/XML is used for connecting with other (modeling) tools, particularly TTool. The TTool analyzes the performance parameters (e.g., system response) based on the provided security requirements while responding to the selected system model.

The security level corresponds to various strength of the security, which was selected based on several different criteria mentioned before. The SL_{min} is the minimal level, which must be fulfilled based on local law or/and specific standard, best practice, use-case environment or/and others. On the other hand, t_{max} is the maximal acceptable response time, which might be appropriate in the selected use-case (this should serve as an example for selected performance and security parameters for a demonstrative purpose). These two parameters create in the end the difficult challenge for trade-off between security and performance:

$$(SL \geq SL_{min}) \wedge (t \leq t_{max}). \tag{1}$$

(1) is the condition, which selects the direction whether the PLC will continue to the next stage or it will be necessary to make significant changes and start with the process again. Also, if the SL will drop under the minimum value (SL_{min}), the model must be changed, e.g., we must consider changing the hardware setup for the model or reconsider our security approach and significantly change the security direction in the specific project. This also proves the necessity of this approach earlier in the PLC stages. The right selection might save valuable investments and may contribute to saving unnecessary costs of the development of a product, which would not fulfil the required parameters in the end.

A. Secure Software Development Life Cycle

The Secure Software Development Life Cycle (SSDLC) is a complex software developed during the AQUAS project to help in the initial stages of PLC. It aims to define a general set of security requirements concerning safety and performance. It also defines threat models for specific use-cases and creates a simple tool for implementing the security requirements in a PLC. Moreover, it provides a common vulnerability scoring system, scoring severities of potential threats and offers appropriate mitigation. The SSDLC approach includes several security aspects into development in PLC and provides:

- General security controls for effective risk management of information systems.
- Flexible catalog of security controls to meet the security threats, requirements and technologies.
- Involving activities to ensure security of the whole PLC (creating security requirements, verification of the analysis and design from the security point of view, developer guidelines, code reviews, verification of security requirements, security assessment).
- Treatment of various attack vectors based on generally accepted standards (NIST, OWASP, Microsoft SDL) .
- Measurement metrics for security control effectiveness. Secure SDLC approach includes several security aspects into development in PLC and provides particularly:
  – Integration of security activities into the application development process (creation of security requirements, design analysis, guidelines for developers, code review, penetration tests and others).
  – Depending on the business criticality of the developed application, the appropriate security engagement in the development is made to ensure that security activities do not disturb the development of the application unnecessarily.
  – Secure SDLC is based on generally accepted standards and best practice (NIST FIPS, STIG, Microsoft SDL, CIS Benchmarks) to treat various attack vectors with even quality.
  – Ability to integrate security activities into different developmental models (especially Waterfall, Agile development).

The SSDLC tool is an environment where different approaches from standards, recommendations, best practice, and many others (e.g., ISO/IEC 27034, ISO/IEC 11073-00103 or the frameworks from HITRUST, NIST or FDA/FTC) are brought together into a sorted list of requirements divided into security levels (e.g., based on strength of the algorithms). Connecting the SSDLC with other tools (e.g., TTool) improves the automation process of PLC. The SSDLC gives a connection and context between security, safety and performance parameters. Compared to the static security requirements definition, the SSDLC provides simple future extension and straight integration into the PLC process that requires none- or almost-none personal interaction.

B. TTool and SysML-Sec

TTool is a modeling and verification framework for embedded systems. TTool supports several modeling profiles, including SysML-Sec [23]. The main idea of SysML-Sec is to support safety, security and performance mechanisms all along its method (Fig. 2). The main features are:

1) Analysis is built upon requirements, fault trees and attack trees.
2) System-level HW/SW partitioning is made upon a functional view (functions and their logical communications), an architectural view (abstract HW components e.g. CPUs, buses, etc.), and finally a mapping view in which functions and communications are mapped to the architecture. A function mapped onto a processor becomes a software function. Conversely, a function mapped onto a HW accelerator becomes a HW function.
3) Software design is mainly built upon the design of software components and their deployment on more concrete hardware components.

Verification based on formal verification techniques or simulation can be performed throughout the SysML-Sec method using a press-button approach. Reachability of a given fault or attack can be checked to assess whether a countermeasure is efficient. Mapped software models can be checked against safety, security and/or performance properties. Safety verification relies on an internal model-checker. Security verification is performed by an external tool (ProVerif) according to a default attacker model (Dolev-Yao [24]) or to user-specific attack scenarios. Performance evaluation relies on abstract simulation (HW/SW partitioning) or on Cycle Accurate Bit Accurate simulations (deployment). Finally, executable code generation can be performed either from mapping or software models.

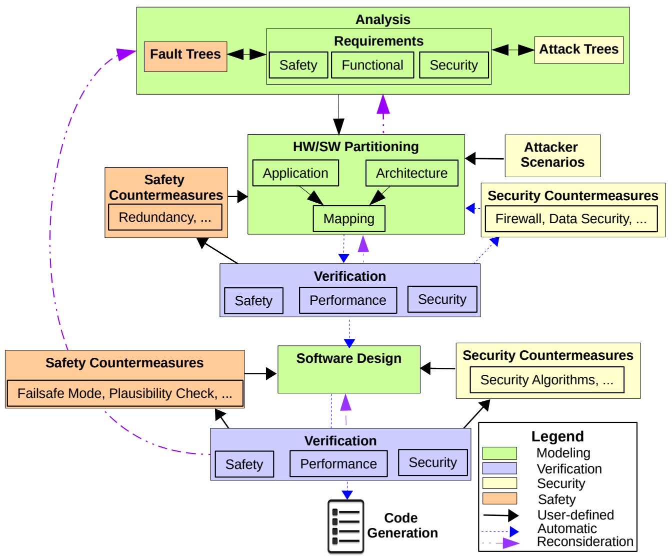

Fig. 2. Diagram of SysML-Sec method used for the modeling in TTool.

IV. Algorithmic Modelling

To determine the trade-offs between security and performance, we need to seek the complexity for different strength of selected security algorithms. As an example, the data confidentiality was selected as sufficient demonstrative case in the general industrial area. The SSDLC defines the AES algorithm as one of the most suitable algorithms for solving the data confidentiality for the given system in the industrial environment based (mainly) on standards IEC 62443 and NIST 800-53. Four main security levels are defined in this context, depending on the key length (and the corresponding number of AES rounds) see Table I.

TABLE I
Defined security levels from SSDLC for data confidentiality solved by AES cipher.

SL	0	1	2	3
Confidentiality	N/A	AES-128	AES-192	AES-256
Rounds (r_{aes-n})	N/A	10	12	14

To estimate the computational complexity, a standard software implementation on a low-end microprocessor has been considered, using well known optimizations (e.g. [25]). To simplify the explanation we assume that encryption and decryption have the same complexity but different complexities could very easily be modelled too. Other targets and implementations (high-end CPU, AES-specific instructions, custom hardware…) could as well be considered, with different figures. In the considered use case the AES is used in the Cipher Block Chaining (CBC) mode, with an initialization vector, as shown in Figure 3, for which a few extra operations (loads, looping, exclusive OR) must also be accounted for.

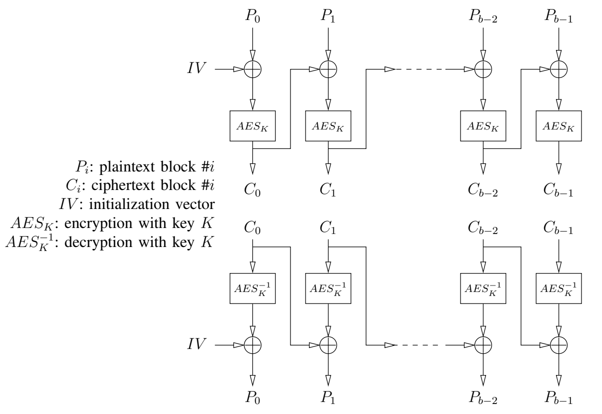
P_0
P_1
P_{b-2}
P_{b-1}
IV
AES_K
C_0
C_1
C_{b-2}
C_{b-1}
P_i: plaintext block #i
C_i: ciphertext block #i
IV: initialization vector
AES_K: encryption with key K
AES_K^{-1}: decryption with key K
AES_K^{-1}

Fig. 3. The AES cipher in CBC mode - encryption (top) and decryption (bottom).

Equation (2) models the number of instructions per 128-bits block encryption for different AES key lengths where i_0 is the initialization and CBC processing, i_m is the number of internal rounds, $i_{r_{aes-n}}$ is the last round and i_{aes-n} is the encryption of a 128-bits block using AES-n in CBC mode.

$$i_{aes-n} = i_0 + (r_{aes-n} - 1) \times i_m + i_{r_{aes-n}}. \quad (2)$$

The estimated number of instructions, for AES encryption, is displayed in Table II

TABLE II
The estimated number of instructions for different parts of AES algorithm and different key-sizes.

i_0	i_m	$i_{r_{aes-n}}$	$i_{aes-128}$	$i_{aes-192}$	$i_{aes-256}$
100	300	200	3000	3600	3900

Finally, equation (3) models the total number of instructions for a b-blocks message.

$$i_{aes-n-cbc-b} = b \times i_{aes-n}. \quad (3)$$

The model used in TTool is described in Figure 4. There are two communication sides (T1, T2), for which the encryption/decryption logic $E(K, M) = C$ and $D(K, C) = M$ (K for key, M for message, C for cipherdata, E for encryption, and D for decryption) and defined communication link between these two sides have been implemented.

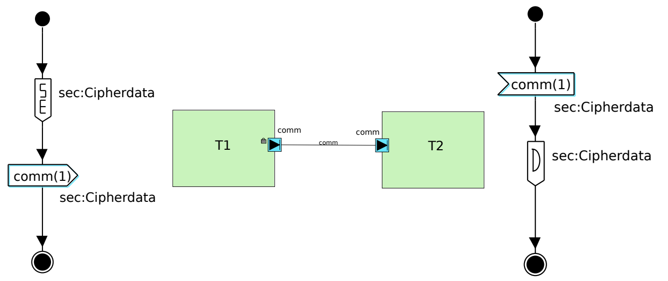

Fig. 4. The encryption and decryption model used in simulation environemtn TTool framework

V. Experimental Results

The experimental measurements were conducted with the following settings for CPU: 1 core of CPU, 8 B of data size, idle time 10 cycles, pipeline of 5 slots, task switch time 20 cycles, one instruction per clock cycle, and CPU clock rate was set to 2,9 GHz. For each testing, the input data was set on 1024 B with an ideal transmission channel. The final results from experimental measurements are displayed in Figure 5.

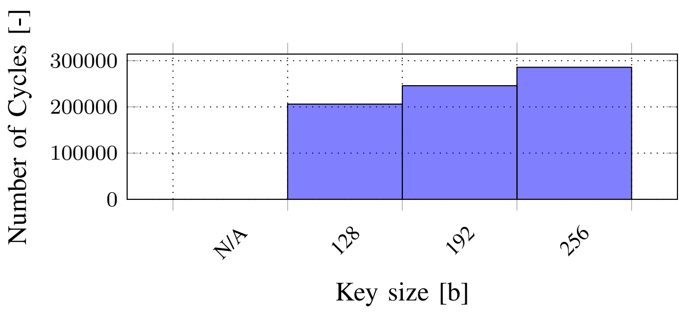

Fig. 5. Computational complexity for different key-sizes of AES cipher.

However, generalization and more relative numbers are needed for trade-off investigation. Tab. III shows comparison of two different known public benchmarks for cryptographic libraries CMTool [26] and Crypto++ [27].

TABLE III
ENCRYPTION SPEED BASED ON DIFFERENT SECURITY LEVELS (KEY-SIZES) IN CYCLES PER BYTE.

SL	0	1	2	3
Simulated	N/A	201.236	240.116	278.996
CMTool	N/A	17.783	20.283	23.217
Crypto++	N/A	12.600	15.400	18.200

As the table shows the accuracy of relative numbers is varied and it does not fully correspond to the real use-cases. This is caused by different approaches, implementations and/or optimization methods (memory, sbox, speedup, security and many others) [28], [29], [30], [31], [32], [33]. Therefore, we might look at the percentual difference of complexity in security levels. We define TTool simulation as the default value and use the delta approach to define the difference, see Tab. IV

TABLE IV
RECOMPUTED PERCENTUAL DIFFERENCES IN SPEED FOR DIFFERENT SECURITY LEVELS.

SL	0	1	2	3
Simulated	N/A	100%+0	+19%+0	+38%+0
CMTool	N/A	100%+0	+14%-5	+31%-7
Crypto++	N/A	100%+0	+22%+3	+44%+6

These results show that each security level slows down the speed by $19 \pm 5\%$ (difference between SL1 and SL3 $38 \pm 7\%$). However, the issue with different implementation still remains, which might be a hard challenge. We might look at the current public benchmark of lightweight BearSSL [34]. There are different implementations of AES: (i) a classic implementation with lookup tables (not constant-time) - "*big*", (ii) a compact implementation with small tables (non constant-time) - "*small*", (iii) a constant-time implementation with bitslicing over 32-bit registers with two blocks processed in parallel when the encryption mode allows it - "*ct*", and (iv) a constant-time implementation to "ct" with using 64-bit variable - "*ct64*". Three different types of CPUs have been measured: (i) an Intel Xeon CPU (E3-1220 V2) at 3.10 GHz (x64) - "*amd64*", (ii) an Intel Xeon CPU (E3-1220 V2) at 3.10 GHz (x86) - "*i386*", and (iii) an ARM Cortex M0+ at 48 MHz - "*m0+*". The recomputed results for percentual delta difference and speed in cycles per byte are displayed in Tab. V.

The results show that the $38 \pm 8\%$ is accurate for the difference between SL1 and SL3 and correspond to all deltas in the table above. This approach helps when there is an investigation of whether or not the solution might move to the next security level (e.g., because of new regulations or others) without any necessary additional implementations or tests. This, of course, saves many resources on the side of development and might help with the decision process, e.g., with addressing the minimum time needed for delivering critical messages or different performance questions.

TABLE V
RECOMPUTED PERCENTUAL DIFFERENCES IN SPEED FOR DIFFERENT SECURITY LEVELS IN BEARSSL FOR DIFFERENT IMPLEMENTATIONS AND HARDWARE.

CPU		big	small	ct	ct64
amd64	SL 1 [cyc./B]	18.257	77.384	56.353	33.557
	SL 3 [cyc./B]	23.957	105.802	77.365	43.928
	Δ [%]	**31.2**	**36.7**	**37.3**	**30.9**
i386	SL 1 [cyc./B]	24.369	95.946	72.396	83.468
	SL 3 [cyc./B]	31.084	130.088	98.163	111.712
	Δ [%]	**27.62**	**35.6**	**35.6**	**33.8**
m0+	SL 1 [cyc./B]	246.457	640.512	425.909	455.927
	SL 3 [cyc./B]	332.341	880.411	580.411	610.454
	Δ [%]	**34.7**	**37.5**	**36.3**	**33.9**

ACKNOWLEDGMENT

This article has received funding within the National Sustainability Program under grant LO1401. Our research and the idea of the paper is coming from the research conducted and supported by research project Aggregated Quality Assurance for Systems (AQUAS H2020-EU.2.1.1.7 ID: 737475). For the research, the infrastructure of the SIX Center was used.

VI. CONCLUSION

The paper discusses the importance of the product life cycle process and its stages. In this context, a particular trade-off issue between performance and security is discussed. Several approaches from different international projects were analysed, with particular focus on AQUAS project. The security-level approach derived from IEC 62443 and other international standardization such as NIST 800-53 or best practice was highlighted. The security requirements were processed via the SSDLC catalogue and particular confidentiality requirements were selected with solution via Advanced Encryption Standard (AES). These serve as an example for introduced appraoch in early-stages of PLC. The security levels were settled and computational complexity was computed. These results serve as an input to the TTool simulation environment. The final results of the complexity analysis and simulation showed that it is possible to show the percentual relation between each security level independently on hardware and implementation. This should serve in the decision processes in PLC phases before design or for example after regulation (law) change. It is obvious that it is necessary to think about the performance and security parameters already in the early stages of PLC as it might reduce significant number of issues caused by insufficient number proposal, which will lead to going back in PLC.

REFERENCES

[1] J. Stark, "Product lifecycle management," in *Product Lifecycle Management (Volume 1)*. Springer, 2015, pp. 1–29.
[2] L. Gould, "Additional abcs about plm," *Automotive Design and Production*, 2005.

[3] F. Tao, Y. Wang, Y. Zuo, H. Yang, and M. Zhang, "Internet of things in product life-cycle energy management," *Journal of Industrial Information Integration*, vol. 1, pp. 26–39, 2016.

[4] F. Shrouf and G. Miragliotta, "Energy management based on internet of things: practices and framework for adoption in production management," *Journal of Cleaner Production*, vol. 100, pp. 235–246, 2015.

[5] R. Fujdiak, P. Mlynek, P. Blazek, M. Barabas, and P. Mrnustik, "Seeking the relation between performance and security in modern systems: Metrics and measures," in *41st International Conference on Telcommunications and Signal Processing (TSP2018)*. IEEE, 2018, pp. 1–6.

[6] D. C. Schmidt, "Model-driven engineering," *COMPUTER-IEEE COMPUTER SOCIETY-*, vol. 39, no. 2, p. 25, 2006.

[7] T. Gruber, C. Schmittner, M. Matschnig, and B. Fisher, "Co-engineering-in-the-loop," in *DECSoS workshop at the SAFECOMP 2018: The 37th International Conference on Computer Safety, Reliability and Security*. IEEE, Springer, 2018, pp. 1–12.

[8] EU Publication Office, "Engineering compass: encompas h2020 project official website," https://cordis.europa.eu/project/rcn/205599_en.html, 2017, (H2020-EU.2.1.1, ID: 723833). Accessed: 2018-06-08.

[9] EU Publication Office, "Critical system engineering acceleration: Crystal fp7 project official website," https://cordis.europa.eu/project/rcn/111278_en.html, 2013, (FP7-JTI, ID: 332830). Accessed: 2018-06-08.

[10] EU Publication Office, "Multi-concerns interactions system engineering: Merge project official website," http://www.merge-project.eu/, 2017, accessed: 2018-06-08.

[11] EU Publication Office, "Architecture-driven, multi-concern and seamless assurance and certification of cyber-physical systems: Amass h2020 project official website," https://cordis.europa.eu/project/rcn/202642_en.html, 2017, (H2020-EU.2.1.1.7, ID: 692474). Accessed: 2018-06-08.

[12] EU Publication Office, "Automated proof based system and software engineering for real-time applications: Assert fp6 project official website," https://cordis.europa.eu/project/rcn/71564_en.html, 2009, (FP6-IST, ID: 004033). Accessed: 2018-06-08.

[13] EU Publication Office, "Engineering virtual domain-specific service platforms: Idenica fp7 project official website," https://cordis.europa.eu/project/rcn/95795_en.html, 2017, (FP7-ICT, ID: 257483). Accessed: 2018-06-08.

[14] EU Publication Office, "Aggregated quality assurance for systems: Aquas h2020 project official website," https://cordis.europa.eu/project/rcn/210527_en.html, 2017, (H2020-EU.2.1.1.7, ID: 737475). Accessed: 2018-06-08.

[15] I. Zolotova, R. Hosak, and M. Pavlik, "Supervisory control sustainability of technological processes after the network failure," *Elektronika ir Elektrotechnika*, no. 9 (125), pp. 3–7, 2012.

[16] J. Mocnej, W. K. Seah, A. Pekar, and I. Zolotova, "Decentralised iot architecture for efficient resources utilisation," *IFAC-PapersOnLine*, vol. 51, no. 6, pp. 168–173, 2018.

[17] J. Mocnej, T. Lojka, and I. Zolotová, "Using information entropy in smart sensors for decentralized data acquisition architecture," in *2016 IEEE 14th International Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, 2016, pp. 47–50.

[18] K. Zhou, T. Liu, and L. Zhou, "Industry 4.0: Towards future industrial opportunities and challenges," in *Fuzzy Systems and Knowledge Discovery (FSKD), 2015 12th International Conference on*. IEEE, 2015, pp. 2147–2152.

[19] P. T. Theron, "Proposals from the erncip thematic group, case studies for the cyber-security of industrial automation and control systems, for a european iacs components cyber-security compliance and certification scheme," 2014, (Report EUR 27098 EN).

[20] R. M. Lee, M. J. Assante, and T. Conway, "Ics cp/pe (cyber-to-physical or process effects) case study paper german steel mill cyber attack," 2014, (Industrial Control Systems, SANS ICS 2014).

[21] A. Ginter, "The top 20 cyber attacks against industrial control systems," 2017, (Waterfall Security Solution 2017 White-paper/Report).

[22] N. Jazdi, "Cyber physical systems in the context of industry 4.0," in *Automation, Quality and Testing, Robotics, 2014 IEEE International Conference on*. IEEE, 2014, pp. 1–4.

[23] L. Apvrille and Y. Roudier, "SysML-sec: A sysML environment for the design and development of secure embedded systems," in *APCOSEC 2013, Asia-Pacific Council on Systems Engineering, September 8-11, 2013, Yokohama, Japan*, Yokohama, JAPON, 09 2013. [Online]. Available: http://www.eurecom.fr/publication/4186

[24] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, March 1983.

[25] G. Bertoni, L. Breveglieri, P. Fragneto, M. Macchetti, and S. Marchesin, "Efficient software implementation of aes on 32-bit platforms," in *Cryptographic Hardware and Embedded Systems - CHES 2002*, B. S. Kaliski, ç. K. Koç, and C. Paar, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 159–171.

[26] M. Katarzyna, "Cmt: Crypto metrics tool," http://www.qopml.org/wp-content/uploads/CMTool/Benchmarks/benchmarks.html, 2014.

[27] D. Wei, "Crypto++ 5.6.0 benchmarks," https://www.cryptopp.com/benchmarks.html, 2009.

[28] D.-H. Bui, D. Puschini, S. Bacles-Min, E. Beigné, and X.-T. Tran, "Aes datapath optimization strategies for low-power low-energy multisecurity-level internet-of-things applications," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 12, pp. 3281–3290, 2017.

[29] S. Gueron and R. Shemy, "Two are better than one: Software optimizations for aes-gcm over short messages," in *Information Technology-New Generations*. Springer, 2018, pp. 187–191.

[30] A. Hafsa, N. Alimi, A. Sghaier, M. Zeghid, and M. Machhout, "A hardware-software co-designed aes-ecc cryptosystem," in *Advanced Systems and Electric Technologies (IC_ASET), 2017 International Conference on*. IEEE, 2017, pp. 50–54.

[31] J. Liu, C. Fan, X. Tian, and Q. Ding, "Optimization of aes and rsa algorithm and its mixed encryption system," in *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*. Springer, 2017, pp. 393–403.

[32] B. Peccerillo, S. Bartolini, and Ç. K. Koç, "Parallel bitsliced aes through phast: a single-source high-performance library for multi-cores and gpus," *Journal of Cryptographic Engineering*, pp. 1–13, 2017.

[33] L. Sarti, L. Baldanzi, B. Carnevale, and L. Fanucci, "An automated s-box optimization based on composite field arithmetic," in *Ph. D. Research in Microelectronics and Electronics (PRIME), 2017 13th Conference on*. IEEE, 2017, pp. 85–88.

[34] T. Pornin, "Bearssl: a smaller ssl/tls library," https://bearssl.org/speed.html, 2018.