Key users' needs addressed by SSPID innovation with respect to state-of-the-art

This section presents the state-of the-art in relation to the main technological areas covered in SSPID and in relation to the identified key users' needs. We will illustrate the potential and relevance of SSPID innovation to current and future approaches and strands of work on engineering secure IT-based systems. The final part of the document presents the main innovations of SSPID beyond state-of-the-art.

The engineering of secure IT systems has been the target of research works intended to deal with the increasing complexity of systems and characteristics such as distribution, real-time constraints and heterogeneity and with the need to provide increasing levels of security and privacy. Unfortunately, no integral and comprehensive approach has been able to successfully address those challenges and gain acceptance in practice. Regular software engineering methodologies are not extensively used in the industry as concluded in several studies [1,2]. If we focus on the development of secure systems, the situation is even worse and there is a growing awareness that the lack of adequate engineering processes and tools supporting the treatment of security aspects throughout the whole lifecycle is becoming a veritable Sword of Damocles hanging over many promising IT-based paradigms like Cloud computing, Internet of Things, Smart Cities, Cyber-Physical Systems, Smart Grids, etc.

Current practices for developing secure systems are still closer to art than to an engineering discipline. But the complexity of IT-based systems is scaling rapidly, and has already reached a point that exceeds the capacity of any security artisan, no matter how knowledgeable they may be. As a result, the security of current IT-based systems is far from acceptable, as it is obvious from the myriad of news in the media about attacks and vulnerabilities discovered and exploited every day.

Most researchers and practitioners in the field believe that this situation can only be improved by redefining the way we develop systems so that security becomes an integral part of the engineering activities [3] and by providing extensive computer-support for engineers throughout the whole system lifecycle [4]. Security is still treated as an after-thought and is, therefore, not fully integrated into software development practices and tools.

Together with researchers from NIST and MITRE, we performed an in-depth study [5] of the current approaches for security engineering and the potential for integration of security and system engineering and we concluded that the origin of this situation is in the lack of a comprehensive and practical approach featuring a tight integration of three essential components:

(i) A practical engineering process that covers the full development lifecycle of systems in any domain;
(ii) Means to capture and reuse security expertise; and
(iii) Computer automation and assistance mechanisms to support developers in identifying and fulfilling the security needs of their systems during the development stages, and in maintaining adequate security levels when systems are in operation.

In the following subsections we present, for each of the key users' needs identified, a summary of the state of the art, the weaknesses and limitations that we have identified, and SSPID approach to overcome those weaknesses and limitations.

Need of Security-aware Expertise-driven Engineering Process and Methodology

There is a growing awareness of the importance of enhancing system engineering practices in order to address the development of secure systems with reasonable guarantees. In fact, all around the world governments are actively promoting cybersecurity initiatives to react to the pressing need to make IT-based systems more secure [6-9]. In response to these mandates, several cybersecurity frameworks and initiatives have been proposed or are under development [10-11].

However, me must highlight that these cybersecurity frameworks and strategies remain at the organizational and procedural levels and therefore need to be complemented by analogous initiatives at the engineering level. Take for instance the cybersecurity framework developed by NIST [10], which is probably the most well-known cybersecurity framework today. The NIST framework describes the actions needed to develop a secure system at a very high level. It begins by describing well-established (ISO) standards for system and software engineering and infuses them with system security engineering techniques, methods and practices. Unfortunately, it only offers guidance on how to proceed and lack concrete means to capture security expertise, tool-support, etc. Likewise, the Software Engineering Institute of Carnegie Mellon has proposed a cyber-security engineering process [13] for providing security focused in the early stages of its development. As with the NIST, it does not provide support for selecting and integrating security solutions and has no tool to facilitate the adoption and to ensure higher levels of consistency and to provide assistance and validation. Of course, those initiatives are extremely important and valuable, but they focus on the organizational and awareness aspects of the problem. Therefore, complementary initiatives focusing on a more technical level are required in order for the former ones to achieve their full potential.

Moreover, these frameworks are heavily influenced by the traditional way security artisans have performed their work, and therefore consider threats and risks as the single entry point to the engineering activities. Again, this is not a problem at the higher level that these frameworks target, but it is a real problem in practice for technical engineering approaches. The main problems of the threats-centric approach are that threats change when context, technology, etc. changes even if the protection goals remain the same, and that it is difficult to identify the abstract protection goals from a threat-based specification. Therefore, threat- driven engineering approaches miss the “why” in the requirements specification and therefore provide poor support for complex and continuously evolving software systems. However, threat-based engineering has been successfully used to develop innumerable systems and is an extremely valuable element in the activity. Our position is that threats must be considered, but cannot be the single entry point for engineering activities. In our opinion there is a need for a change in current engineering approaches as almost all of them have a predefined entry point. Approaches with a “single entry point” are not realistic, or at least not practical. When developing a system, one cannot force developers to start by doing one specific activity or collecting a specific type of information (be it risks, threats or whatever else). What happens in real world is that you have a mix of information to start with (some risks, some properties, several threats, etc.) and therefore, successful and practical engineering approaches should allow developers to start working with any kind of information that is available to them. Moreover, we believe that there is a need to capture the relations between these elements (risks, threats, properties, solutions, etc.) in rich and rigorous knowledge-representation artefacts in order to enable multiple entry-point approaches.

In a category of its own, the Tropos [13] methodology is an agent-oriented software engineering process that models the requirements of the system as a composition of subsystems. Although promising, the approach remains at a very high level of abstraction where goals and interests of different stakeholders are the focus. Tropos has tool-support, but covers only the “initiation” activities when stakeholders have to come to agreements and establish trade-offs and high level strategies to achieve the goals. Connecting a Tropos model with other approaches covering

technical engineering and design of secure systems is still not well supported, although it remains an interesting problem to address.

Most of the existing security engineering processes are based on models, with UML being the most popular modelling framework [15]. UML has been used to model specific security aspects such as access control [16,17,18] and has been adopted as the preferred architectural representation of solutions in security patterns [19]. However, none of these approaches has focused on representing general security concepts in a comprehensive and integrated way and focusing on the engineering of secure systems as opposed to the representation of security solutions.

Some UML-based modelling languages such as SysML [20] have included additional support for security and privacy aspects. SysML is an interdisciplinary language that extends UML with various elements and processes such as requirements modelling, but have not really achieved a consistent and comprehensive framework for engineering secure systems, and moreover, lack integration with the regular system engineering practices.

Also based on UML, Model-Driven Security (MDS) [21] is an approach that intends to apply model-driven approaches to automatically generate technical security implementations from security requirements models. MDS uses SecureUML to model systems and design their security but, unfortunately, the only security aspects that can be described using this approach are role-based access control policies. Moreover, the approach tends to become complex and very dependent on the availability of security experts for critical activities like design decisions. For instance, Basin et al. [22] propose this model-driven methodology for developing secure data-management applications. To do so, system engineers have to model three different views of their desired application (data model, security model and GUI model). The process uses SecureUML in order to include security in the process but it requires system engineers to have a high level of security expertise in order to create the required solutions. Also the process does not take advantage of security expertise artefacts, which means that the solutions are not reused and must be manually integrated in the system.

Model-based security engineering [21] has been proposed as a general term to represent the major strand of research focusing on using models for security engineering. Opposed to Model-driven approaches, no automated transformations between models are considered here. However, most of these works still depend on complex and expert-provided inputs that require a large amount of security expertise, are weak in reusability of security knowledge and lack decision and validation support.

In fact, a major drawback of current approaches, even in the cases with well established methodology and a certain level of maturity, is that they fail to provide a reasonable support for systematic engineering since the identification, characterization and specification of the protection goals and the related threats as well as the selection of appropriate mechanisms and countermeasures depends on the experience of the engineers.

The approaches described above represent only minor improvements over the security craftsmanship era.

Key users' need identified. There is a pressing need for a systematic and comprehensive system engineering approach that: (i) Drives upon experts' security knowledge, but does not depend on their direct participation in the engineering and design processes; and (ii) Provides a high level of computer assisted decision support covering the complete IT systems engineering lifecycle.

SSPID approach to address identified users' need (beyond state-of-the-art). SSPID provides companies with an engineering process characterised with high-level of modularity, flexibility and adaptability that best scales to and integrates with the most common industry engineering practices. SSPID will allow companies to: (i) Achieve a security engineering process tailored to their systems' needs and practices, and (ii) Apply world-class security expertise and solutions into their systems

with high-level of rigor and precision.

Need of Machine-processable and Engineering-oriented Security Expertise Representation

Regarding the representation of security expertise, different artefacts have been proposed for representing security solutions, focusing on different engineering phases and following different paradigms. A comprehensive survey on the artefacts defined for representing security aspects [23] highlights the fact that most current artefacts are small extensions and improvements of artefacts that were developed in a time in which security threats and needs were very different and much less crucial. Other studies highlight the lack of mechanisms to enable the reuse of security knowledge with adequate levels of rigor and precision [24].

As a representation of security solutions, security patterns [25] are a mature paradigm intended to reuse the security expertise needed to design secure systems. Security patterns have demonstrated that their use, especially at the software architecture level, can be helpful for system designers as they provide documented, validated and tested solutions. There exist several security patterns libraries for this task but (i) there are no mechanisms to support trust and to evaluate the quality of the patterns; and (ii) in their current form, security patterns offer little to no support for rigorous and systematic pattern selection and practically no support for the integration into a system being developed.

Security pattern diagrams [26] have been presented as a mechanism for facilitating the selection of patterns. However, arranging patterns in diagrams that relate them is an over-simplification that does not provide rich selection capabilities (after all a diagram can only organize elements based on a very reduced set of attributes). In order to cope with the complexity of the design decisions in current and future systems we need much more powerful and rigorous decision support mechanisms.

Similarly to pattern diagrams, risk modelling [27] and threat modelling [28] have frequently used tree-based representations that (i) assume that navigation on the diagram will be based on one or two attributes, which is an oversimplification; and (ii) are isolated from other security aspects. Therefore they do not relate risks with properties, or solutions and threats, which is highly relevant and useful information for the system engineers. Moreover, it is frequent that these trees and diagrams are presented as “one-size-fits-all” ignoring the subtle but crucial differences between different application scenarios, technologies and context.

UML has also been used as the language to represent some aspects of security knowledge [16,17,18]. Although UML-based artefacts are usually tool-supported, most approaches have been limited to specific aspects such as access control and authorization.

Key users’ need identified. There is a pressing need to define novel knowledge representation artefacts that are able to capture security knowledge from experts with the goal of using those artefacts for assisting system engineers in the task of engineering secure systems. We believe that security patterns are one promising candidate for representing security solutions, but we reckon that in their current form they do not provide any support for computer processing. Moreover, other artefacts are necessary for representing other types of security knowledge and to integrate the different concepts in a consistent modelling formalism.

SSPID approach to address identified users’ need (beyond state-of-the-art). SSPID innovation will offer companies of any size an affordable and easy access to up-to-date word-class security expertise and solutions in a computer-processable format allowing them to streamline security and

privacy of their systems by design. SSPID users will have access to an extensive knowledge base for a wide range of domains such as cloud- and service-oriented computing, cyber-physical systems, Internet of things, e-government, etc.

Need of Security Engineering Toolset with High-degree of Decision Support and Process Automation

One of the most important activities when designing a secure system is deciding how to fulfil the targeted security requirements with sound and compatible security solutions. This process is usually manual and currently requires a high level of security expertise from the system engineer. In this respect, engineering process automation will provide crucial support to system engineers not only in the selection of the most suitable security solutions to addresses their requirements, but also in identifying security requirements, analysing those, integrating the selected solutions in the system model, and maintaining the consistency of the integrated security solutions.

A model-based method developed by Beckers et al. [29] defines functional safety requirements using a given set of safety goals and enables a rigorous validation of several constraints expressed in OCL. Also Riaz et al. [30] have developed a tool-assisted process that takes as input a set of natural language artefacts and, using machine learning techniques, automatically identifies security-relevant sentences in the artefacts and classifies them according to the security objectives. Moradian et al. [31] propose a process that supports verification of fulfilment of security goals and validation of security requirements during different phases of the development lifecycle. This is achieved by means of meta-agents, which automatically create a security checklist and a list of actions to be taken by system engineers.

A supporting tool for automation of security management is the Cyber Security Evaluation Tool [32]. It guides users through a step-by-step process to assess their control system and information technology network security practices against recognized industry standards. The output of the tool is a prioritized list of recommendations for improving the cybersecurity characteristics of the system. The tool derives the recommendations from a database of cybersecurity standards, guidelines, and practices.

Based on the popularity of UML, some works have proposed methodologies for verifying UML-based security requirements on system models [33,34]. For example, a recent work by Aoki and Matsuura [34] proposes a method to verify UML-based security requirements using model checking technique and Common Criteria security knowledge. In the proposed work, Common Criteria supports engineers in defining adequate security requirements derived from human-readable format documentation. This helps developers verify whether UML-based requirements analysis meet those requirements in the early stages of software development. Unfortunately, the method requires an extensive expertise from system engineers in security specification and does not provide any decision-support tools to help them define the security requirements.

Georg et al. [33] proposed an aspect-oriented risk-driven development methodology for developing secure systems. This methodology supports designers in defining system assets, identifying potential attacks against them, and evaluating system risks. When a risk is unacceptable designers mitigate the associated threat by incorporating security mechanisms into the system design. The methodology lacks a library of security knowledge that can be used in the design of the system and an engineering automation process that simplifies the decision-support of the security mechanisms.

Based on our analysis on the state-of-the-art automation methods and tools for engineering processes [5], we conclude that although some of them provide ways to support recommendations as well as to analysing the fulfilment of system security requirements, they provide no means to select and integrate security expertise, and manage adoption of trusted, certified and recognised security solutions with strong traceability and automation capabilities.

In summary, tool-supported security engineering processes are currently manual and requires a high level of security expertise from system engineers.

Key users' need identified. There is a pressing need for an engineering tool that provides support to system engineers in decision-making, minimising error-prone activities and avoiding that security engineering interferes with functional system engineering activities.

SSPID approach to address identified users' need (beyond state-of-the-art). SSPID innovation will provide crucial computer-assisted decision support to system engineers in the selection of the most suitable security solutions to address their requirements, and in the integration of selected solutions in their system model. SSPID toolset is designed to hide complexity and minimise error-prone activities, while at the same time documenting and tracing with appropriate evidences the sound use of security solutions.

Summary of main innovations of SSPID

Table 1 shows a summary of the main innovations of SSPID beyond the state-of-the-art.

Field	Advances
Security Knowledge Representation	Novel security patterns specification and expertise-representation artefacts enabling: • Detailed and precise security expertise and knowledge representation for computer-oriented processing and analysis; • Computer-oriented engineering of secure systems with expert-knowledge-driven decision support for systems engineering; • Evolution-friendly expertise representation and reasoning; • Trust-verifiable computing of expertise trustworthiness during system engineering.
Systems Engineering	Novel systems engineering process model allowing high level of: • Modularity, flexibility and adaptability to cope and scale to the diverse nature of current and future IT-based systems; • Reuse of knowledge provided by security experts; • Expertise-driven automation and computer assisted support to engineers; • Flexibility of adoption by customers to best suit their current engineering practices.
Engineering process automation	Novel expertise-driven decision-support methods, techniques and mechanisms to: • Support system engineers in decision-making and minimise error-prone activities; • Avoid secure system engineering interfere with functional system engineering by providing computer-supported on-the-fly verification of security in system models; • Support traceability of engineering processes, and ensure adequate documentation of security design decisions to facilitate evaluation, certification and labelling; • Support system evolution based on up-to-date security knowledge, and system resilience based on collective intelligence approaches.

Table 1: Main innovations of the SSPID solution beyond the state-of-the-art.

Conclusions

We have presented several problems and limitations in different areas of the security engineering that, in our opinion, are behind the limited adoption in industry. Bearing in mind the strong consequences unsecure systems may have, it becomes clear that a computer-oriented engineering approach capable of providing supporting-tools, trusted security solutions and engineering automation is of high need to industry. This is precisely the central objective of SSPID innovation.

We are well aware that a scientifically- and technically-sound solution is not enough. Without ensuring usability and adequate measures to foster adoption and overcome market barriers, such solution would never be used in practice despite of its theoretical merit [35,36].

Bibliography

[1] Klatt, K. and Marquardt, W. (2009), *Perspectives for process systems engineering—Personal views from academia and industry*, in Journal of Computers & Chemical Engineering, Elsevier, Vol. 33, Issue 3, pages 536-550, 2009

[2] John Hutchinson, Jon Whittle, Mark Rouncefield (2014). *Model-driven engineering practices in industry: Social, organizational and managerial factors that lead to success or failure*. Science of Computer Programming, Vol. 89, Part B, pp. 144–161, Elsevier

[3] Pretz, K. *IEEE Brings Together Top Security Specialists to Outsmart Hackers*. In the March 2015 special report on cybersecurity. The Institute. IEEE.

[4] Kissel, R., Stine, K., Scholl, M., Rossman, H., Fahlsing, J., & Gulick, J. (2008). *Security considerations in the system development lifecycle*, NIST Special Publication 800-64 Rev. 2

[5] Maña, A., Ross, R., McEvilley, M., Rudolph, C. and Ruiz, J.F. (2013) Integrating Security Engineering and System Engineering. In Annual Computer Security Applications Conference (ACSAC), New Orleans, USA. http://bit.ly/1Gvd1oz

[6] B. Obama. *International Strategy For Cyberspace. Prosperity, Security, and Openness in a Networked World*. May 2011.

[7] The White House. Executive Order 13636: *Improving Critical Infrastructure Cybersecurity*. Federal Register, Vol. 78, No. 33, February, 2013. Available online at https://www.federalregister.gov/executive-order/13636

[8] B. Obama. Executive Order -- *Promoting Private Sector Cybersecurity Information Sharing*. February 13, 2015

[9] European Commission. *Cybersecurity Strategy of the European Union: An Open, Safe and Secure Cyberspace* - JOIN(2013) 1 final – February 2013

[10] Ross, R.; Carrier Oren, J.; McEvilley, M. (2014), *Systems Security Engineering: An Integrated Approach to Building Trustworthy Resilient Systems,* NIST Special Publication 800-160.

[11] European Commission. *Proposal for a Directive of the European Parliament and of the Council concerning measures to ensure a high common level of network and information security across the Union* - COM(2013) 48 final – February 2013

[12] ENISA. Nation*al Cyber Security Strategies in the World.* February 2013. Available online at http://www.enisa.europa.eu/activities/Resilience-and-CIIP/national-cyber-security-strategies-ncsss/national-cyber-security-strategies-in-the-world

[13] Software Engineering Institute, Carnegie Mellon, USA. http://www.cert.org/sse/ (accessed March 09, 2015).

[14] Tropos Software Methodology - http://www.troposproject.org/ (accessed March 09, 2015)

[15] Unified Modelling Language. http://www.uml.org/ (accessed March 09, 2015)

[16] Jürjens, J. (2002) *UMLsec: Extending UML for secure systems development.* International Conference on The Unified Modeling Language (UML 2002).

[17] Basin, D., Doser, J. and Lodderstedt, T. (2006) *Model Driven Security: From UML Models to Access Control Infrastructures*. Journal of ACM Transactions on Software Engineering and Methodology (TOSEM). Volume 15, Pages 39-91. ACM, New York, USA

[18] Lodderstedt, T., Basin, D. and Doser, J. (2002) *SecureUML: A UML-Based Modeling Language for Model-Driven Security*. Proceedings of UML 2002 - Model Engineering, Concepts and Tools 5th International Conference. Dresden, Germany. Pages 426-441. Springer Berlin.

[19] Eduardo Fernandez-Buglioni (2013). *Security Patterns in Practice: Designing Secure Architectures Using Software Patterns*. Wiley, ISBN: 978-1-119-99894-5. 11

[20] Systems Modelling Language. *SysML Open Source Specification Project*. http://www.sysml.org/ (accessed March 09, 2015).

[21] Jurjens, J. (2010) *Model-Based Security Engineering with UML: The Last Decade and Towards the Future (Keynote)*. IEEE Symposium on Visual Languages and Human-Centric Computing. September 2010. DOI: 10.1109/VLHCC.2010.10. IEEE

[22] Basin, D. et al. (2014) *A Model-Driven Methodology for Developing Secure Data-Management Applications*. IEEE Transactions on Software Engineering, Vol. 40, Issue 4, pp. 324-337.

[23] M. Felderer, B. Katt, P. Kalb, J. Jürjens, M. Ochoa, et al. (2014). *Evolution of Security Engineering Artifacts: A State of the Art Survey*. International Journal of Secure Software Engineering, Vol. 5, Issue 4.

[24] Sharma, N.; Singh, K.; Goyal, D.P. (2012). *Adoption of Knowledge Management Practices in Software Engineering Organizations: A Survey of Software Engineers' Perceptions*. Second Intl. Conference on Advanced Computings & Communication Technologies, pp. 24-29, IEEE.

[25] Schumacher, M., Fernandez, E., Hybertson, D. and Buschmann F. (2005) *Security Patterns: Integrating Security and Systems Engineering*. John Wiley & Sons. ISBN: 0470858842

[26] E. B. Fernandez, G. Pernul, M. M. Larrondo-Petrie. *Patterns and Pattern Diagrams for Access Control.* In Trust, Privacy and Security in Digital Business. Lecture Notes in Computer Science Volume 5185, 2008, pp 38-47.

[27] Lund, M. S.; Solhaug, B.; Stølen, K. *Model-Driven Risk Analysis. The CORAS Approach*. Springer. 2011.

[28] Morikawa, I.; Yamaoka, Y. *Threat Tree Templates to Ease Difficulties in Threat Modeling.* 14th International Conference on Network-Based Information Systems (NBiS), 2011.

[29] Beckers K. et al. (2014) *Systematic Derivation of Functional Safety Requirements for Automotive Systems*. 33rd International Conference on Computer Safety, Reliability & Security (SAFECOMP). Florence, Italy. Pages 65-80. Springer Switzerland.

[30] Riaz, M., King, J., Slankas, J. and Williams, L. (2014) *Hidden in Plain Sight: Automatically Identifying Security Requirements from Natural Language Artefacts*. IEEE 22nd International Requirements Engineering Conference. Karlskrona, Sweden. Pages 183-192. IEEE

[31] Moradian, E. and Hakansson, A. (2010) *Controlling Security of Software Development with Multi- agent System*. In 14th International Conference in Knowledge-Based and Intelligent Engineering & Information Systems. Cardiff, UK. Pages 98-107. Springer Berlin.

[32] ICS-CERT: *Industrial Control Systems Cyber Emergency Response Team. Cyber Security Evaluation Tool* (Homeland Security, USA) https://ics-cert.us-cert.gov/Assessments (accessed March 09, 2015)

[33] Georg, G. et al. (2010) *Verification and Trade-Off Analysis of Security Properties in UML System Models*. IEEE Transactions on Software Engineering, Volume: 36, Issue: 3. Pages: 338-356.

[34] Aoki, Y., Matsuura, S. (2014) *Verifying security requirements using model checking technique for UML-based requirements specification*. In Proceedings of IEEE International Workshop on Requirements Engineering and Testing. Karlskrona, Sweden, Pages: 18-25.

[35] Cranor, L.F. and Buchler, N. (2015) *Better Together: Usability and Security Go Hand in Hand*. IEEE Security & Privacy, Vol. 12, Issue 6.

[36] Duckmanton, J. (2014). *Accelerating adoption of software tools and practices*. Jazz.net Blogs. http://bit.ly/1npJC9