Demonstrating Enterprise System Security Using an Asset-Centric Security Assurance Framework

Nikola Luburić, Goran Sladić, Branko Milosavljević, Aleksandar Kaplar
Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia
{nikola.luburic, sladicg, mbranko, aleksandar.kaplar}@uns.ac.rs

***Abstract*—Enterprise systems often require a high level of security. While securing such systems is a challenge in and of itself, proving that a system is sufficiently secure is an additional problem which is rarely discussed.**

This paper presents a security assurance framework that can be used to prove that an information system is reasonably secured. The framework is structured around data assets, their security goals, and their flow throughout a system. We present the domain model for our framework and describe how the asset inventory, data flow diagrams, and security assurance cases are used by it.

Finally, we demonstrate how the framework produces a scalable body of evidence that can be used to demonstrate the exercise of due care and due diligence, mitigating some of the issues that arise from expert judgment-based approaches.

I. Introduction

Large-scale enterprise information systems represent a complex, heterogeneous environment of application software which supports the workflow and business processes of organizations. Such systems are designed to manage large volumes of sensitive data and support mission-critical business operations. Apart from the functional requirements, these systems need to meet many quality requirements, such as usability, maintainability, efficiency, and security.

Security has become a leading concern for organizations of all scales. To protect the environment with which an organization interacts, regulatory bodies have issued data protection laws and regulations. Laws such as HIPAA, Sarbanes-Oxley, and more recently GDPR, impose fines for organizations that do not comply with their data protection requirements. In the event of a breach, organizations are required to demonstrate due care. This entails the provision of evidence which proves an organization was compliant with all security requirements defined by law at the time of the incident [1].

Additionally, security has become a business driver as it protects trade secrets, brand integrity, and customer trust. While a company might be legally protected if it can demonstrate due care, the company's brand might still be damaged. Demonstrating due diligence, on the other hand, might nullify all reputation damaged caused by the incident. Due diligence is a superset of due care and pertains to best practices that an organization should follow to keep its systems and data secure [1].

Achieving due care and due diligence is a challenge that requires significant effort. Likewise, proving that due care, and especially due diligence, was exercised, can be a difficult task if the undergone efforts aren't properly documented. Establishing and maintaining a body of evidence which demonstrates security assurance in a large-scale enterprise system can be a daunting task, due to the complexity of the system and the volume of assets that require protecting [2].

In this paper, we help solve this problem by providing a framework for documenting evidence for security assurance. We construct a domain model for our framework that contains information derived from:

- The asset inventory, which identifies assets;
- Data flow diagrams of a particular asset, listing components that manipulate the asset;
- The security policy that defines protective mechanisms and activities for components that manipulate assets.

By using this information, we produce asset-centric security assurance cases that demonstrate the level of the security of the system in question.

The rest of the paper is structured as follows: Section 2 defines the problem addressed by this research and lists the relevant related work. Section 3 presents our framework, describing the domain model and each step of its usage process. Section 4 provides an illustrative example of the usage process. In Section 5 the real-world application of this framework is discussed, and further research goals are proposed. Finally, in Section 6, we conclude our work.

II. Problem Statement

The work presented in this paper aims to answer the question of how to demonstrate that an enterprise information system is sufficiently secure. To precisely define our scope and research questions, we first discuss what a system is and what makes a system sufficiently secure.

A system can be defined as: (i) a product or component within an organization, (ii) the infrastructure needed to combine the products or components (i.e. network equipment, hardware), (iii) applications that are used to support everyday activities of an organization, (iv) the users, such as technical staff, management or customers, which interact with the information technology deployed in an organization. In general, a system is all of these things combined [3].

In order to answer what makes a system sufficiently secure, we analyzed the definitions of information security

and cybersecurity and examined leading security standards, such as the ISO 27k series [4] and several NIST security-related publications from the 800 series, including 800-53 [5] and 800-37 [6].

As a result of this analysis, we conclude that a system is sufficiently secure if the security goals of all its assets are reasonably achieved throughout the assets' lifecycle.

In this work, we limit our scope and only examine data assets. Therefore, the proposed security assurance framework can be used to demonstrate that due care and due diligence was exercised, but only in the context of data assets. Assets such as hardware, personnel, etc. are out of our scope.

We define our research question as follows:

How to structure a body of evidence which demonstrates that the security goals of all data assets are reasonably achieved throughout the asset's lifecycle? More succinctly, how to document proof that all sensitive data assets are reasonably protected?

Traditional techniques for demonstrating security assurance rely on expert judgment. This approach has been criticized for having several disadvantages, most of which are caused by the invisibility of the reasoning process, which, as a result, can't be maintained, improved or appropriately validated [7].

An alternative technique for demonstrating security assurance is by using security assurance cases [2]. This technique relies on semi-structured arguments and fixes some of the disadvantages listed in [7] by providing completeness of decomposition of claims and richness of argumentation. Several problems regarding security assurance cases have been highlighted, including the lack of support for structuring the information, scalability problems, and other [8]. The scalability problem is one our proposed framework aims to solve.

In order to demonstrate that the security goals of all data assets are reasonably achieved throughout their lifecycle, we need to define:

- What are data assets and how to represent them;
- What are security goals and how to represent them;
- How to track an asset through its lifecycle;
- How to document protective mechanisms and activities as evidence for security assurance.

To answer these questions, we constructed a security assurance framework, described in the following chapter.

III. Proposed Framework

The following section describes our proposed framework. First, we examine the domain model and describe each section of this model. Then we examine the usage process of our security assurance framework.

A. *Domain Model*

The proposed security assurance framework relies on the domain model which is presented in Fig. 1.

The main building block of the security assurance cases constructed by our framework is the asset. An asset can be described by a wide array of information, as detailed in standards such as ISO 27001 [4] and NIST 800-37 [6]. For our purposes, we require that an asset has a name with which it is identified and a description that signifies its purpose in the system.

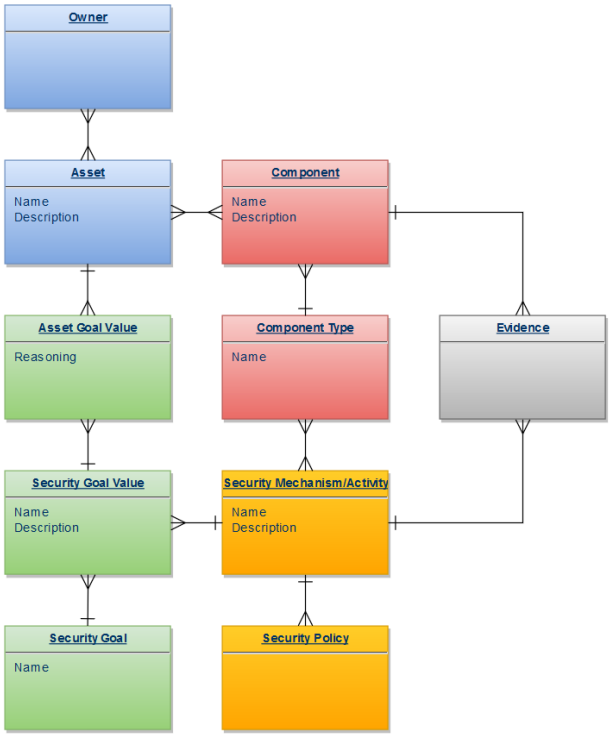

Figure 1 - Security assurance framework domain model

Each asset must have an owner. By assigning an owner to the main block of the security assurance case, we enable the scalable construction of the body of evidence. The owner is responsible for his or her assets and is likewise responsible for populating the domain model with information related to his or her asset.

A sensitive asset in our context is an asset that has one or more security goals related to it. A security goal can be, for example, confidentiality, integrity, or availability.

Each security goal has a set of possible values. A value for a security goal is identified by its name and offers a description which aids the asset owner in determining if his or her asset should be assigned the given security goal value. For example, for the security goal of confidentiality, the set of possible security goal values might include top secret, internal use, and public.

Once the appropriate security goal value has been selected for the asset under examination, the asset owner assigns the asset goal value and documents the reasoning behind choosing this value.

Protecting a data asset throughout its lifecycle can be achieved by providing appropriate protection to each component that stores, processes, or transmits the data asset. Therefore, our domain model entails the identification of components which manipulate a given asset. Examples of such components include a particular SQL database, HTTP communication between two application servers, as well as the application servers themselves. The component is identified by its name and further context is provided through its description.

As large systems can have hundreds of components, we group components according to their type. By introducing the component type, we can generalize security measures that need to be applied to all components within a system of a given type.

Next, we introduce the security controls in the form of security mechanisms and activities. Examples of security mechanisms include symmetric cryptography, access control checks, and logging controls. Examples of security activities include penetration testing, security awareness education, and system hardening.

According to the security policy, security mechanisms and activities are defined. They are identified by their name and further described through some form of description. Furthermore, a set of security mechanisms and activities is defined for each component type that manipulates data assets containing a particular goal value. For example, all components of the HTTP communication type that transfer top secret data assets need to be encrypted. The security mechanism for this scenario would be the application of TLS, while an additional security activity might be the hardening of the TLS configuration according to best practice guides.

The final block in our domain model is the evidence store, which contains a set of records that prove that a particular security mechanism or activity has been applied to a particular component. For the previous example, the appropriate evidence could be a TLS hardening report as well as a 3rd party security audit that confirms that TLS is installed.

B. Framework Usage

By following the framework usage steps, the domain model is populated with information which can then be used to generate security assurance cases. The steps of the framework usage process are as follows:

1. Asset inventory construction;
2. Data flow analysis;
3. Security control identification;
4. Security assurance case generation.

The first step entails the identification of assets and their owners. With the aid of the asset owner, the security goals related to the asset are identified and asset goal values are assigned. As mature enterprises should already have an asset inventory constructed, this step might already be completed when utilizing our framework.

Next, data flow diagrams [9] are used to map the flow of data assets across system components. By constructing data flow diagrams, all system components which store, process or transfer the data asset are identified and entered into the domain model. As with the previous step, mature enterprises might already have data flow diagrams constructed for their sensitive assets, resulting in the completion of this step before the introduction of the framework.

The third step of the framework usage process entails the identification of the security mechanisms and activities used to protect the system. The important step here is to tie each security control with the appropriate security goal values and component types. While mature enterprises often have an established security policy, transformative functions might be needed to adapt these policies to our domain model.

The last step of our usage process requires that evidence is entered into the domain model – documented proof that a certain security control is applied to a certain component. The framework can then be used to generate a security assurance case [8], as demonstrated in Fig. 2.

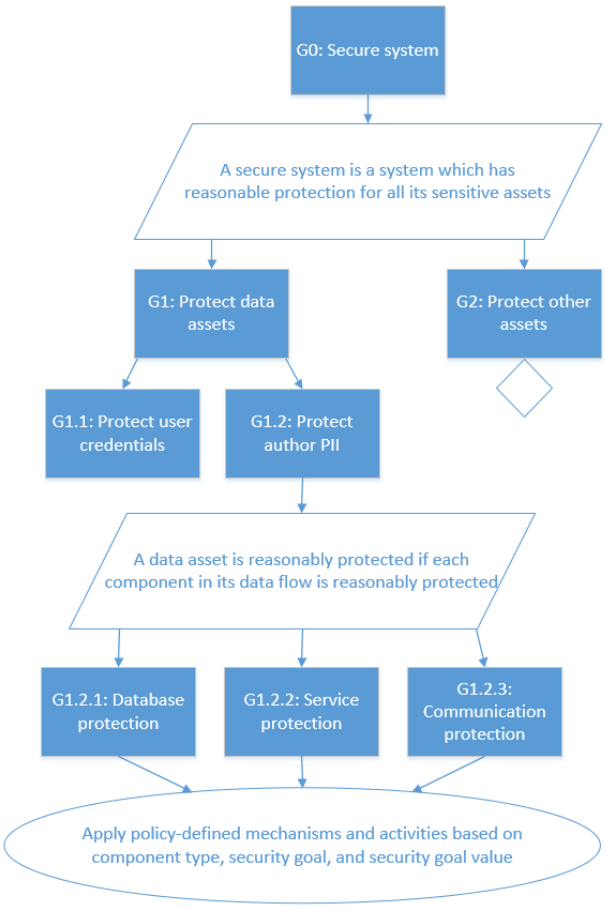

Figure 2 – Structure of the security assurance case generated by the proposed framework

The base goal of the enterprise system is for it to be secure. This goal is decomposed by following the definition that a system is secure if all its sensitive assets are reasonably protected. As the focus of our work is related to data assets, we decompose the goal into two goals – the protection of data assets and the protection of non-data assets.

The goal of protecting data assets is decomposed into N subgoals, where N is the number of data assets in the asset inventory, constructed in step 1. Each of these goals is related to the protection of a specific asset from the inventory.

Specific asset protection goals are further decomposed by following the statement that a data asset is sufficiently protected if all components that manipulate it are reasonably protected. By using information gathered using data flow diagrams in step 2, the asset protection goal is decomposed into a set of subgoals for each identified component.

Guided by the security policy constructed in step 3, tooling reports, auditing reports, and other forms of evidence can be constructed to prove that a specific mechanism or activity has been applied to a specific to a specific component.

The complete security assurance case can then be used to argue that the data assets of a system are reasonably protected. The authority examining the security assurance case needs to confirm that the reports are correct (and not, for example, forged), that the asset classification is reasonable and that the data flow diagrams are complete.

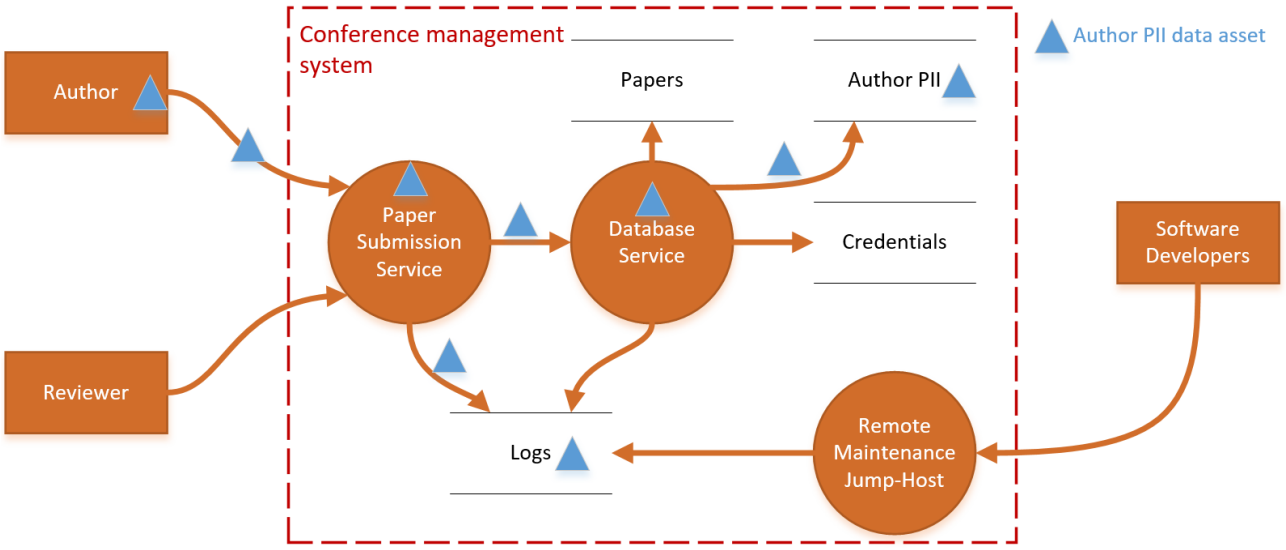

Figure 3 - Data flow diagram of the conference management system paper submission subsystem

IV. Illustrative Example

In this section, we illustrate the use of our framework on a subsystem of a hypothetical conference management system, related to paper publishing. The data flow diagram of this system is presented in Fig. 3.

We demonstrate the usage of our framework by analyzing the personally identifiable information (PII) of the authors using the system to submit their papers.

The first step is related to the construction of the asset inventory. In the context of this illustrative example, this entails the identification of the Author PII data asset. An owner of this asset is selected, from the pool of employees of the organization running the conference management system. By working with the asset owner, the security goals relevant to this data asset can be identified. As this system can accept submissions from authors who are residents of the European Union, GDPR might be a concern for this data asset. Therefore, the security goal of privacy is identified, and a security goal value of private is attributed to the Author PII data asset.

The second step entails the construction of data flow diagrams, in order to identify components that manipulate the Author PII data asset. This data asset originates from the authors themselves, being created using a web browser. The asset is then transported over HTTP to the Paper Submission Service, where it is forwarded to the Database Service that stores it. An additional location where the Author PII is stored is in the log files of the Paper Submission Service component. The log files are accessed by system developers to provide remote maintenance services but are not transported to the organization the system developers work for. Once all components are registered, the component types are defined for each component. For our example, this could include:

- The MySQL database for the Database Service that stores the Author PII;
- The Apache Tomcat application server running a Java Spring web application for the Paper Submission Service;
- The Linux file system that stores the Logs;
- HTTP communication for the flows between the Author and the Conference Management system as well as the communication between the Paper Submission Service and the Database Service.

As part of the third step, the security mechanisms and activities applied to each type of component are listed, based on the security goal values of the assets manipulated by components of this type. In the context of this example, this includes:

- Least privilege access control mechanisms applied to the MySQL database and the Java Spring application functionality;
- Input validation mechanism applied to the Java Spring application;
- System hardening activities applied to Apache Tomcat application servers;
- TLS mechanism applied to all HTTP communication;
- Least privilege access control lists applied to logs.

Finally, documentation in the form of tooling reports, auditing reports, or other forms of assessment can be produced to confirm that the previously mentioned security mechanism and activities have indeed been put into place to protect the given components.

V. Discussion

In this section, we discuss the practical application of our security assurance framework and point out its limitations.

The proposed framework relies on two strict assumptions for the complete reasoning of the produced security assurance case to hold:

1. The asset inventory must be complete;
2. The data flow diagrams are complete and correct.

Without listing all data assets, the security assurance case cannot claim that all data assets are sufficiently protected. Likewise, by not identifying all components that manipulate the data asset, the produced security assurance case cannot prove that a specific asset is sufficiently protected. For example, if the data flow diagram presented in Fig. 3 did not account for the logs produced by the Paper Submission Service, there would

be no evidence to prove that the logs have been protected by least privilege access control lists.

Our framework relies on the asset inventory, data flow diagrams for these assets, and the security policy of an organization. Mature enterprises have some, if not all of these elements already constructed. In this scenario, the practical application of our framework is viable, as it mostly requires the restructuring of existing documentation and evidence.

The benefit of this restructuring is two-fold. On the one hand, the resulting security assurance case introduces benefits discussed in [7]. This includes complete decomposition of the main goal of owning a secure system, as well as richness of the argumentation that proves this. On the other hand, by transforming existing documentation into our domain model, the enterprise can detect where the weakest links are, what assets are not sufficiently protected and plan their risk management accordingly.

For organizations that do not have these elements pre-built, using our framework and constructing and maintaining these elements can be a daunting task. Security assurance cases have been criticized for their scalability problems [8]. To help solve this issue, we based our framework on the asset, where the asset owner is responsible for all the relevant documentation for his or her asset. While this does not reduce the total work that needs to be done, it does offer a simple way to distribute and scale the workload.

The main limitation of our work is introduced by the asset, and this is the question of asset granularity [10]. Identifying data assets is a non-trivial task. A data asset can be generalized, as in the previous example, where Author PII was identified. However, a data asset can also be the email address of an author.

Likewise, data flow diagrams can be decomposed, where each component can be described in more detail by constructing a subdiagram, also called a higher-level data flow diagram [9]. For example, the Paper Submission Service can be decomposed into process nodes that include the operating system, the application server, and the application software. Likewise, the application software node can be further decomposed to describe the various modules of the software. The issue that arises here is that the enterprise might not be aware of the underlying technologies used by its software. Therefore, higher level data flow diagrams can only be produced by the software developers, which can be imposed as a requirement by the enterprise to the software vendor.

Granularity of data assets should be handled by the enterprise's asset management strategy. As for the granularity of data flow diagrams, this falls within the domain of threat modeling [11] and risk assessment. By having a high level of granularity of data assets, and detailed analysis of their data flow, an enterprise can demonstrate a mature security analysis of its system. Paired with sufficient security controls, this can be used to prove due diligence.

VI. Conclusion

In this paper, we presented a framework that is used to produce asset-centric security assurance cases. This documentation structure presents a body of evidence that can be used to prove due care and due diligence was exercised. The resulting structure can be constructed in a scalable way and can be more precisely evaluated than traditional judgment-based approaches.

The framework relies on several steps which include the construction of the asset inventory, the mapping of the data flows for each asset, and the determination of security controls for each component that manipulates a given asset. By providing evidence that proves the controls are in place, and using the reasoning logic provided by the produced security assurance case, the security of the system's data assets can be proved.

We have outlined the usage process, offering an illustrative example for better explanation. Finally, we have discussed the real-world application of the framework, listing best practices for its adoption and use.

Further work includes empirical evaluation of the framework and additional study and customization for real-world enterprises. Concretely, we aim to see how the framework can be integrated into the security development lifecycle of software vendors and how it can best be tailored to the agile development methodology, building on the work presented in [12].

References

[1] Gordon, A. ed., 2015. Official (isc) 2 Guide to the Cissp Cbk. CRC Press.

[2] Goodenough, J., Lipson, H. and Weinstock, C., 2007. Arguing Security - Creating Security Assurance Cases. Carnegie Mellon University.

[3] Anderson, R.J., 2010. Security engineering: a guide to building dependable distributed systems. John Wiley & Sons.

[4] Disterer, G., 2013. ISO/IEC 27000, 27001 and 27002 for information security management. Journal of Information Security, 4(02), p.92.

[5] NIST, S., 2003. 800-53. Recommended Security Controls for Federal Information Systems, pp.800-53.

[6] NIST, S., 2010. 800-37, Revision 1. Guide for Applying the Risk Management Framework to Federal Information Systems: A Security Life Cycle Approach, 16.

[7] Dawson, S., 2005. The Genesis of Cyberscience and its Mathematical Models (CYBERSCIENCE). SRI INTERNATIONAL MENLO PARK CA SYSTEMDESIGN LAB.

[8] Ankrum, T.S. and Kromholz, A.H., 2005. Structured assurance cases: Three common standards. In High-Assurance Systems Engineering, 2005. HASE 2005. 9th IEEE Int'l Symposium on. IEEE.

[9] Bruza, P.D., and Van der Weide, T., 1989. The semantics of data flow diagrams. University of Nijmegen, Department of Informatics, Faculty of Mathematics and Informatics.

[10] Shedden, P., Smith, W. and Ahmad, A., 2010. Information security risk assessment: towards a business practice perspective.

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

[12] Vivas, J.L., Agudo, I. and López, J., 2011. A methodology for security assurance-driven system development. Requirements Engineering, 16(1), pp.55-73.