Aligning Business Process Access Control Policies with Enterprise Architecture

Roman Pilipchuk
FZI Research Center for Information Technology
Germany
pilipchuk@fzi.de

Stephan Seifermann
Karlsruhe Institute of Technology KIT
Germany
stephan.seifermann@kit.edu

Robert Heinrich
Karlsruhe Institute of Technology KIT
Germany
robert.heinrich@kit.edu

ABSTRACT

Access control policies are a fundamental building block in meeting security and privacy requirements in organizations across business processes, enterprise architectures, and software architectures. Usage of different models for business processes and software makes eliciting and enforcing access control policies hard. Approaches like enterprise architecture management target complex mutual interdependencies between business and IT models but can be hard to apply. We suggest an approach to derive access control requirements from business processes and test compliance of software designs by data flow analyses. As a result, business processes and software designs are aligned w.r.t. access control requirements.

KEYWORDS

Business processes, Enterprise architecture, Access control requirements, Role-based access control, Data flow

ACM Reference Format:
Roman Pilipchuk, Stephan Seifermann, and Robert Heinrich. 2018. Aligning Business Process Access Control Policies with Enterprise Architecture. In *Central European Cybersecurity Conference 2018 (CECC 2018), November 15–16, 2018, Ljubljana, Slovenia.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3277570.3277588

1 INTRODUCTION

Organizations use business processes, enterprise architectures (EA), and software architectures to cope with the rising amount of functional and non-functional requirements stemming from different stakeholders. Security and privacy is a non-function requirement that affects business processes as well as IT systems. Adhering to security requirements given by law, by corporate risk management, or by customers is crucial. Access control is a fundamental building block of security and privacy that has to be implemented accurately.

Defining and enforcing access control policies is a challenging task. According to common IT service management practices like specified in ITIL [2], proper access control requires three steps on the business process level: First, service design managers and compliance managers must define assets to be protected. Second, organization-wide security strategies define the implementation of protection mechanisms. Third, compliance of the security strategy with regulations and laws for example the IT Security Act [14], General Data Protection Regulation (GDPR) [18], and the US Health Insurance Portability and Accountability Act [20] has to be ensured. A previous security review [15] of an information system exemplifies this. System designers have to carefully adopt the mechanisms defined on the business level. This is, however, not trivial as there are gaps in terminology, domain knowledge, domain-specific models, and used modeling tools. Enterprise architectures, especially enterprise application architectures, are supposed to fill these gaps.

According to Gartner [8], EA requires a holistic approach that does not only focus on technical solutions but involves stakeholders and business people. Alpers et al. [1] point out that this holds true with respect to privacy-aware modeling approaches for EAs in particular. To the best of our knowledge, there is no such approach to analyze whether EAs comply with access control requirements of IT security and privacy stemming from the business level.

We propose an approach to automatically transform access control requirements from the business level to the software design level. Business processes defined in the de-facto standard notation Business Process Model and Notation (BPMN) are source for deriving access permissions. Software architects use an architectural description language (ADL) to design the software system that supports structure, behavior, and deployment aspects such as UML or the Palladio Component Model (PCM) [16]. Data flow analysis tracks information exchange in the architecture and detects flaws concerning access control policies. A transformation creates the analysis goals based on the derived access control policies.

Our approach establishes traceability between access control requirements from business processes and realizations in software systems. Additionally, it makes mutual dependencies between models of business and IT comprehensible. This is done by mapping access control relevant elements between business processes and software systems to transfer information. As a result, the models used during design activities are consistent. It is intended to impose only little additional effort because we reuse already existing models of business processes and software architectures that have to be defined anyway. Assuming that business process designers created legally correct business processes, we can verify access control policies automatically.

The paper is structured as follows: Section 2 gives an overview on the state of the art. We introduce our running example in Section 3. In Section 4, we explain our approach and exemplify our defined process by applying it to an example. A discussion of our approach is covered in Section 5. Section 6 concludes the paper.

CECC 2018, November 15–16, 2018, Ljubljana, Slovenia

ACM ISBN 978-1-4503-6515-4/18/11…$15.00
https://doi.org/10.1145/3277570.3277588

2 STATE OF THE ART

Our approach focuses on supporting consistency between requirements of business processes regarding access control and their realization in the software design. We identified two types of work closely related to our approach: a) modeling approaches for consistent handling of information between business processes and system design, and b) access control policy analyses of systems.

Consistency between business processes and software systems provides considerable benefits as described by Giaglis [5]. We call this *alignment*. Enterprise Architecture Management (EAM) is a common approach to achieve this alignment. In architecture-driven IT management (ADRIMA) [12], EAM involves initiating, establishing of processes and governance, and definition of application scenarios. The initiation of EAM defines the involved models and their lifecycles. Frameworks including TOGAF [17] and FEAF [19] define model types and their relations to build a complete EA. While they are well known, they are hard to apply because of the various reasons Kotusev [10] collected. One reason is the complexity of the whole process and the related detailed planning. Löhe and Legner [12] consider EAM a challenging task, in general. The EAM approaches provide the required models to represent security policies, threats, counter measures, and so on. Creating them, however, requires many stakeholders and deep knowledge of the specific models. In contrast, our approach exploits well-known and already used models and thereby lowers the inhibition threshold and effort for applying it. Heinrich et al. [6] proposed an approach to reflect mutual dependencies between business processes and EA but he focuses on performance predictions rather than access control.

Analysis approaches for software design focus on detecting contradictions between requirements and modeled systems. Nguyen et al. collected model-based security analyses in their survey [13]. Many approaches such as UMLSec [9] or SecureUML [11] leverage UML profiles to extend UML models and analyze security properties such as access control. They, however, focus on analyses of structural views or control flows, which can lead to less precise analysis results. This is not sufficient because only wrong calls can be identified instead of errors in the underlying behavior.

3 RUNNING EXAMPLE

We illustrate our approach on the running example of the community-driven case study Common Component Modeling Example (CoCoME) [7]. CoCoME is subject to a broad amount of security regulations [15]. The system covers the IT systems required by a supermarket chain. Our running example starts with the basic CoCoME version and we assume that there is a loyalty program and a marketing division. Besides other requirements, we define that customers must agree on using their order for marketing purposes.

We focus on two business processes: The process in Figure 1 shows preparation of advertisements and discounts by the marketing manager and store manager based on loyalty program data. The store manager creates an advertisement request. The marketing manager creates an advertisement schedule, prepares customer profiles using a set of *LoyaltyOrder*, and creates the advertisements.

An excerpt of the enterprise application architecture supporting the business process defined before is shown in Figure 2. Grey elements are part of the default CoCoME without any modifications

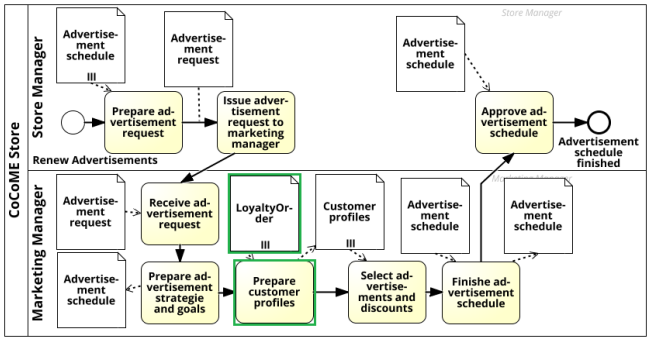

Figure 1: Business process of preparing ads and discounts.

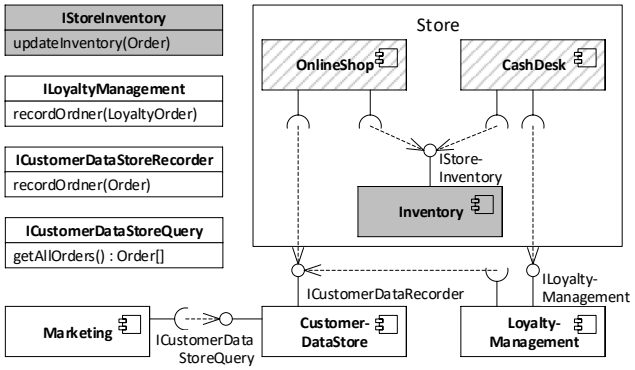

Figure 2: Simplified enterprise application architecture.

applied; shaded elements are extended by us to fit our scenario. The *Store* handles sales and the inventory. The store informs the *LoyaltyManagement* if the customer handed a loyalty card to the cashier. The *CashDesk* combines the information about the loyalty card and the order in a *LoyaltyOrder*. The *LoyaltyManagement* performs calculations about gained points and sends the order to the *CustomerDataStore* recorder. This information is available to the *Marketing* component. Marketing managers use the *Marketing* component to create advertisements based on previous orders.

We assume that business processes and EA are compliant for the setting described above. We now evolve the system to include an online shop in the store. The extension requires new business processes and software components. We focus on changes affecting the described business processes and software system. The CoCoME technical report [7] covers a full description of further changes.

The online shop introduced the type *OnlineOrder*. Business experts adjust the business processes and introduce *OnlineOrder* as input data in the customer support process. The shop shall not support the loyalty program, so other processes do not change.

Software architects have to adjust the information system as well. They introduce a new component *OnlineShop* to the store. The architects connect the *OnlineShop* to the *CustomerDataStore* to record orders of the *OnlineShop* as well. They do this because the customer support process requires common handling of online shop and loyalty orders and the requirements did not clearly state that only loyalty orders shall be processed for marketing purposes. The marketing manager can now use both types of orders to create advertisements, which is not intended and breaks compliance.

Table 1: Extracted role model (r: read, w: write).

Role	Permission
Store Manager	r/w ad request; r ad schedule
Marketing Mngr	r/w ad request; r/w ad schedule; **r LoyaltyOrder**; r/w Profile
Customer	r/w Support ticket
Customer Service	r/w Support ticket; **r LoyaltyOrder**; **r OnlineOrder**

4 ANALYSIS APPROACH

Our proposed approach supports the alignment of business processes and EA with respect to access control policies. We assume that business processes and an EA already exist. We assume business processes to be compliant with law as making them compliant is not the focus of our approach. Our automated approach extracts access control requirements from BPMN and transforms them to a role model for role-based access control systems (RBAC) as well as to architectural data flow constraints to identify forbidden data flows in the EA. We roughly describe the five phases of our approach and apply them to our running example. The phases do not have to be conducted in a top down fashion. An organization can start at any phase. Most organizations will omit phase 1 because they already have modeled processes and EA. Phase 2 and 3 produce the RBAC role model. Phase 4 and 5 analyze compliance of architectures with requirements from business level.

Phase 1) Modeling business processes and enterprise architecture: In this phase, organizations model their business processes and EA. New organizations typically do this to reflect their organizational services and products. Most medium to large organizations will omit this phase as they already have both. In our running example, we showed excerpts of business processes and an EA in Section 3.

Phase 2) Extracting access control requirements: Our approach extracts access control requirements from business processes in BPMN without user interaction or any prior model modifications. Instead, a transformation model combines elements of business processes with elements of access control policies. This establishes traceability between models. BPMN lanes and data objects map to roles and permissions. Business processes and activities are in between of them. Consequently, each role occurs in certain business processes where it has an amount of activities. To carry out these activities, it needs specific permissions in form of object-operation pairs. We compile the complete policies by applying the transformation to each business process. Table 1 illustrates the role model. Finally, searching roles with a subset of permissions of other roles forms a simple hierarchy. In terms of completeness, the role model lacks technical access control requirements. Granularity and coverage of business processes affects the results as well.

Table 1 shows the role model containing the extracted access control requirements from the business processes of the running example after applying the above-mentioned approach. The highlighted roles and permissions are most relevant for the running example. The green highlighted parts of the BPMN in Figure 1 imply that the *Marketing Manager* needs read access to *LoyaltyOrder* during the activity *Prepare customer profiles*. Read access arises from the arrow pointing to the activity, meaning a data input. Arrows in the other direction meaning a data output indicating a write access. This applies to all data objects in the BPMN, as well as to the customer support process explained in Section 3 implying that the *Customer Service Employee* needs read access to *LoyaltyOrder* and *OnlineOrder*.

Phase 3) Security experts refine initial role model: This phase is independent from our approach and has to be conducted in any case. Security experts engineer the role model [4]. We ease this phase by providing an initial role model in phase 2. This frees security experts from mining requirements by conducting interviews and business process analyses. Instead, security experts only refine the initial role model with missing technical access control requirements. As a benefit, the role model is aligned with the business level needs and security experts are bothered less with business level artefacts.

Phase 4) Architectural data flow analysis of enterprise architecture: The extracted access control requirements from phase 2 and 3 are merged into the architectural data flow analysis model and analyzed afterwards. The model consists of a) users and their interaction (service calls) with the system, b) the signature of system services including their behavior description, and c) the assignment of services to computing resources. The service behavior consists of data processing operations and data transmissions between them. A simplified version of the system behavior without service boundaries using the DeMarco syntax for data flow diagrams [3] is given in Figure 3. The figure illustrates data transmissions and data processings. For every operation, we define the effect on characteristics of the processed data, which is not shown in the figure. Access restrictions is the characteristic relevant for our scenario. In our running example, the *Prepare Customer Profiles* operation takes *Order* data and produces *Profile* data. This operation does not perform anonymization. Therefore, we define it to copy the access restrictions from the input data to the output data: *Order* is a shorthand writing for data that can be *LoyaltyOrder* or *OnlineOrder*. Access to *Order[]* data loaded from the *Customer Data Store* is restricted to the customer service because all others do not have access rights for both *LoyaltyOrder* and *OnlineOrder*. Therefore, *Profile* is only accessible by the customer service as well. If the operation would perform anonymization, we would define it to assign the extracted access rights for *Profile*, which would include the marketing manager. Software architects have to define the operations and their effect on data characteristics only once but use instances of this operation many times in their models. Finally, the merging algorithm assigns the extracted roles to users and services in case of system actors that trigger actions.

The analysis looks at every data flow originating from a user or system actor. For every operation, the analysis derived the characteristics of the processed data. As soon as data arrives at a system actor, the analysis compares the roles of the actor with the derived access restrictions and reports violations.

While using the approach is meant to be straightforward, defining data processing operations can be challenging. Software architects have to choose abstractions carefully to facilitate reusability and analyzability. Mapping business process and software architecture entities can be challenging as well. We suggest a matching by type and name that software architects refine manually.

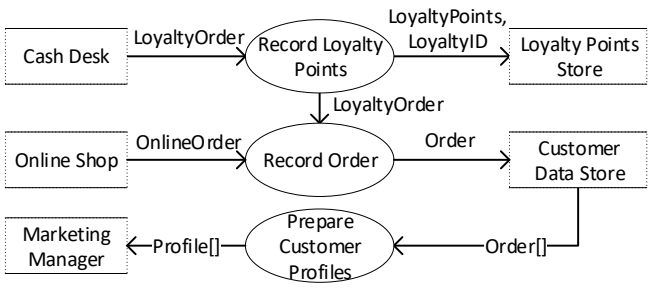

Figure 3: Simplified data flow diagram of order recording.

Phase 5) Error resolution in the enterprise architecture: The enterprise architect uses the result of the previous phase for architecture refinement. Due to the traceability of access control information between models, the visualization of the violations is able to show the data path, violated access rights, and the affected business process including the responsible activity. This information helps enterprise architects understanding design decisions of business level and information that is allowed to flow. Both helps to solve the violation faster and correctly. In our running example, the marketing component must only use *LoyaltyOrder* data instead of all orders.

Even if our approach supports identifying issues and their cause, fixing design flaws is still challenging: Shown causes do not have to be root causes. Proper identification of the root cause still requires communication between business experts and software architects. Architects can test solutions w.r.t. requirements but should still question business decisions implying high costs and efforts.

5 DISCUSSION

Our approach is applicable to many organizations because it operates on widely used and de facto standard modeling languages. Neither extensions to BPMN need to be used, nor any additional information needs to be provided. In case of the EA, the most common language UML could be used for data flow analysis but also other architectural design languages like for example Palladio Component Model [16] are possible. Our approach requires modeling of structural, behavioral, and deployment aspects. The behavioral aspects have to be given in terms of data flows or it has to be possible to derive data flows from the behavior description. Nevertheless, the main point is that for business processes and for EA only few additional information is required.

Our approach is not limited to newly created organizations but can be integrated with existing architectures. Even if we defined a sequential process, the origin of the required input data is not relevant. Organizations can use our approach to cross-check existing artifacts by comparing them with our generated artifacts. For instance, if organizations already have a RBAC policy, it can cross-check their policy with the derived policy to identify mismatches.

Our approach supports comprehensibility by business experts and enterprise architects by introducing traceability between elements and data flows in the EA and elements of business processes. Dependencies between both are hard to detect and often complex. Enterprise architects now better understand design decisions and the requirements behind them. The architectural data flow analysis allows identifying forbidden data flows resulting from design errors and enables the enterprise architect to solve these issues. Organizations can understand problems originating from evolution of processes and systems easier and with less effort.

6 CONCLUSION & FUTURE WORK

Alignment between business processes and system designs is not trivial. We proposed an approach to tackle this challenge w.r.t. access control policies: Business experts derive policies from business processes and refine them. Software architects use the policies to analyze compliance of software behavior using data flow analyses.

The benefit of applying our approach is a better understanding of implications of business process requirements on software design. Single steps of our approach support business experts and software designers to check feasibility of their modeled artifacts. In evolution scenarios, our approach eases detection of introduced flaws.

In our future work, we will implement the approach for specific modeling tools and afterwards apply it to several case studies. To construct valuable case studies, we plan to mine repositories of companies that model business processes and information systems.

ACKNOWLEDGMENTS

This work was supported by the German Federal Ministry of Education and Research under grant 01IS17106A (Trust 4.0).

REFERENCES

[1] S. Alpers et al. 2018. Identifying Needs for a Holistic Modelling Approach to Privacy Aspects in Enterprise Software Systems. In *ICISSP'18*. 74–82.
[2] AXELOS. 2011. ITIL Edition 2011. Retrieved 22.03.18 from https://www.axelos.com/best-practice-solutions/itil/what-is-itil
[3] Tom DeMarco. 1979. *Structured analysis and system specification.* Prentice-Hall.
[4] P. Epstein et al. 2001. Engineering of role/permission assignments. In *Seventeenth Annual Computer Security Applications Conference*. 127–136.
[5] G. M. Giaglis. 2001. A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques. *Int J Flex Manuf Syst* 13, 2 (2001), 209–228.
[6] R. Heinrich et al. 2015. Integrating business process simulation and information system simulation for performance prediction. *SoSyM* (2015), 1–21.
[7] R. Heinrich et al. 2016. *The CoCoME Platform for Collaborative Empirical Research on Information System Evolution.* Technical Report 2. Karlsruhe.
[8] Gartner Inc. 2009. Gartner Identifies Ten Enterprise Architecture Pitfalls. Retrieved 04.07.18 from https://www.gartner.com/newsroom/id/1159617
[9] J. Jürjens. 2005. *Secure systems development with UML.* Springer.
[10] S. Kotusev. 2017. Critical Questions in Enterprise Architecture Research. *IJEIS* 13, 2 (2017), 50–62.
[11] T. Lodderstedt et al. 2002. SecureUML: A UML-Based Modeling Language for Model-Driven Security. In *UML'02*. 426–441.
[12] J. Löhe et al. 2014. Overcoming implementation challenges in enterprise architecture management: a design theory for architecture-driven IT Management (ADRIMA). *ISeB* 12, 1 (2014), 101–137.
[13] P. H. Nguyen et al. 2015. An extensive systematic review on the Model-Driven Development of secure systems. *IST* 68 (2015), 62–81.
[14] Federal Republic of Germany. 2015. IT Security Act. Retrieved 22.03.18 from http://www.bgbl.de/xaver/bgbl/start.xav?startbk=Bundesanzeiger_BGBl&jumpTo=bgbl115s1324.pdf
[15] R. Pilipchuk et al. 2017. Defining a Security-Oriented Evolution Scenario for the CoCoME Case Study. In *EMLS'17 (Softwaretechnik Trends)*, Vol. 37. 60–77.
[16] R. H. Reussner et al. 2016. *Modeling and Simulating Software Architectures – The Palladio Approach.* MIT Press.
[17] The Open Group. [n. d.]. TOGAF Standard, Version 9.2. http://pubs.opengroup.org/architecture/togaf9-doc/arch/. accessed 16.05.18.
[18] European Union. 2016. General Data Protection Regulation (GDPR). Retrieved 22.03.18 from http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679&from=EN
[19] L. Urbaczewski et al. 2006. A Comparison of Enterprise Architecture Frameworks. *IIS* 7, 2 (2006), 18–23.
[20] U.S. Department of Health & Human Services. 2015. Health Information Privacy. https://www.hhs.gov/hipaa. accessed 30.08.18.