I. Schaefer, D. Karagiannis, A. Vogelsang, D. Méndez, C. Seidl (Hrsg.): Modellierung 2018,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2018 71

Towards a run-time model for data protection in the cloud

Zoltán Ádám Mann, Andreas Metzger, and Stefan Schoenen [1]

Abstract: The protection of sensitive data in the cloud is a challenge of increasing importance. It is made particularly difficult by the complex and dynamic interactions of many entities (hardware and software, as well as organizations and individuals). A model-based approach can be used to reason about these interactions and their impact on data protection during deployment and at run time. The basis for such an approach is a model of all relevant socio-technical cloud entities, which is created during deployment and kept alive at run-time to support adaptations. In this paper, we focus on the meta-model of this model. The meta-model is created during design and instantiated during deployment. We discuss what entities must be present in the meta-model to allow reasoning about data protection. In particular, we discuss to what extent the results of previous cloud modeling efforts can be reused and what extensions are necessary because of the particular requirements of data protection.

Keywords: Models@runtime; cloud computing; data protection; privacy

1 Introduction

The compelling advantages of cloud computing, such as the instantaneous access to services and seemingly infinite compute power without the need for costly IT equipment, have made the cloud the platform of choice in many domains. The cloud is a complex and highly dynamic environment. For example, the active user base of cloud services is continuously changing, and so is the intensity with which cloud users use the services. Also the specific requirements of the users keep changing. New services are added, existing services upgraded, old services removed, and so on. To adapt to changes in the workload, software components are scaled in or out or migrated between servers.

The flexibility and dynamism offered by cloud computing is an advantage for cloud users, as they can access and pay for compute and software resources on demand [Ma15b]. Yet, at the same time, this flexibility and dynamism implies data protection risks. Protecting sensitive data in the cloud is becoming an important limiting factor of cloud adoption [Mo13]. Also requirements and constraints regarding data protection[2] can change continuously. As an example, users may change their preferences regarding how their personal data is handled by cloud services. The General Data Protection Regulation (GDPR) of the European Union

[1] The authors are with paluno – The Ruhr Institute for Software Technology, University of Duisburg-Essen

[2] In this paper, we use the term *data protection* to refer to the protection of sensitive data. This includes privacy (protection of personal data). Security concerns are included, as long as they relate to the protection of sensitive data (but security issues that do not impact sensitive data are not relevant here).

[Co16] increases the breadth and depth of control that users (in this context called data subjects) have about their data. This makes it easier for users to withdraw their consent to processing and storage of their data, but in turn exhibits new challenges for cloud service providers. In the flexible and dynamic setting of cloud systems, the applicability of traditional security mechanisms that were designed to keep the system in a stable secure state is limited [BKW14]. In particular, security-by-design methodologies are not sufficient anymore, due to uncertainty at design time of how the cloud and privacy requirements may dynamically evolve and change at run time.

A possible approach to cope with continuously changing data protection requirements in a dynamic cloud environment is to apply run-time monitoring and adaptation. This way the system can adapt at run time to changes in both the cloud and the data protection requirements, ensuring that requirements are met in the presence of changes, with minimal impact on other quality metrics like performance and costs [Ma15a]. To enable effective adaptation at run time, a run-time model, i.e., a model of the system and its environment available for reasoning at run time, is of central importance [Am12].

Therefore, our aim is to devise a run-time model of the cloud, which is useful for detecting and mitigating data protection violations. More specifically, we focus on the key modeling concepts required to cover the main elements of a cloud system in a run-time model of the cloud, leading to a *cloud meta-model for data protection*. The challenge in devising the meta-model of the cloud is to determine a sufficient level of detail as well as the necessary scope of modeled entities. Data protection concerns relate to all layers of the cloud stack, including, for example, secure hardware capabilities, co-location of virtual machines of different tenants on the same server, encryption of the communication between application components, and data anonymization. Moreover, the actors (organizations and individuals) as well as their goals and relations may play an important role. The challenge, therefore, is to devise a holistic model encompassing all relevant entities.

The approach followed in this paper can be summarized as follows. (i) We identify the types of information that the meta-model must contain in order to serve as a basis for assessing data protection issues. (ii) We specify the entities and their most important relations in a possible cloud meta-model. (iii) A scenario from an industrial context is used to validate the applicability of the suggested meta-model.

2 Industrial cloud scenario

To devise an appropriate model, it is important to first understand the purpose of modeling. In our case, this means that we need to understand the types of data protection violations that we want to be able to detect. To this end, we look at an industrial cloud setup and its implications on data protection. This scenario has been defined in the context of the project "RestAssured – Secure Data Processing in the Cloud"[3] jointly with several industry partners.

[3] https://restassuredh2020.eu/

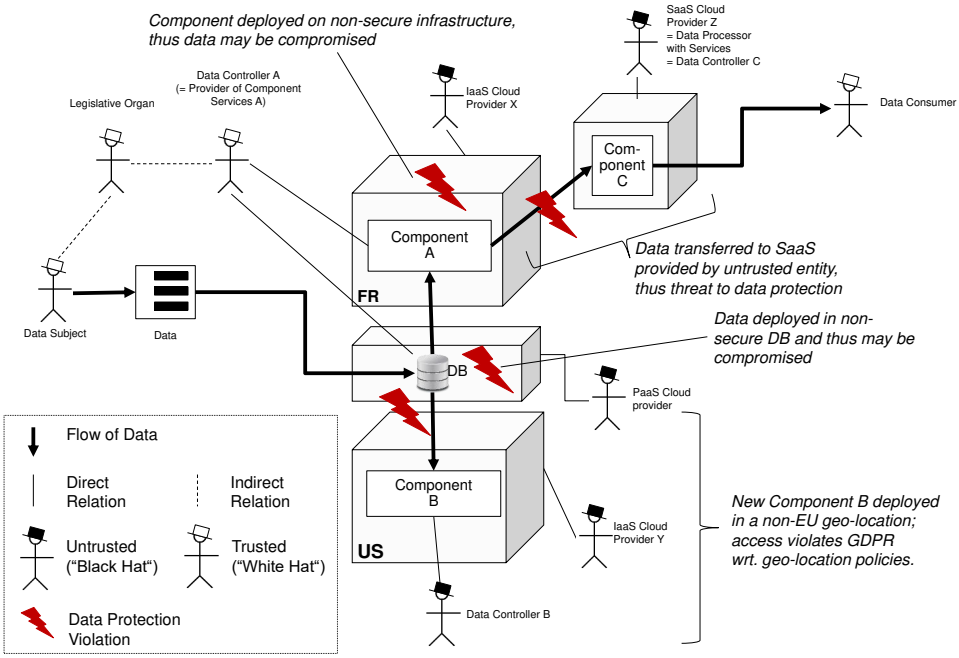


Fig. 1: An industrial cloud scenario

While the scenario abstracts from specific applications, it has been validated to reflect the typical data protection concerns of practical cloud systems.

In Fig. 1 we see a typical cloud scenario where multiple parties are involved. In this scenario, personal data about individual users (Data Subject[4]) are captured by a company (Data Controller A), with explicit consent of the users and under legislative control. The company stores the data in an unencrypted database (DB) operated by a PaaS (Platform as a Service) provider and deploys its application (Component A) using the infrastructure (including a virtual machine in which Component A runs and a physical machine in which this virtual machine runs) provided by IaaS (Infrastructure as a Service) provider X. Another actor (Data Consumer) uses an application (Component C) to communicate with Component A and get access to the data. Component C is run by SaaS (Software as a Service) provider Z. Another company (Data Controller B) uses an application (Component B) run on the infrastructure of IaaS provider Y that accesses the same database DB.

It is important to note that Data Controller A and Data Consumer are trusted by Data Subject, as shown by their white hats[5] in the figure. Nevertheless, as the data traverse between the trusted parties, several untrusted actors (PaaS provider, IaaS provider X, SaaS provider

[4] The names of the roles are based on the terminology of the GDPR

[5] Note that here white and black hats encode trustworthiness and do not refer to white-hat / black-hat hackers.

Z) may get unauthorized access to the data along the way. Moreover, other cloud tenants using the same public database offering (DB) can also get access to the data. The access by Component B also poses a further problem because Component B is hosted in the US, but European regulations prohibit processing personal data of EU citizens outside the EU.

As we can see, a cloud setup can be complex and dynamic, with many different socio-technical interactions, posing a wide-ranging set of threats to data protection.

3 Related work

Multi-layer cloud meta-models Shao et al. proposed a model-based approach for cloud monitoring [Sh10]. Their approach, called RMCM (Runtime Model for Cloud Monitoring), differentiates between three roles: cloud operator, service developer, and end user. Monitoring is used to adjust a model of the cloud consisting of four layers: (i) infrastructure, (ii) middleware, (iii) application, and (iv) interactions between the roles and the cloud. The models inside the layers are not specified; the focus of that work was rather on how the different layers can be monitored.

The NIST Cloud Computing Reference Architecture [Na11] also defines a kind of model of cloud computing. It mainly focuses on the definition of roles (cloud service provider, cloud service consumer, cloud broker, cloud auditor, cloud carrier) and the associated activities. In connection with the service provisioning and orchestration activities, the model foresees three layers: (i) physical resource layer, (ii) resource abstraction and control layer, (iii) service layer. However, the model does not include the actual components that make up a cloud system. Privacy is mentioned as an important requirement, but no details are provided.

Marquezan et al. [Ma14] developed a conceptual model of all the key entities relevant for adaptations in the cloud, based on a survey of the literature, discussions with industrial partners, and the analysis of commercial solutions. The resulting model consists of four layers: (i) The *physical layer* contains the physical equipment, like servers. (ii) The *virtualization layer* consists of virtual resources, like virtual machines. (iii) The *logical application architecture layer* is composed of the software components needed to support the logical architecture of the application, like application servers. (iv) The *application business logic layer* contains the components actually implementing the business logic of the application. Since the model of [Ma14] focuses on the adaptation possibilities of the cloud, it contains beside the actual cloud entities also the adaptation techniques (e.g., load balancing) in the same model. On the other hand, data protection was not in focus, so that the model does not support reasoning about pieces of data, actors, and relations among them.

Meta-models for cloud applications Several works focused on the highest cloud layer, i.e., the application layer. Chapman et al. addressed the problem of defining the architecture of multi-cloud software systems and proposed a model-based approach [Ch12]. The resulting

models target the application layer and only include some references to lower layers. The language defined in that paper relies on the Open Virtualization Format (OVF), but extends it with several new concepts, e.g., for elasticity rules. Nagel et al. presented a meta-model supporting the adaptation of business processes realized as cloud services [Na12]. The meta-model focused on the business process models, their adaptation and their mapping on cloud services, and abstracted from the technical infrastructure underlying the cloud services. However, some important threats to data protection relate to the underlying technology stack.

Heinrich proposed an architectural run-time model-based approach for analyzing cloud applications [He16]. That work uses a megamodel to connect a design-time architecture model, an architectural run-time model, and the actual implementation to enable correct interpretation of monitoring events and keeping the run-time model in sync with the state of the program. That method could be combined with our approach to provide effective monitoring for analyzing data protection issues. Bergmayr et al. proposed the Cloud Application Modeling Language (CAML) to facilitate expressing cloud-based deployments directly in UML [Be14]. Later on, that approach has also been extended to enable application provisioning by means of TOSCA [Be16]. That approach underlines the applicability of UML for modeling not only applications but also cloud environments. However, the approach lacks support for several concepts related to data protection, like the trust among stakeholders.

Model-based approaches for cloud security and privacy Similarly to our approach, the work of Schmieders et al. also applies model-based adaptive methods to data protection in the cloud [SMP15a, SMP15b]. That work, however, is limited to one specific type of privacy goals: geo-location constraints. Our work, in contrast, addresses data protection goals in a much broader sense.

Kritikos and Massonet proposed a domain-specific modeling language for modeling security aspects in cloud computing [KM16]. This includes security controls, security properties, security metrics, and security capabilities. In contrast, our work focuses on modeling the cloud – in particular, the possible attack surfaces and the configurations that may lead to data protection violations.

Other cloud models The MODACLOUDS project proposed a model-based approach to design, deploy, and maintain cloud applications [Ar12]. To enable multi-cloud deployments and migrations between clouds, MODACLOUDS adopted a model-driven approach: a Cloud-enabled Computation Independent Model (CIM) is transformed semi-automatically via a Cloud-Provider Independent Model (CPIM) to a Cloud-Provider Specific Model (CPSM). Unfortunately, the approach gives little support on what exactly needs to be in the models.

Industry cooperation resulted in TOSCA, a standard for cloud deployment topology and orchestration specification [OA13]. TOSCA advocates a generic template-based specification approach, in which services are specified in terms of node templates and relationship templates, and deployed by applying a deployment plan. While the generality of TOSCA allows the specification of any service, it lacks support for explicitly reasoning about data-protection-relevant aspects, like the location of data. Lejeune et al. took an even more generic approach and developed an abstract service model [LAL17]. This allows to describe any cloud service in terms of used services and components and offered SLAs. However, the abstract model hides the underlying technical components and their interrelations, which can be important for assessing the fulfillment of data protection policies.

Model-based approaches have also been proposed for evaluating the design of cloud systems. Relating to privacy, Ahmadian et al. devised a methodology based on the concept of Privacy Level Agreements [Ah17]. Such approaches are orthogonal to ours and an interesting path for future research is to investigate the integration of design models and run-time models.

Recently, we have introduced a method for detecting violations of data protection policies in cloud systems at run time [SMM17]. Our method is based on identifying "risk patterns" – configurations that would lead to unacceptably high risks of data protection violations – in a model of the cloud. Thus, having an appropriate run-time model of the cloud is a prerequisite for the method to work.

4 Design considerations for the meta-model

Based on the requirements from the example of Sec. 2 and the analysis of related work in Sec. 3, the cornerstones for an appropriate meta-model can be established as follows.

- Often, data protection violation is not confined to a specific cloud layer, but arises from the *interplay of entities belonging to different cloud layers*. For instance, a data record accessed by an application hosted by a virtual machine on a physical machine might constitute a data protection violation because it potentially allows the administrator of the physical machine to access the data. However, if the data are encrypted, or the application is protected by appropriate access control mechanisms, or the physical machine supports secure hardware enclaves, data protection can still be ensured [MM17]. Therefore, it is important to model the attributes of as well as interactions among entities on different cloud layers.
- Beyond technical entities like physical and virtual machines, also *actors* and their attributes and relationships need to be modeled. For example, if the data belonging to actor A can get accessed by actor B, this may or may not constitute a data protection violation depending on whether A *trusts* B or not.
- In existing cloud models, *data* was also missing. For reasoning about data protection, we need to add support for modeling data. This is not to be confused with "data

modeling," which is about modeling the logical concepts captured by the data. For our purposes, other attributes of data objects matter, like their sensitivity (e.g., personal data) and where they are stored and processed.

- *UML* provides sufficient expressiveness to model cloud systems, as shown by [Be14]. Although there are also alternatives, we stick to UML because of its wide-spread adoption and the available tool support. More specifically, we will use UML class diagrams for the meta-model and object diagrams for the model of the cloud.
- Beyond the mere detection of data protection violations, the model should also support finding the right mitigation action. For this purpose, it is vital to also model the possible *data protection mechanisms* and their impact on security attributes. Moreover; there may be multiple mechanisms that can be used to achieve the same security goal; in this case it is useful to select the one that has the smallest impact on other goals like performance or costs. For supporting such decisions, it is important to also model the *impact* of the available security mechanisms on those other *goals*.
- To be useful, the cloud meta-model needs to mirror the used technologies. However, there are several different technologies used in cloud computing and the technologies are also subject to change. For example, some cloud systems use virtual machines, others use containers, and a combination of the two is also possible. Therefore it is not feasible to strive for a cloud meta-model that is generic enough to capture all possible technical cloud realizations and at the same time also detailed and specific enough to allow reasoning about data protection impacts of a given cloud configuration. Rather, we argue that the exact cloud meta-model has to be created during system design, taking into account the specific technologies that are foreseen for the given system. We support that process by identifying the sorts of entities that need to be modeled, resulting in a framework for the meta-model, and giving examples of the modeling of entities that play an important role in most cloud systems.

5 Proposed meta-model

Based on the considerations of Sec. 4, we now propose a cloud meta-model for data protection. This meta-model can be seen as an extension of the existing multi-layer models of cloud systems discussed in Sec. 3. We extend those models with several concepts that are vital for data protection, like explicitly modeling data and actors.

5.1 Structure of the meta-model

Independently from the used technologies, our meta-model framework is structured into four high-level packages as shown in Fig. 2 and explained below:
Assets: configuration of the cloud, including all the physical and virtual entities that are

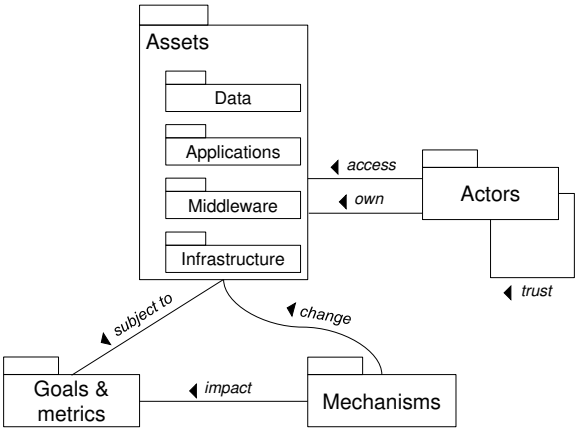


Fig. 2: The structure of the proposed meta-model

important for data protection
Actors: stakeholders and their roles relevant for data protection in the cloud at run time
Goals & metrics: non-functional properties that the system should fulfill
Mechanisms: possibilities for adaptation, including structural changes and changes to specific attributes

Fig. 2 also highlights the most important relations between the packages. In particular, assets may be owned and/or – independently from that – accessed by actors. In terms of relations among actors, *trust* is of special importance. We use a white-list approach to trust, i.e., every trust relation must be explicitly established (e.g., by means of a contract). Also, trust relations can be limited to specific types of actions on specific data. Goals & metrics relate to some assets, e.g., by specifying a response time constraint on an application. The mechanisms change some assets and impact the goals & metrics. For example, encrypting a piece of data changes an attribute of that data object and it has some given positive impact on confidentiality but negative impact on performance.

The Assets package is further subdivided into Infrastructure, Middleware, Applications, and Data. Traditionally, cloud models include only the first three of those, but we also included data because of its obvious importance to data protection. Data is the primary asset at risk, but the other layers are also important because they act as additional attack surface.

Altogether, the proposed meta-model consists of seven *sub-models*: the four packages within Assets, plus Actors, Goals & metrics, and Mechanisms. These sub-models are detailed next.

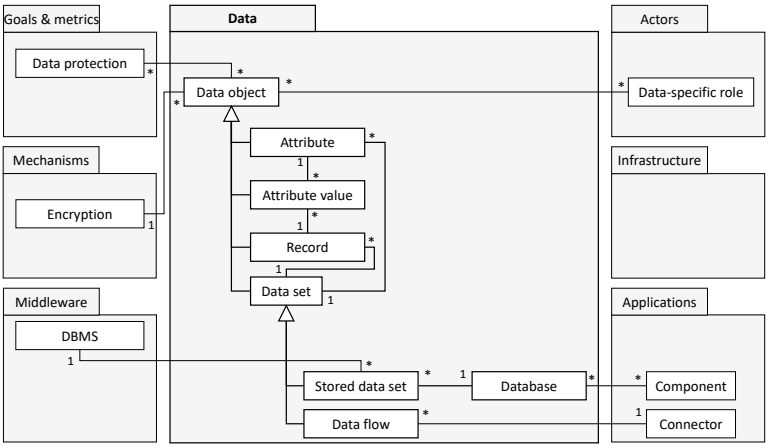


Fig. 3: The sub-model Data and its relations to the other sub-models

5.2 Contents of the sub-models

The structure of the meta-model presented above is technology-independent and hence it can be expected to remain stable for a wide range of cloud systems. In contrast, the contents of the sub-models may depend on specific technologies and hence may differ from system to system. For example, the contents of the Data sub-model may depend on whether structured, semi-structured, or non-structured data are used; the contents of the Infrastructure sub-model will be different depending on whether virtual machines or containers are used etc. Therefore in the following we show examples of what the contents of the sub-models may be. Still, we try to be generic enough so that these models likely apply to many different cloud systems with no or little modification and can be used as starting point for modeling other cloud systems as well.

The Data sub-model shown in Fig. 3 is based on a relational model (like in [Ri13]) but abstracts from details that are not important for us (e.g., domains of attributes) and adds others that are important (e.g., relating to the storage of data). The smallest unit of data is the "Attribute value," corresponding to a cell in a relational table. Attributes (columns) and Records (rows) contain multiple Attribute values; a Data set (table) contains multiple Attributes and multiple Records. A Data set can either be stored or transferred, represented by the entities Stored data set and Data flow inheriting from Data set. A database consists of multiple stored data sets. A Stored data set can be stored either in a local Database associated with a specific application Component or using a database management system (DBMS) provided by the Middleware. Attribute Values, Attributes, Records and Data Sets are all considered "Data objects." For each Data object there can be multiple Data protection requirements. Data objects may be accessed by different Actors in Data-specific roles. Encryption can be used to alter the security attributes of a Data object.

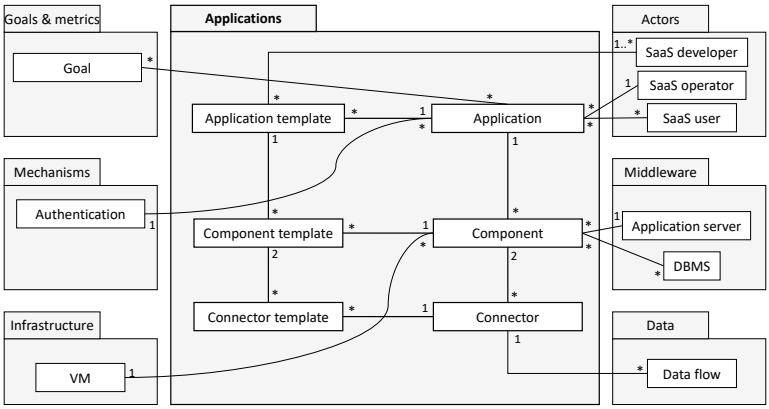


Fig. 4: The sub-model Applications and its relations to the other sub-models

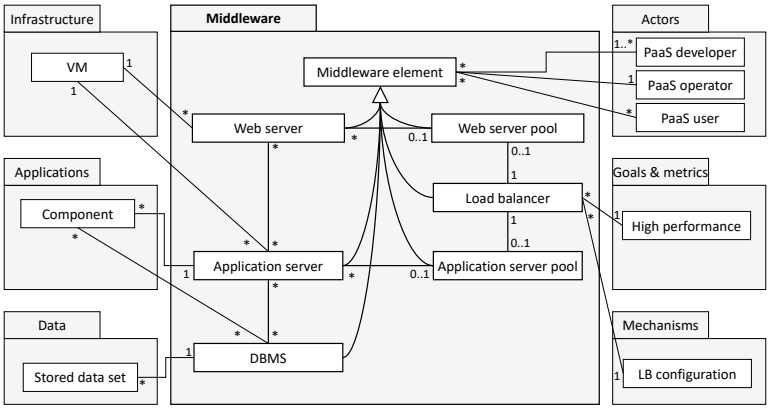


Fig. 5: The sub-model Middleware and its relations to the other sub-models

Applications (Fig. 4) consist of multiple Components that are linked by Connectors. The logical structure of an Application is defined by an Application template, from which the specific Application, Component, and Connector instances are derived and scaled as necessary. SaaS (Software as a Service) developers work with Application templates. Applications are managed by SaaS operators and used by SaaS users. Certain Goals may apply to an Application and Authentication mechanisms can be turned on or off for an Application. Each Component is deployed in a VM (Virtual Machine) and controlled by an Application server. A Connector may accommodate Data flows.

The Middleware sub-model (Fig. 5) supports multi-tier web applications with Web server, Application server, and DBMS entities. Web servers and Application servers can be clustered into Web server pools and Application server pools, respectively, which can be associated to a Load balancer. All these are Middleware elements, which are created, operated, and used

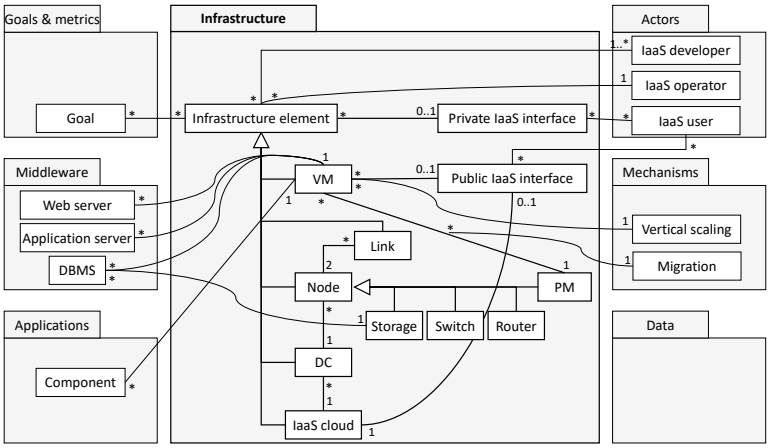


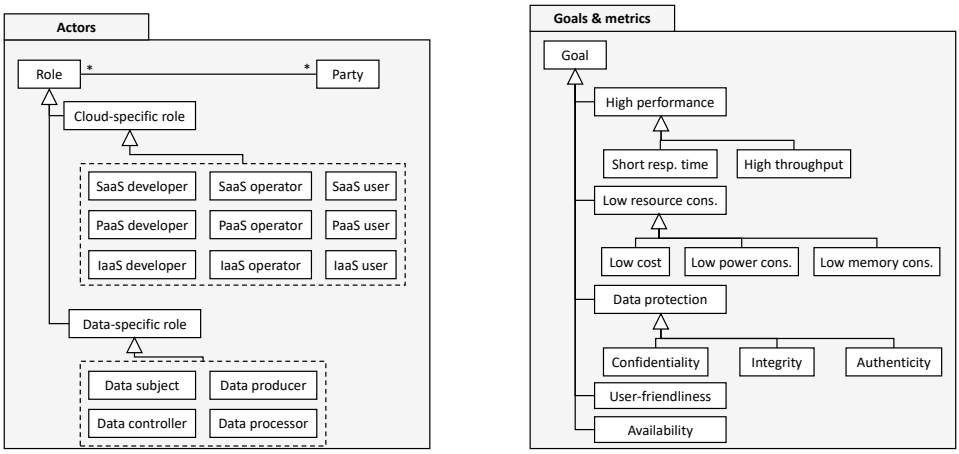
Fig. 6: The sub-model Infrastructure and its relations to the other sub-models

by PaaS (Platform as a Service) developers, PaaS operators, and PaaS users, respectively. Web servers and Application servers are hosted by VMs, Application servers manage application Components, and DBMS store data sets. A Load balancer has an associated Performance goal and can be configured by an appropriate configuration mechanism.

In the sub-model Infrastructure (Fig. 6), an IaaS (Infrastructure as a Service) cloud consists of multiple data centers (DC); each DC consists of multiple network nodes, connected by links. Nodes include storage, switches, routers, and physical machines (PM); each PM may host multiple virtual machines (VM). IaaS clouds, DCs, Nodes, Links, and VMs are considered Infrastructure elements. An IaaS cloud can be accessed through a Public or Private cloud interface. The former allows access to VMs only, while the latter allows access to all Infrastructure elements. Different Goals may apply to Infrastructure elements. Application Components as well as Web and Application servers are hosted on VMs. An IaaS user can access VMs by using a Public IaaS interface, whereas IaaS developers and operators have access to any Infrastructure element. VMs can be scaled using Vertical scaling and their mapping to PMs can be changed by Migration.

To improve visibility, reduce redundancy, and save space, for the following sub-models we do not show their relations to the other sub-models. In the sub-model Actors (Fig. 7a), each Party can have multiple cloud-specific or data-specific Roles. As Cloud-specific roles we differentiate users, developers and operators on SaaS, PaaS, or IaaS level. Within Data-specific roles, we differentiate Data subjects, Data producers, Data processors, and Data controllers, as stipulated by the GDPR. Cloud-specific roles may relate to different Application, Middleware, and Infrastructure elements. Similarly, Data-specific roles relate to Data objects. A Party in a specific Role may have multiple Goals.

The sub-model Goals & metrics (Fig. 7b) contains the non-functional requirements that



(a) (b)

Fig. 7: The sub-models (a) Actors and (b) Goals & metrics. The dashed boxes indicate that all contained classes inherit from the same superclass

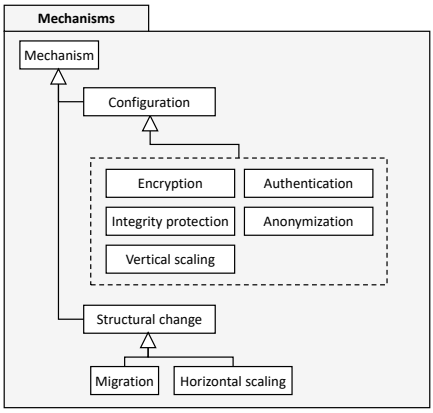


Fig. 8: The sub-model Mechanisms

the system must fulfill. This includes High performance, Low resource consumption, Data protection, User-friendliness, and Availability. These high-level goals can be decomposed into more specific goals; in particular, Data protection is decomposed into Confidentiality, Integrity, and Authenticity. Goals may relate to different Assets. In particular, Data protection goals relate to Data objects. A Goal is posed by a Party in a specific Role and may be impacted by different Mechanisms.

As shown in Fig. 8, Mechanisms can be of two kinds: Configuration or Structural change. Configuration may mean that a feature such as Encryption within an application Component

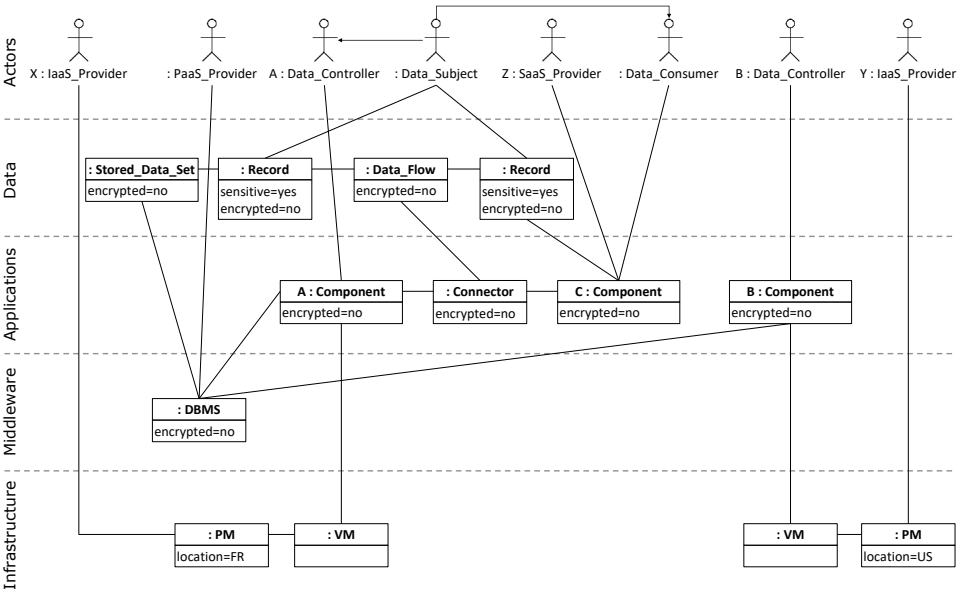


Fig. 9: Object model resulting from applying the meta-model of Sec. 5 to the example of Sec. 2

is switched on or off or some more subtle parameter change, e.g., the key length of the encryption algorithm is changed. Similar mechanisms are Authentication, Integrity protection (e.g., using cryptographic hashes), Anonymization of data, or Vertical scaling of VMs. Structural changes, on the other hand, create or remove entities or change interconnections among entities. In particular, Horizontal scaling of VMs creates or removes VM instances, whereas Migration of VMs between PMs changes the interconnection between the affected PMs and VM. These are powerful mechanisms to achieve several different Goals, including Data protection (e.g., by means of co-locating Components so that the Data flow between them does not have to traverse the network).

6 Application to the industrial cloud scenario

To validate the applicability of the proposed meta-model, we revisit the industrial cloud scenario from Sec. 2. We modeled that example using the proposed meta-model. An excerpt of the resulting model is shown in Fig. 9, consisting of the four Assets and the Actors sub-models and their interconnections.

The resulting model captures all details that are necessary to automatically identify all data protection issues that were previously identified by experts in Fig. 1. For example, Fig. 9 shows that the data subject trusts data controller A and the data consumer, but all other

parties are not trusted. We can see the different actors that can access components A, B, and C, the data record containing sensitive data of the data subject, the joint use of the DBMS, as well as the used infrastructure elements and the associated providers. On the basis of this model, the problems marked in Fig. 1 can be identified using the run-time model (instance of the meta-model). The run-time model is shown as an object model in Fig. 9. The model shows that the PaaS provider has access to the DBMS, which contains a sensitive data record of a data subject in an unencrypted form, and the PaaS provider is not trusted by the data subject. This clearly constitutes a high risk of data protection violation. Such violations can be found automatically based on a pre-defined set of forbidden subgraphs using graph pattern matching algorithms [Ma11], as discussed in [SMM17].

Once a data protection violation has been detected, the model may also be used to investigate the available mechanisms. For each of them, it can be checked whether (i) it is applicable in the given situation, (ii) it would break the forbidden subgraph found in the run-time model, and (iii) it would not introduce another forbidden subgraph. In our example, turning on encryption would be such a mechanism. If multiple appropriate mechanisms are available, the one with the best (i.e., least disadvantageous) impact on the other goals is selected and applied, thus automatically solving the identified problem. If no appropriate adaptation action can be found, a human operator needs to be alerted. A more detailed discussion on the process of applying the run-time model for identifying data protection violations can be found in the companion paper [SMM17].

7 Conclusions and future work

In this paper, we argued that data protection in a dynamic cloud setting should be addressed by automatically configuring the system at deployment time and dynamically re-configuring it at run time. A central element of such an approach is a run-time model of the relevant entities. We have presented a way of determining and structuring the corresponding meta-model. Similarly to previous approaches to cloud modeling, cloud assets like infrastructure, middleware, and applications need to be modeled. However, for data protection purposes, more is needed, and hence we introduced further sub-models for data, actors, goals, and mechanisms. We have presented an initial validation of our meta-model by applying it to an industrial case study and found that the resulting model contains all necessary information for detecting and automatically mitigating data protection violations.

The next step will be to implement the proposed model together with the reasoning technique for finding risk patterns described in [SMM17]. This can then be used to carry out a more realistic evaluation of the proposed approach, also showing how much the meta-models of the run-time model of different cloud systems differ from each other. Moreover, it should be investigated how other, more sophisticated mechanisms to protect privacy, such as controlled interaction [BMZ16] can be incorporated.

Acknowledgments. This work received funding from the European Union's Horizon 2020 research and innovation programme under grant 731678 (RestAssured). Useful discussions with project partners are gratefully acknowledged.

References

[Ah17] Ahmadian, Amir Shayan; Strüber, Daniel; Riediger, Volker; Jürjens, Jan: Model-Based Privacy Analysis in Industrial Ecosystems. In: Proceedings of the 13th European Conference on Modelling Foundations and Applications. pp. 215–231, 2017.

[Am12] Amoui, Mehdi; Derakhshanmanesh, Mahdi; Ebert, Jürgen; Tahvildari, Ladan: Achieving dynamic adaptation via management and interpretation of runtime models. Journal of Systems and Software, 85(12):2720–2737, 2012.

[Ar12] Ardagna, Danilo; Di Nitto, Elisabetta; Casale, Giuliano; Petcu, Dana; Mohagheghi, Parastoo; Mosser, Sébastien; Matthews, Peter; Gericke, Anke; Ballagny, Cyril; D'Andria, Francesco et al.: MODAClouds: A model-driven approach for the design and execution of applications on multiple clouds. In: Proceedings of the 4th International Workshop on Modeling in Software Engineering. IEEE Press, pp. 50–56, 2012.

[Be14] Bergmayr, Alexander; Troya, Javier; Neubauer, Patrick; Wimmer, Manuel; Kappel, Gerti: UML-based Cloud Application Modeling with Libraries, Profiles, and Templates. In: CloudMDE@MoDELS. pp. 56–65, 2014.

[Be16] Bergmayr, Alexander; Breitenbücher, Uwe; Kopp, Oliver; Wimmer, Manuel; Kappel, Gerti; Leymann, Frank: From Architecture Modeling to Application Provisioning for the Cloud by Combining UML and TOSCA. In: Proceedings of the 6th International Conference on Cloud Computing and Services Science. pp. 97–108, 2016.

[BKW14] Busch, Marianne; Koch, Nora; Wirsing, Martin: SecEval: An Evaluation Framework for Engineering Secure Systems. In: Modellierung. pp. 337–352, 2014.

[BMZ16] Biskup, Joachim; Menzel, Ralf; Zarouali, Jaouad: Controlled Management of Confidentiality-Preserving Relational Interactions. In: International Workshop on Data Privacy Management. pp. 61–77, 2016.

[Ch12] Chapman, Clovis; Emmerich, Wolfgang; Márquez, Fermin Galán; Clayman, Stuart; Galis, Alex: Software Architecture Definition for On-demand Cloud Provisioning. Cluster Computing, 15(2):79–100, 2012.

[Co16] Council of the European Union: , General Data Protection Regulation. `http://data.consilium.europa.eu/doc/document/ST-5419-2016-INIT/en/pdf`, 2016.

[He16] Heinrich, Robert: Architectural run-time models for performance and privacy analysis in dynamic cloud applications. ACM SIGMETRICS Performance Evaluation Review, 43(4):13–22, 2016.

[KM16] Kritikos, Kyriakos; Massonet, Philippe: An integrated meta-model for cloud application security modelling. Procedia Computer Science, 97:84–93, 2016.

[LAL17] Lejeune, Jonathan; Alvares, Frederico; Ledoux, Thomas: Towards a generic autonomic model to manage Cloud Services. In: 7th International Conference on Cloud Computing and Services Science. 2017.

[Ma11] Mann, Zoltán Ádám: Optimization in computer engineering – Theory and applications. Scientific Research Publishing, 2011.

[Ma14] Marquezan, Clarissa Cassales; Wessling, Florian; Metzger, Andreas; Pohl, Klaus; Woods, Chris; Wallbom, Karl: Towards exploiting the full adaptation potential of cloud applications. In: Proceedings of the 6th International Workshop on Principles of Engineering Service-Oriented and Cloud Systems. pp. 48–57, 2014.

[Ma15a] Mann, Zoltán Ádám: Approximability of virtual machine allocation: much harder than bin packing. In: Proceedings of the 9th Hungarian-Japanese Symposium on Discrete Mathematics and Its Applications. pp. 21–30, 2015.

[Ma15b] Mann, Zoltán Ádám: Modeling the virtual machine allocation problem. In: Proceedings of the International Conference on Mathematical Methods, Mathematical Models and Simulation in Science and Engineering. pp. 102–106, 2015.

[MM17] Mann, Zoltán Ádám; Metzger, Andreas: Optimized Cloud Deployment of Multi-tenant Software Considering Data Protection Concerns. In: 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. pp. 609–618, 2017.

[Mo13] Modi, Chirag; Patel, Dhiren; Borisaniya, Bhavesh; Patel, Avi; Rajarajan, Muttukrishnan: A survey on security issues and solutions at different layers of Cloud computing. The Journal of Supercomputing, 63(2):561–592, 2013.

[Na11] National Institute of Standards and Technology: , NIST Cloud Computing Reference Architecture. NIST Special Publication 500-292, https://www.nist.gov/publications/nist-cloud-computing-reference-architecture, 2011.

[Na12] Nagel, Benjamin; Gerth, Christian; Yigitbas, Enes; Christ, Fabian; Engels, Gregor: Model-driven specification of adaptive cloud-based systems. In: Proceedings of the 1st International Workshop on Model-Driven Engineering for High Performance and Cloud computing. 2012. article 4.

[OA13] OASIS: , Topology and Orchestration Specification for Cloud Applications (TOSCA) v1.0. http://docs.oasis-open.org/tosca/TOSCA/v1.0/os/TOSCA-v1.0-os.html, 2013.

[Ri13] Ristić, Sonja; Aleksić, Slavica; Čeliković, Milan; Luković, Ivan: An EMF Ecore based relational DB schema meta-model. In: Proceedings of the 6th International Conference on Information Technology ICIT. 2013.

[Sh10] Shao, Jin; Wei, Hao; Wang, Qianxiang; Mei, Hong: A runtime model based monitoring approach for cloud. In: IEEE 3rd International Conference on Cloud Computing. pp. 313–320, 2010.

[SMM17] Schoenen, Stefan; Mann, Zoltán Ádám; Metzger, Andreas: Using Risk Patterns to Identify Violations of Data Protection Policies in Cloud Services. In: 13th International Workshop on Engineering Service-Oriented Applications and Cloud Services. 2017.

[SMP15a] Schmieders, Eric; Metzger, Andreas; Pohl, Klaus: Architectural runtime models for privacy checks of cloud applications. In: Proceedings of the Seventh International Workshop on Principles of Engineering Service-Oriented and Cloud Systems. pp. 17–23, 2015.

[SMP15b] Schmieders, Eric; Metzger, Andreas; Pohl, Klaus: Runtime model-based privacy checks of big data cloud services. In: International Conference on Service-Oriented Computing. pp. 71–86, 2015.