Towards Auditing of Cloud Provider Chains using CloudTrust Protocol

Thomas Rübsamen, Dirk Hölscher and Christoph Reich

Institute for Cloud Computing and IT Security, Furtwangen University, Robert-Gerwig-Platz 1, Furtwangen, Germany

Keywords: Cloud Computing, Audit, Federated Cloud, Security, Digital Evidence.

Abstract: Although cloud computing can be considered mainstream today, there is still a lack of trust in cloud providers, when it comes to the processing of private or sensitive data. This lack of trust is rooted in the lack of transparency of the provider's data handling practices, security controls and their technical infrastructures. This problem worsens when cloud services are not only provisioned by a single cloud provider, but a combination of several independent providers. The main contributions of this paper are: we propose an approach to automated auditing of cloud provider chains with the goal of providing evidence-based assurance about the correct handling of data according to pre-defined policies. We also introduce the concepts of individual and delegated audits, discuss policy distribution and applicability aspects and propose a lifecycle model. Our previous work on automated cloud auditing and Cloud Security Alliance's (CSA) CloudTrust Protocol form the basis for the proposed system for provider chain auditing.

1 INTRODUCTION

An important problem that is commonly associated with the use of cloud services is the loss of control about who is processing data, how it is used and whether or not it is shared with third parties. A possible approach to mitigate this problem is to provide additional information to stakeholders (cloud users, cloud auditors). That information can be obtained as a part of regular cloud audits where evidentiary information about data processing is collected and compared against agreed-upon policies (such as terms of service or privacy policies). This way, what happens in the cloud becomes more transparent to the user, which could lead to improved trust in the cloud by providing additional information on data processing in a cloud service. Therefore, a system for automated audits is needed that provides meaningful evidence and shows how and where data is processed.

With lacking transparency comes low trust (Knode, 2009). Low trust in the cloud (especially in the security and privacy of data) hinders the growth of cloud computing as a business (Cloud Security Alliance, 2013). Customers are less likely willing to move their business applications into the cloud when they have no chance evaluating, whether or not their data is processed according to company policies. Transparency is not just showing cloud consumers how and where data is processed, it is also important to know by whom the consumer's data is processed. However, transparently showing by whom the data might be processed is not common practice today. Furthermore, cloud providers can incorporate services provided by Nth-level providers into their own, which makes it even harder for the auditor to follow where the consumer's data is currently being stored.

It makes sense for Software as a Service (SaaS) providers to utilize a third-party Infrastructure as a Service (IaaS) provider for service hosting. We consider such cloud provider chains to become increasingly important to look at, when a complete picture about data processing in a particular service should be provided. Most importantly, the compliance with data processing and privacy policies of all involved parties needs to be assessed. For instance, in some situations a cloud provider might be forced to transfer consumer data from one cloud provider to another, or from one geographic location to another for load-balancing or cost optimization. In such a case, the new provider has to ensure that all policies are covered just as much as the others in the chain have to.

Cloud auditing is becoming increasingly important for certification (e.g., FedRAMP (FedRAMP, 2015), ISO27001 (ISO, 2013) with the proposed ISO27017 (ISO, 2015) and ISO27018 (ISO, 2014)). However, the required audit is still largely a manual process. In this paper, we propose an extension to our previous work see (Rübsamen and Reich, 2013; Rübsamen et al., 2015)) on automated, evidence-

Rübsamen, T., Hölscher, D. and Reich, C.
Towards Auditing of Cloud Provider Chains using CloudTrust Protocol.
In *Proceedings of the 6th International Conference on Cloud Computing and Services Science (CLOSER 2016) - Volume 1*, pages 83-94
ISBN: 978-989-758-182-3

based cloud auditing, that provides improved transparency to cloud stakeholders about data processing in the cloud. The extension introduces the concept of cloud provider chains, data processing and privacy policies with an extended scope on all involved providers, and audit evidence exchange. The main contribution of this paper introduces the concept adapting CSA CloudTrust Protocol (CTP) (Cloud Security Alliance, 2015) for the use in inter-provider exchange of evidence during cloud audits.

Using the introduced CTP extension, additionally generated evidence (i.e., information not specified in the original CTP) will be used to enhance CTP reports. Furthermore the extension enables auditing of third-party contractors within provider chains to show that each statement made by the provider, with respect to the users established policies (e.g., data location, service availability), is fulfilled.

This paper is organized as follows: In the next Section 2 related work is presented. Section 3 introduces the concept of service provision chains and discusses the scopes and applicability of policies. In Section 4 two approaches towards cloud auditing are introduced. After that, we propose a lifecycle model for delegated auditing of cloud provider chains in Section 5. Next, we focus on data exchange patterns (Section 6) that are used in Section 7, where we present our proposed system. Following that, section 8 evaluates the presented results using a scenario description and a discussion of the threat model. In Section 9 we conclude this paper.

2 RELATED WORK

Security and privacy auditing are increasingly important topics in cloud auditing. They demonstrate that security controls are put in place by the provider and also that they are functioning correctly (i.e., data protection mechanisms are working correctly and effectively). There are some projects working on the architectural and interface level regarding the automation of security audits, such as the Security Audit as a Service (SAaaS) project (Doelitzscher et al., 2012; Doelitzscher et al., 2013). SAaaS is specifically designed to detect incidents in the cloud and thereby consider the dynamic nature of such ecosystems, where resources are rapidly provisioned and removed. However, SAaaS does not address provider chain setups or treat gathered data as evidence.

ABTiCI (Agent-Based Trust in Cloud Infrastructure) describes a system used for monitoring (Saleh, 2014). All relevant parts of a cloud infrastructure are monitored to be able to detect and verify unauthorized access. Integrity checks are done at boot-time, using Trusted Platform Module (TPM) boot or at runtime. Monitoring hardware and software configurations allow the system to detect changes at runtime. The aforementioned system is similar to our approach. Instead of using agents we utilize CTP. Furthermore, our approach relies on evidence collection through audits with pull and trigger mechanisms.

A centralized trust model is introduced by Rizvi et al. (Rizvi et al., 2014). Trust between consumer and provider is established by using an independent third-party auditor. With the adoption of a third-party auditing system, consumers are able to create baseline evaluation for providers they have never worked with to generate initial trust. The model acts as a feedback mechanism providing valuable insight into the providers processes. After initial trust was generated the third-party auditor continues to obtain trust values for the consumer. We see initial trust in the provider as a given factor and focus on obtaining trust values based on evidence within a multi-provider scenario.

A completely different approach is proposed by Gonzales et al., where the authors introduce an architecture for the measurement of integrity and confidentiality of a cloud provider (Gonzales et al., 2015). Their approach is based on best practices and security metrics. It uses trust zones to delineate resources (physical, logical or virtual) within multi-tenant IaaS infrastructures. Such a zone is used to separate interests. Sensitive business data is located in a Gold Zone, non-business partners are located in a less privileged zone and can't access the Gold Zone. The focus in this work lies in the separation of concerns. Trust is generated using best practices and security metrics. There is no provider auditing involved, but everything is estimated based on metric values. Whereas, in our approach metrics can be used to collect additional information but the focus lies in evidence collection.

The DMTF is also working on cloud auditing with the Cloud Audit Data Federation (CADF) working group. They focus on developing standardized interfaces and data formats to enable cloud security auditing (Distributed Management Task Force, Inc. (DMTF), 2014). A similar project is the Cloud Security Alliance's Cloud Trust Protocol (CTP), which defines an interface for enabling cloud users to "generate confidence that everything that is claimed to be happening in the cloud is indeed happening as described, ..., and nothing else" (Cloud Security Alliance, 2015), which indicates an additional focus on providing additional transparency of cloud services. The latter two projects, however, do not elaborate on how the interfaces should be implemented nor do they describe explicitly focus on privacy and accountability. We use CTP as a basis and propose its extension and use in our proposed auditing system to enable automated auditing of cloud provider chains.

There are approaches that deal with checking compliance with data location policies. The principle of location transparency of data in the cloud (i.e., a user does not know in which server, data center or even country a specific data object is stored) is contrary to data locality requirements some cloud consumers have to fulfill (e.g., a legal obligation to ensure a certain geographic storage location). Massonet et al. propose a system that exposes infrastructure-level location monitoring information to the cloud consumer (Massonet et al., 2011). We use data location as an use case for the demonstration of our system.

A crucial part of cloud auditing is the collection of data on which an audit can be based upon. That data can be produced on all architectural layers of the cloud (e.g., on the bare-metal, in a VM, in a subsystem). Several approaches to addressing the unique requirements of cloud logging have been proposed. For instance, Marty presents a logging framework and guidelines for IaaS and SaaS logging (Marty, 2011). We have also previously discussed the different sources of data that can be used as evidence during audits. (Rübsamen and Reich, 2013). We demonstrate, how such data can be collected and more importantly used for auditing in a multi-provider scenario.

The CloudTrust Protocol (Cloud Security Alliance, 2015) establishes a mechanism that allows users to audit a CSP. An auditor can choose from a set of transparency elements for instance: geographic location of data objects and affirmation or results of latest vulnerability assessments. CTP has 23 pre-defined transparency elements and supports user-specified elements on which cloud consumer and provider agreed on. The purpose of the CTP is to transparently provide the user with important information about the cloud to show that processing is done as promised. By providing information about the inner-workings of the cloud service (with respect to the transparency elements), trust between the cloud provider and the consumer is supposed to be strengthened. If a consumer can trust his provider he is more likely willing to move sensible business processes into the cloud.

There are two main problems the protocol tries to solve:

- Restoration of control and freedom of choice at the cloud consumer by enabling him to specifically request information on configurations, vulnerabilities and data integrity.
- The provision of a standardized process, which enables providers to generate and expose additional information with respect to the transparency elements.

CTP needs to be beneficial for both cloud providers and consumers. Providers won't invest into structural changes of their services if the expected payoff is small. For this reason the protocol can be adjusted to the trust needs of consumers as well as operational circumstances of the provider. Only the request/response process and the associated data formats are specified, whereas there are no additional restrictions put on the actual implementation of the information gathering process.

Communication handling in CTP is done by two managers. The auditor is using CTP's *Request Manager* whereas the provider is using the *Response Manager*. These two architectural components are responsible for communication, tracking pending requests, CTP translation into service specific API calls and data conversion into CTP format. The data format for reporting is based on XML (used in an extended form in the proposed system) for the 2.0 version, respectively JSON for the currently proposed version 3.0 of CTP. In general CTP's protocol design follows the RESTful paradigm.

3 CLOUD SERVICE PROVISION CHAINS

In this chapter, we describe complex cloud service provision scenarios. We thereby focus on use cases where multiple cloud service providers are involved in the provision of a single cloud service (as perceived by the cloud consumer). In these use cases, the agreement on data processing and privacy policies, that apply on the whole service provider chain, can quickly become a difficult problem. Therefore, the auditing of compliance with such policies along the chain, both increases in complexity and difficulty.

In the following discussion of the different scopes of policy applicability, we assume the following definitions:

Cloud Consumer
We use the definition provided by NIST where a cloud consumer is "a person or organization that maintains a business relationship with, and uses service from, Cloud Providers" (Liu et al., 2011).

Cloud Service Provider
We use to the definition provided by NIST where a cloud provider is "a person, organization, or entity responsible for making a service available to interested parties" (Liu et al., 2011). Furthermore, we

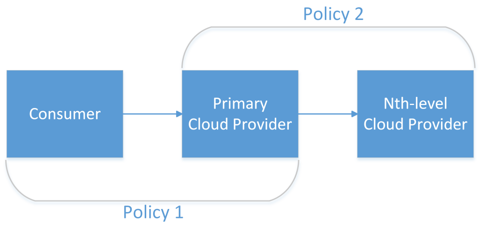

Figure 1: Audit Policy Scopes.

define the provider facing the consumer as the primary provider and each succeeding provider who interacts with the primary provider is defined as a Nth-level provider.

Cloud Service Provider Chain
A Cloud Service Provider Chain is characterized by at least two cloud service providers being part of providing a service by composing their individually offered services.

In the following, we describe three different scopes of policy applicability as depicted in Figure 1. In that scenario we assume a cloud service (provided by a primary cloud provider) that is provided to a service consumer, while utilizing an additional third-party service (provided by the sub-provider on the Nth level).

Scope A: Cloud Consumer / Primary Cloud Provider. In a typical cloud use case, a consumer uses the services provided by a single cloud service provider to accomplish a given task. The details of the service usage are governed by terms of service agreements, privacy polices etc. In this most common scenario, the cloud consumer and the cloud provider agree on these terms before any service is provisioned. Typically, this happens during a registration or contract agreement phase. With respect to data flow between the consumer and the provider, this means that data processing is performed by the cloud provider in compliance with the agreed-upon policies (see *Policy 1* in Figure 1). Personal data that is disclosed by the cloud consumer to the cloud provider as part of regular service use is processed by the cloud provider according to the limits defined in the policy.

Scope B: Primary Cloud Provider / Nth-level Cloud Provider. Similar to the approach described in Scope A, there may be similar agreements between cloud providers. For instance, the primary cloud provider may require resources from the sub-provider, e.g., to extend its own service offering, to address peak loads in service usage or to outsource internal processes such as backups. In this case, the primary cloud provider (as depicted in Figure 1) becomes a cloud consumer itself. The integration of cloud services provided by a sub-provider in cloud services provided by the primary cloud provider is governed by a contractual agreement between the two providers (see *Policy 2* in Figure 1).

Scope C: Cloud Consumer / Nth-level Cloud Provider. In case of a cloud scenario, where multiple service providers are involved in the provisioning of a single service, the cloud consumer may not necessarily be aware of this. Since the cloud consumer has only contact with its immediate provider (primary cloud provider in Figure 1), he might not necessarily be aware, that the primary cloud provider is using an additional external service. A typical example for such a scenario is a SaaS provider hosting its services on resources provided by an infrastructure provider, or a SaaS provider that integrates another SaaS provider's service for data processing. Additionally, a silent change of the supplementary service provider can be imagined, when the primary provider switches to another service (e.g., uses another infrastructure provider for cost efficiency reasons). In this case, the restrictions that governs the policy agreement between the cloud consumer and the primary cloud provider (i.e., Policy 1) must also apply to the sub-provider, if data owned by the cloud consumer is transferred between them. This is the case, when either: i) similar policies policy rules exist in Policy 1 and 2, where the rules defined in Policy 2 are at least as strict as equivalent rules defined in Policy 1 (in this case, a matching of whether or not rules from policy 1 and 2 are compatible needs to be performed), or ii) the downstream provider accepts rules from policy 1 directly.

4 AUDITING CLOUD PROVIDER CHAINS

In this chapter, we illustrate different variations of auditing cloud provider chains. We thereby focus on traditional individual audits and delegated provider audits. Furthermore, we present several information delivery patterns.

4.1 Individual Provider Audits

Figure 2 describes the process of auditing individual providers in a service provision chain. All policies will be distributed to each provider (as seen in Figure 2). Policy distribution can either be:

1. Manual policy evaluation: This approach is based on the specified policy documents (e.g., terms

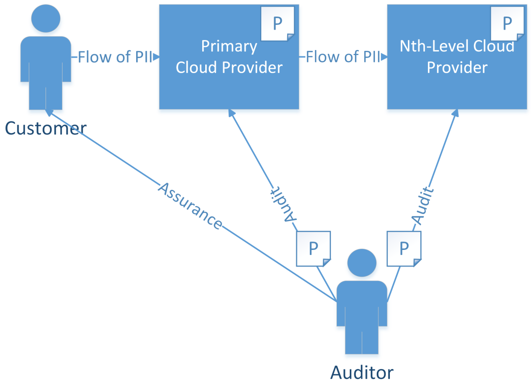

Figure 2: Individual Audit.

of service in human-readable form) given by the provider. The auditor manually maps statements of such documents to information requests for the providers (e.g., asking for specific process documentation or monitoring data and audit logs).

2. Deploying machine-readable policies: In this approach the auditor deploys a machine-readable policy document (XML, JSON) onto the provider. The provider will then automatically audit the tasks specified within the document. The auditor can request the results for the audited policy rules to verify if everything is fulfilled. The policy needs to be deployed to each involved provider. Within this approach new policies can easily be added and deployed for automated auditing.

The audit results are used to assure the consumer that policy and rule compliance is given or not. As previously described, a service provision chain contains at least one provider. In this case, two providers - a primary and a 2nd-level provider. To audit the service as a whole, it is necessary to audit each provider separately and then aggregate the results to form a complete picture of the service from an audit perspective. This means, that regarding data handling policies (e.g., location restrictions, access control etc.), each provider that holds data is audited. The same is true for the auditing of security and privacy controls that are put in place at the providers. Obviously, the consumer-facing provider has to transparently disclose all his sub-providers and notify auditors about every sub-provider his data was stored at and where his data is currently stored. Even though not every provider will get the consumer's data, the auditing process gains more complexity with an increasing number of Nth-level providers. Requests must be sent to each provider separately and each provider will deliver audit reports to the auditor.

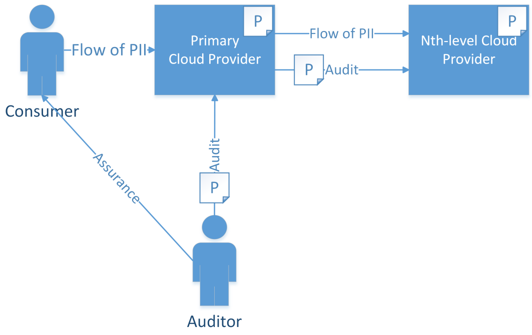

Figure 3: Delegated Audit.

4.2 Delegated Provider Audits

An alternative to individual audits are what we call delegated audits, where the auditor only interfaces with the primary service provider that in turn takes over the auditing of its sub-provider(s). Therefore the auditor only has to audit the primary service provider to obtain policy compliance results of all involved service providers. This allows less influential stakeholders such as the cloud consumer to act as an auditor towards the primary provider while not having the same rights towards the Nth-level provider(s). Whereas the individual audit scenario is an example of how chain audits could be performed with more influential stakeholders, such as data protection authorities. Figure 3 depicts the delegated provider audit scenario. Every audit request is sent to the primary provider who will then extract CTP calls from a previously deployed policy document (machine-readable document deployed by the auditor). Since the primary provider is acting as a mediator he has to delegate requests and communication. Existing problems regarding policy compliance is of major concern for the primary provider because complaints will always be addressed to him, even if he is not responsible for a failed audit. For the case a given audit response did not satisfy policy compliance the consumer will contact the primary provider with a complaint (e.g, data was transferred outside valid location). On the other hand the consumer's payoff can be much higher due to the centralized structure using a mediator ensuing low complexity for the auditor. Therefore, he can always rely on the data controller to forward his request to the data holding sub-provider without the need of adaption (send requests to different entities, use different API-calls).

5 AUDIT LIFECYCLE IN DELEGATED PROVIDER AUDITS

In the following section the audit system lifecycle is described. Figure 4 illustrates the three phases: i) Preparation, ii) Processing and iii) Presentation. In the following, we describe each phase in more detail:

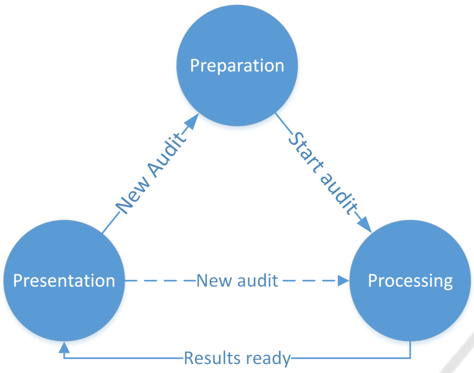

Figure 4: System Lifecycle.

5.1 Preparation Phase

The first phase of the lifecycle is the Preparation Phase in which the system is prepared. The most important task during the Preparation Phase is resource identifier distribution, which is required for request handling.

Request handling is done using unique resource identifiers (URI), which are used to identify any kind of resource that is part of an audit. A URI unambiguously identifies an object within a provider's domain. In our approach, each provider has its own namespace in which identifiers can be assigned arbitrarily.

The preparation process *Policy adding* allows the auditor to create new rules based on already existing policies. For instance, he can specify a new data location rule to ensure that his data will not leave his countries jurisdiction. Newly added rules are written into a machine-readable audit policy that describes evidence that is to be collected, and checks that are to be performed during the audit. From the new rule, auditable elements are derived, that an automated audit process provides all necessary information to enable the possibility of policy compliance assessing. Auditable elements include for example the location of data, logs and configurations.

The *Policy mapping* process, maps each new added rule or policy to transparency elements and associated requests. If a newly added rule cannot be mapped to an already existing transparency element a new element needs to be created. The mapping is done based on the specified policies. For this reason the policy adding process is limited to already defined policies and the associated rules within the contract. During the mapping each non-standard policy (i.e., a policy that requires a transparency element that is not part of CTP) will receive an URI and all necessary data sources needed to answer a request. The mapping process generates URIs and defines all auditable attributes for an element.

The preparation process *Policy distribution* propagates the resource identifiers throughout the system. Each sub-provider sends his resource identifiers to the primary provider. Afterwards, when all identifiers are known by the primary provider, he will forward them to the auditor.

5.2 Processing Phase

With the end of the Preparation Phase the second lifecycle phase starts. In Processing Phase all elements will be collected. For instance, all essential information for the inquired elements are retrieved from the evidence store and written into a CTP response. A policy evaluation is done to determine the policy compliance. All information, collected for one element are written into a response and sent back to the requesting entity.

5.3 Presentation

The last phase in the lifecycle is the Presentation Phase. Within this phase the auditor will be presented with the audit results. Thereby, each requested element will be presented to him containing the name of the policy rule as well as its achieved compliance state.

The lifecycle is complete, when the results were presented to the auditor. After this the lifecycle can continue with Preparation Phase again. Returning to the Preparation Phase is necessary if new policies/rules were added or in a continuous auditing scenario, where policy compliance is audited in short intervals or event-driven (e.g., on new or changed policy, on infrastructure change or on custom triggers defined by the auditor). During the new cycle only newly generated URIs will be distributed.

6 AUDIT INFORMATION EXCHANGE

In this section two information exchange patterns are described that are used for different purposes in our proposed system. Obtaining information as well as providing information can be a difficult task depending on how timely data is needed. Therefore, we specify a pull pattern (see Section 6.1) for non-critical information (e.g., evidence used during infrequent audits) and a push pattern (see Section 6.2) for critical information (evidence used during continuous audits).

6.1 Data Requests (Pull)

Within a pull scenario audit results (transparency elements request results) are only delivered, if there is an existing pull request for an element. Figure 5 shows the individual data request sequence. The shown figure does not imply the use of a pull mechanism, but can be used with one. For each pending request its corresponding data is pulled form a database. Pulled information will run through a compliance check afterwards, to verify if a requested policy with its corresponding rule is fulfilled. This means, that audit data request results are only delivered to the auditor if there is an existing request. When a request for transparency elements arrives the data is pulled from a database (evidence store) and will then go through a validity check to ascertain if a policy is fulfilled or not. The auditor will not necessarily receive critical information in a timely manner, unless he requested the data during the incidents occurrence, which is highly unlikely. Also, choosing the right request interval can quickly become a scalability issue. Polling for new data in relatively short intervals can introduce load problems at the auditing system. This issue is a common problem with pull / polling mechanisms. However, reasonable and usually quite common interval choices such as hourly, daily, weekly and monthly reports do not introduce such problems.

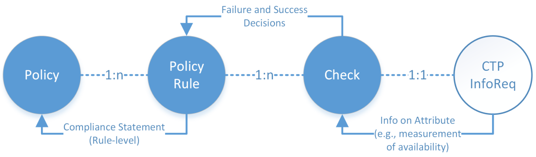

Figure 5: Individual Audit Data Requests.

6.2 Triggered Notification (Push)

Critical information that quickly needs to be processed (e.g., forwarded to an analysis tool or presented to an auditor) like security breaches or integrity violations are time-critical and therefore cannot rely on transport via pull mechanism. Immediate notifications are necessary to avoid data control deprivation. Push mechanisms promise a more reactive and rapid way of transporting critical data. Push mechanisms are typically associated with an event-driven approach, where an event is fired when a condition is met. Such an event could be the occurrence of a data relocation. Such an event needs to be audited if there are policy rules limiting allowed geographical locations. Figure 6 shows the push notification process. An auditor can specify the severity of an occurrence called trigger, to ensure that only significant information are pushed. A configured trigger will only trigger when the condition set by the auditor is fulfilled. After the occurrence an incident notification will be pushed to the auditor. There is no need to define a trigger for all possible auditable elements. Therefore, it is not feasible to send a notification for every small change in the system. If a breach occurred or a vulnerability was found during the audit, a notification is pushed and countermeasures can be taken faster which will immensely reduce reaction time.

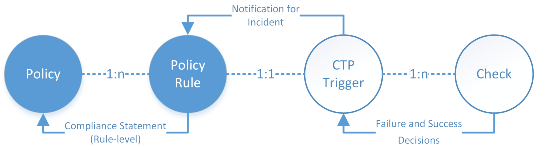

Figure 6: Triggered Audit Data Exchange.

7 EXTENDING CLOUDTRUST PROTOCOL FOR PROVIDER CHAIN AUDITING

In our approach, we leverage CTP as a means for evidence exchange between cloud providers in complex cloud auditing scenarios. Additional functions and components are located above the protocol (as seen in Figure 7) and are responsible to exchange requests and responses with the CTP. This structure enables us to utilize the benefits of CTP out of the protocol's operational area without changing the protocol itself. Although the operational structure of CTP remains unchanged some optimisations for audit reports are required to be able to transfer additional information e.g. more detailed user access lists. In this case, the additional information would give the auditor not only authorized users but also since when they have authorization and who authorized user permissions.

Figure 7 illustrates the systems architecture in a two provider scenario. Within the figure it is assumed, that the Preparation Phase did end and all for the audit request necessary policies and rules were already mapped and distributed. Incoming transparency element requests will directly go to the Remote Evidence Collection component. In the following paragraph the system components of our proposed approach are described:

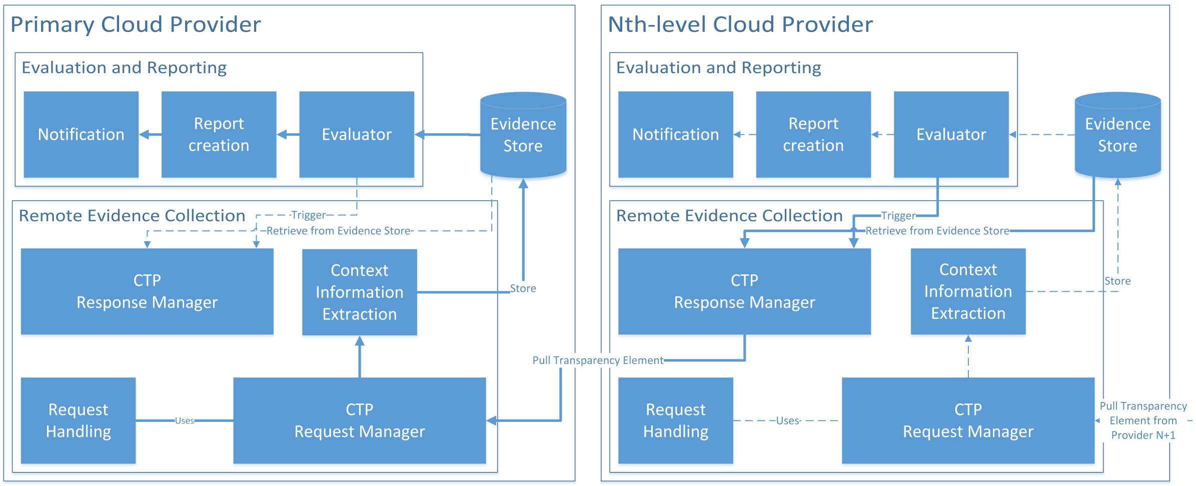

Figure 7: Multi-Provider Audit System Architecture.

Remote Evidence Collection

- **Request Handling:** Every incoming audit request will arrive at the Request handler of the primary provider. A decision is made which resource identifier should be used based on current data location (Nth-level provider, primary provider). The resource identifiers are used to set up CTP-calls. Therefore, it will forward each request to the CTP Request Manager. Each request is processed separately to guarantee that context information or states do not get mixed up.
- **CTP Request Manager:** The Request Manager, sends each given request to the CTP Response Manager of a Nth-level provider(solid line in Figure 7 between both providers) using a pull pattern (see Section 6.1). Inter-provider communication is initiated by the Request Manager.
- **Context Information Extraction:** An incoming CTP-response contains the general response (specified in (Cloud Security Alliance, 2015)) as well as the corresponding context information and the compliance state for the requested element. The context information are extracted from the response and securely stored inside the evidence store. The remaining information which are used for report creation are stored as well for the audit report creation.
- **CTP Response Manager:** After receiving a request the Response Manger pulls data from the evidence store if the Nth-level provider is not able to determine the compliance state of his own or receives the data from the Evaluator. Obtained results are packed into a CTP-response and sent back to the primary providers CTP Request Manager. In case a trigger is fired the Response Manager will push the response immediately to the primary providers CTP Request Manager even in the absence of an audit request for the triggered element. A primary provider might be a Nth-level provider of another provider and thus needs a Response Manager. Requests for context information are send from the auditor to the primary providers Response Manager. Like a normal response the Manager pulls the context information for the requested object from the evidence store and writes them into a CTP-Response.

Evidence Store: The Evidence Store is a database containing all audit results (including context information) for the primary and its Nth-level providers. Each participating provider has its own evidence store where his achieved audit results are stored and can be pulled from by pending requests. The main purpose of the evidence store is to provide audit evidence and to make them accessible to the auditor. If no data was relocated to a Nth-level provider, a response will be generated from audit entries for the primary provider located in the evidence store.

Evaluation and Reporting

- **Evaluator:** The Evaluator runs policy compliance checks on all obtained results used for report creation. Achieved results can get one of three possible compliance states depending on the level of fulfillment:
  - **State 1 Successful:** The results obtained from the database fulfills the policy.
  - **State 2 Partially:** A policy is partially fulfilled.
  - **State 3 Failed:** No records for this policy were found or the given results were unsatisfying.

Configured triggers (see Section 6.2) are fired if a compliance check for a request failed or a deviation from the trigger specification is identified.

- **Report Creation:** The stored content (state, CTP transparency elements results) is used to create the final report. The report can be of different types, for instance a representation of the results on a web dashboard or in a auto-generated document.
- **Notification:** An audit can take some time to finish. This largely depends on the size and scope of the audit. Therefore, asynchronous mechanisms are required to present audit results. An auditor can be notified via mail when his audit is finished and his audit report is available.

Implementation of each above described part is mandatory for every provider. It is possible that a primary provider is a Nth-level provider in a different audit-chain, whereas a Nth-level provider might be a primary provider in another audit chain.

8 EVALUATION

In the following Section, we evaluate our proposed approach towards auditing cloud provider chains. We split the evaluation in two parts: i) a functional evaluation using a fictitious cloud scenario with two providers involved in the service provision, and ii) a security analysis of the proposed approach.

8.1 Functional Analysis by Scenario

For the functional look at our proposed solution, we assume the following scenario (as depicted in Figure 1):

- Cloud provider 1 (CSP1) is a SaaS provider (and primary cloud provider) that hosts on the virtual resources provided by CSP2.
- Cloud provider 2 (CSP2) is an IaaS provider (and Nth-level cloud provider) with a data center in Germany and in Russia.
- A Cloud consumer (CC) uses the service provided by CSP1.
- CC and CSP1 agree that CC's data must not leave Germany.
- The auditor checks the compliance with the data location requirement on behalf of CC.

In this case the consumer may believe that his data is located within CSP1's datacenter in Germany. Due to CSP1 not having actual computing resources by its own, the data is actually located in CSP2's data centers, either in Germany, Russia or both. However, CSP1 is still obligated to adhere to the data locality restriction. CSP2 enables CSP1 to audit policy compliance by offering access to our tools for automated auditing. CSP1 establishes regular audits and evidence collection that is focused on data location. The auditor now audits the provider chain with CSP1 as a starting point. CSP1 also runs our audit tool. Its main user is the auditor acting on behalf of the consumer. The communication between the audit tools is implemented using CTP as described in Section 7. Both providers use the audit tool to collect information required for the audits. In the following, the request / response process for this scenario is described:

1. CSP1 is forwarding the audit to CSP2, requesting audit results regarding data location if the data is pulled or waiting till the data was moved which would cause the trigger to fire. For most elements it may be sufficient to list their state in a report but there are elements where immediate notification is indispensable. Such elements require CTPs trigger mechanism as described in Sections 6.2 and 7. In this case the data location request does not need a trigger.
2. At a later point, after the trigger has fired or the information was pulled, the response arrives at CSP1's Request Manager.
3. After receiving the response, CSP1 begins to extract all context information from the response and stores it in the evidence store. This step is necessary to ensure that the audit trail remains available and protected for either archival purposes (e.g., required by law) or re-use at a later point in time to claim remediation. This way, an auditor that feels the need to further investigate a statement made by any of the providers, can retrieve stored evidence from the evidence store.
4. The remaining information (state, CTP response) are used to create the final audit report. The final report for the element contains the policies name as well as the compliance state. The policy compliance check showed that the policy is partially fulfilled. Such an outcome would mean, that the data left Germany at one time.
5. Now the auditor has the possibility to request context information. To access the evidence store CTP is used and each request requesting context information is sent directly to the Response Manager. The Evidence Store will then create a standardized report containing context information, about where the data is currently being stored (city, state, data center) and where it had been (country code), and send it back to the auditor.

With the additional information the auditor can validate how severe the policy breach was. Therefore, network efficiency in multi-tenant environments is required to satisfy each tenants expectations.

8.2 Threat Model

It is important to consider the security of the proposed system to achieve confidence in the acquired audit results. Therefore we perform a security analysis of our proposed approach to cloud provider chain auditing. We follow a simple methodology of defining threat scenarios, categorizing them using the STRIDE (Microsoft Developer Network, 2014) threat model and proposing mitigation strategies for each of the identified threats. The mitigation of the threat categories will be discussed in more detail in section 8.3. STRIDE categorizes threats as follows:

- **S**poofing Identity
- **T**ampering with Data
- **R**epudiation
- **I**nformation disclosure
- **D**enial of Service
- **E**levation of Privilege

We have identified the following major threats to the evidence transfer and processing in multi-provider audits:

- *Unauthorized access (S,I)*: Using our system exposes valuable information such as internal logging, infrastructure design etc. to external entities in an automated way. A malicious external user may steal or otherwise illegitimately gain access to the API that is used for data exchange between the providers. While there is no direct access to consumer data provided by our system, transferred information usually contains metadata about a consumer's system/data properties. The given responses has the potential to expose potentially sensitive metadata. Such information may include but is not limited to data regarding configuration, access control lists and installed software from which vulnerabilities and attack vectors can be deferred. Another potential adversary is a malicious insider at a cloud provider. He can potentially gain access evidence data by directly attacking the evidence store or by intercepting communication between the system components on the internal network.
- *Data leakage (I)*: Audit trail data may intentionally or unintentionally become available. By collecting audit trails from the various evidence sources into the evidence store, a new data source becomes available. Security mechanisms of the evidence store may fail, which could lead to data leakage.
- *Eavesdropping, (I)*: A malicious external user may try to eavesdrop on audit information while it is being transmitted either to the auditor (audit result including audit trails), between cloud providers (information on transparency elements) or internally at a cloud provider (raw data flowing between evidence source and evidence store).
- *Denial of Service (D)*: Denial of Service attacks have unfortunately become a very common type of attack against networked computer systems, that's in many cases trivial to carry out. External adversaries attack either the system directly by exploiting flaws in the implementation or by generating bogus load with the goal of shutting the service down completely.
- *Audit trail manipulation (T,R,I)*: The data generated by our system is supposed to be used during automated audits. The results of these audits should be dependable and believable. An adversary may manipulate audit trail data at various points in the system. For instance, a malicious insider may manipulate the results that are returned by the API. Preserving the integrity of the audit data is therefore of utmost importance.

8.3 Security Analysis by Scenario

Some of the aforementioned threats can be mitigated by implementing appropriate security controls. In the following we present, how the threat categories are addressed in our system:

In order to mitigate the risk of spoofed identities and unauthorized access, we use strong authentication mechanisms based on user and system identification using certificates. This way simple brute force attacks against our publicly available API in order to guess access credentials can be prevented.

Risks of information disclosure and tampering/manipulation (i.e., data integrity) are typically addressed by introducing data encryption and hashing schemes. We encrypt data at rest (e.g., while it is stored in the evidence store) and in transit (e.g., while it is transmitted between providers and/or the auditor or between system components). To have a full protection against data leakage, ideally there is also encryption of data during processing. While not the focus of this paper, more details on the data at rest and in transit encryption as well as the integrity protection implementation in our system can be found in (Rübsamen et al., 2015). However, this currently severely limits the usefulness, processing options and processing performance (Lopez et al., 2014).

Denial of Service risks can only be addressed by considering the software and the environment it runs in. Directed attacks on the application level can be mitigated using application level firewalls, code audits and security checks, whereas network-based attacks typically require the capability to filter malicious traffic upstream.

9 CONCLUSIONS

In this paper we introduced a system which allows automated auditing of provider chains. We discussed two different types of chain audits: i) individual provider audits where the auditor has to audit each Nth-level provider separately and ii) delegated provider audits where the primary cloud provider acts as an mediator. Our proposed system focuses on the latter approach. We consider the main goal of our cloud audit system to strengthen trust and transparency in cloud services. This could lead to an even better adoption of cloud computing. We also discussed the applicability of data handling and privacy policies and how they apply in complex scenarios where multiple providers share a cloud consumer's data. In the latter part of this paper, we focused on the architectural integration of the CloudTrust Protocol in the evidence collection and transport of our audit system. Finally, we concluded this paper with an evaluation of our proposed approach. We evaluated the functional soundness by demonstrating an audit scenario that involves a cloud consumer using a service that is intransparently provided by two different cloud providers. Additionally, we evaluated our approach by defining a threat model using threat scenarios and addressing those threats.

In our future work, we focus on even more complex service provision scenarios, where even more layers of service providers are involved. We will also put special focus on ensuring the scalability of our approach. Another interesting topic emerges, when any of the cloud providers is considered untrustworthy. This can be the case when a malicious insider tries to intrude in our system. We consider ensuring the integrity of evidence in the whole chain of providers to be a major challenge.

ACKNOWLEDGEMENTS

This work has been partly funded from the European Commission's Seventh Framework Programme (FP7/2007-2013), grant agreement 317550, Cloud Accountability Project - http://www.a4cloud.eu/ - (A4CLOUD).

REFERENCES

Cloud Security Alliance (2013). The notorious nine - cloud computing top threats in 2013. https://downloads.cloudsecurityalliance.org/initiatives /top_threats/The_Notorious_Nine_Cloud_Computing_ Top_Threats_in_2013.pdf.

Cloud Security Alliance (2015). Cloud Trust Protocol. https://cloudsecurityalliance.org/research/ctp.

Distributed Management Task Force, Inc. (DMTF) (2014). Cloud auditing data federation (cadf) - data format and interface definitions specification. http://www.dmtf.org/sites/default/files/standards/docu ments/DSP0262_1.0.0.pdf.

Doelitzscher, F., Reich, C., Knahl, M., Passfall, A., and Clarke, N. (2012). An Agent Based Business Aware Incident Detection System for Cloud Environments. *Journal of Cloud Computing: Advances, Systems and Applications*, 1(1):9.

Doelitzscher, F., Rübsamen, T., Karbe, T., Reich, C., and Clarke, N. (2013). Sun behind clouds - on automatic cloud security audits and a cloud audit policy language. *International Journal On Advances in Networks and Services*, 6(1 & 2).

FedRAMP (2015). Federal Risk and Authorization Program. http://www.fedramp.gov.

Gonzales, D., Kaplan, J., Saltzman, E., Winkelman, Z., and Woods, D. (2015). Cloud-trust - a security assessment model for infrastructure as a service (iaas) clouds. *Cloud Computing, IEEE Transactions on*, PP(99):1–1.

ISO (2013). ISO27001:2013 - Information technology – Security techniques – Information security management systems – Requirements. http://www.iso.org/iso/catalogue_detail?csnumber=54 534.

ISO (2014). ISO/IEC FDIS 27018 - Information technology – Security techniques – Code of practice for protection of personally identifiable information (PII) in public clouds acting as PII processors. http://www.iso.org/iso/catalogue_detail.htm?csnum ber=61498.

ISO (2015). ISO/IEC FDIS 27017 - Information technology – Security techniques – Code of practice for information security controls based on ISO/IEC 27002 for cloud services. http://www.iso.org/iso/catalogue_detail?csnumber=43 757.

Knode, R. (2009). Digital trust in the cloud. http://assets1.csc.com/cloud/downloads/Digital_Trust_ in_the_Cloud.pdf.

Liu, F., Tong, J., Mao, J., Bohn, R., Messina, J., Badger, L., and Leaf, D. (2011). Nist cloud computing reference architecture. http://www.nist.gov/customcf/get_pdf.cfm?pub_id=909505.

Lopez, J., Rübsamen, T., and Westhoff, D. (2014). Privacy-friendly cloud audits with somewhat homomorphic and searchable encryption. In *Innovations for Community Services (I4CS), 2014 14th International Conference on*, pages 95–103.

Marty, R. (2011). Cloud application logging for forensics. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, SAC '11, pages 178–184, New York, NY, USA. ACM.

Massonet, P., Naqvi, S., Ponsard, C., Latanicki, J., Rochwerger, B., and Villari, M. (2011). A monitoring and audit logging architecture for data location compliance in federated cloud infrastructures. In *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on*, pages 1510–1517.

Microsoft Developer Network (2014). The Stride Threat Model. https://msdn.microsoft.com/en-US/library/ee823878(v=cs.20).aspx.

Rizvi, S., Ryoo, J., Liu, Y., Zazworsky, D., and Cappeta, A. (2014). A centralized trust model approach for cloud computing. In *Wireless and Optical Communication Conference (WOCC), 2014 23rd*, pages 1–6.

Rübsamen, T., Pulls, T., and Reich, C. (2015). Secure Evidence Collection and Storage for Cloud Accountability Audits. In *CLOSER 2015 - Proceedings of the 5th International Conference on Cloud Computing and Services Science, Lisbon, Portugal, May 20 - 22, 2015*. to appear.

Rübsamen, T. and Reich, C. (2013). Supporting cloud accountability by collecting evidence using audit agents. In *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, volume 1, pages 185–190.

Saleh, M. (2014). Construction of agent-based trust in cloud infrastructure. In *Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference on*, pages 941–946.