Security Prospects through Cloud Computing by Adopting Multiple Clouds

Meiko Jensen, Jörg Schwenk
Chair for Network and Data Security
Horst Görtz Institute for IT-Security
Ruhr-University Bochum, Germany
{*meiko.jensen, joerg.schwenk*}@*rub.de*

Jens-Matthias Bohli, Nils Gruschka
NEC Laboratories Europe
Heidelberg, Germany
{*bohli, gruschka*}@*neclab.eu*

Luigi Lo Iacono
European University
of Applied Sciences
Brühl, Germany
l.lo_iacono@eufh.de

***Abstract*—Clouds impose new security challenges, which are amongst the biggest obstacles when considering the usage of cloud services. This triggered a lot of research activities in this direction, resulting in a quantity of proposals targeting the various security threats.**

Besides the security issues coming with the cloud paradigm, it can also provide a new set of unique features which open the path towards novel security approaches, techniques and architectures. This paper initiates this discussion by contributing a concept which achieves security merits by making use of multiple distinct clouds at the same time.

***Keywords*-Cloud; Security; Intercloud; Application Partitioning; Tier Partitioning; Multi-party Computation**

I. Introduction

Cloud computing offers dynamically scalable resources provisioned as a service over the Internet. The third-party, on-demand, pay-per-use and seamlessly scalable computing resources and services offered by the cloud paradigm promise to reduce capital as well as operational expenditures for hardware and software. Recent figures published by the pioneering cloud service providers show that this has been recognized and partially already adopted by cloud users [1]. At the end of the fourth quarter of 2009, 102 Billion objects have been stored in Amazon's Simple Storage Service (S3) [5]. At the end of the fourth quarter of 2010, the number of objects stored in S3 grew by 257% to 262 Billion. Thus, the cloud is a successful business model, and it is foreseen to remain important in the future [2]. As one consequence of this success, the number of cloud service providers offering cloud services increased so that cloud users now have a rich set of services to choose from. In the following some prominent examples will be named while further introducing cloud foundations.

One way of categorizing clouds takes the physical location from the viewpoint of the user into account [3]. A *Public Cloud* is offered by third-party service providers and involves resources outside the user's premises. In case the cloud system is installed on the user's premise—usually in the own data center—this setup is called *Private Cloud*. A hybrid approach is denoted as *Hybrid Cloud*. This paper will concentrate on Public Clouds, since these services demand for the highest security requirements but also—as this paper will start arguing—includes high potential for security prospects.

Another categorization depends on the type of resources or services delivered by the cloud and distinguishes three distinct layers [3]. *Infrastructure-as-a-Service* (IaaS) is the name for cloud environments that provide their users with basic infrastructure facilities including CPU, memory, and storage instances. These infrastructure components are operated and maintained by the IaaS provider. The most prominent examples of this type of cloud services are Amazon's Elastic Compute Cloud (EC2) [4], the aforementioned Amazon's Simple Storage Service (S3) [5], Savvis Symphony [6], and RackSpace Cloud [7].

Platform-as-a-Service (PaaS) describes a platform delivery model. Here, the cloud provider offers specific runtime environments to be used in the user's own application contexts. Examples would be providing database services or specific application runtime environments. On top of these platforms, the cloud user is able to implement and operate own applications. Hence, the PaaS provider is responsible for providing the hardware and the particular platform (including update management and bug fixing), and the cloud user is responsible for the specific implementations that use the given platform APIs. Examples for PaaS offerings are Google's App Engine [8] for Web application development, Microsoft SQL Azure [9] for databases, and Cloudscale [10] for real-time data analysis tasks.

Software-as-a-Service (SaaS) refers to the approach of providing a full software application most commonly via browser-based techniques. Here, the cloud provider is responsible for all parts of the application stack: hardware, operating system, application runtime, and the software implementation itself. The cloud users in this scenario are humans that interact with the cloud services via browser interfaces. Popular examples include Salesforce for a Customer Relationship Management (CRM) system [11] and the provisioning of office suites by Google [12] and Zoho [13].

All of the three layers share the commonality that the end-

978-0-7695-4460-1/11 $26.00 © 2011 IEEE
DOI 10.1109/CLOUD.2011.85

users' digital assets are taken from an intra-organizational to an inter-organizational context. This creates a number of issues, amongst which security aspects are regarded as the most critical factors when considering cloud computing adoption [14].

II. Cloud Security Issues and Challenges

Cloud computing creates a large number of security issues and challenges. A list of security threats to cloud computing is presented in [15]. These issues range from the required trust in the cloud provider and attacks on cloud interfaces to misusing the cloud services for attacks on other systems.

A. Scope of Cloud Security

The main problem that the cloud computing paradigm implicitly contains is that of secure outsourcing of sensitive as well as business-critical data and processes. When considering using a cloud service, the user must be aware of the fact that all data given to the cloud provider leaves the own control and protection sphere. Even more, if deploying data-processing applications to the cloud (via IaaS or PaaS), a cloud provider gains full control on these processes. Hence, a strong trust relationship between the cloud provider and the cloud user is considered a general prerequisite in cloud computing.

Depending on the political context this trust may touch legal obligations. For instance, Italian legislation requires that data of Italian citizens remains within Italy. Thus, using a cloud provider from outside of Italy for realizing a service provided to Italian customers would immediately violate this obligation. Hence, the cloud users must trust the cloud provider hosting their data within the borders of the country and never copying them to an off-country location (not even for backup) nor providing access to the data to entities from abroad.

B. Attacks on Cloud Security

Even though in the majority of the cases it may be legitimate to assume a cloud provider to be honest and handling the customers' affairs in a respectful and responsible manner, still there remains a risk of the own cloud system getting compromised by third parties. In [16] an overview of such kind of attacks are given. Some examples and more recent advances are briefly discussed in the following.

Ristenpart et al. [17] presented some attack techniques for the virtualization of the Amazon EC2 IaaS service. Their approach is the following: the attacker allocates new virtual machines until one runs on the same physical machine as the victim's machine. Then, the attacker can perform cross-VM side-channel attacks to learn or modify the victim's data. The authors present strategies to reach the desired victim machine with a high probability. Finally, they propose the usage of blinding techniques to fend cross-VM side-channel attacks.

In [18] a flaw in the management interface of Amazon's EC2 was found. The SOAP-based interface uses XML Signature as defined in WS-Security for integrity protection and authenticity verification. The authors of [18] discovered that the EC2 implementation for signature verification is vulnerable to the *Signature Wrapping Attack* [19]. In this attack, the attacker—who eavesdropped a legitimate request message—can add a second arbitrary operation to the message while keeping the original signature. Due to the flaw in the EC2 framework, the modification of the message is not detected and the injected operation is executed on behalf of the legitimate user and billed to the victim's account.

A major incident in a SaaS cloud happened in 2009 with Google Docs [20]. Google Docs allows users to edit documents (e.g. text, spreadsheet, presentation) on-line and share these documents with other users. However, this system had the following flaw: once a document was shared with anyone it was accessible for everyone the document owner has ever shared documents with before. For this technical glitch not even any criminal intent was required to get unauthorized access to confidential data.

Recent attacks introduced in [21] demonstrate that cloud systems of major cloud providers have shown severe security flaws in different types of clouds.

C. The Threat of Compromised Clouds

As can be seen from the related work on attacks on cloud systems, the cloud computing paradigm contains an implicit threat of working in a compromised cloud system. If an attacker is able to infiltrate the cloud system itself, all data and all processes of all users operating on that cloud system may become subject to malicious actions in an avalanche manner. Hence, the cloud computing paradigm requires an in-depth reconsideration on what security requirements might be affected by such an exploitation incident. For the common case of a single cloud provider hosting and processing all of its user's data, an intrusion would immediately affect all security requirements: accessibility, integrity, and confidentiality of data and processes may become violated, and further malicious actions may be performed on behalf of the cloud user's identity.

Following, a novel concept is introduced which enables the cloud user to construct more secure and dependable cloud deployments whose security guarantees hold even in the presence of malicious or compromised clouds.

III. Cloud Security Prospects

Technical cloud security mechanisms have limitations. There is e.g. no feasible way to perform any kind of data-dependent operation in cloud systems without breaking integrity and confidentiality. As a consequence, the establishment and maintenance of a strong trust relationship between a cloud user and a cloud service provider is still indispensable.

This paper suggests a novel approach which makes use of essential cloud properties to reduce the dependency of the trust relationship. The basic underlying idea is to use multiple distinct clouds at the same time in order to mitigate the risks of malicious data manipulation, disclosure, and process tampering. By integrating distinct clouds the trust assumption can be lowered to an assumption of non-collaborating cloud service providers.

The idea of making use of multiple clouds has already been proposed [22], [23]. The scope of this previous work is however not on security. It focuses on the federation of different clouds in order to distribute computing load to overcome resource restrictions of single particular clouds. To the knowledge of the authors, this is the first attempt to introduce and discuss the use of multiple clouds to construct security prospects out of cloud properties.

The introduced concept includes three distinct architectural patterns.

- **Replication of Applications** allows to receive multiple results from one operation performed in distinct clouds and to compare them. This enables the user to get an evidence on the integrity of the result (see Section III-A).
- **Partition of Application System into Tiers** allows to separate the logic from the data (see Section III-B). This gives additional protection against data leakage through flaws in the application logic.
- **Partition of Application Tiers into Fragments** allows distributing fine-grained fragments of the logic as well as the data to distinct clouds. None of the involved cloud providers possess a view on all the data and on all of the processing logic which provides a safeguard of the data's and application's confidentiality (see Section III-C).

Each of the introduced patterns provides an unique security merit. By combining them, the security merits are combined as well. The following subsections present the three patterns in more detail and investigate their merits and flaws with respect to the stated security requirements under the assumption of one or more compromised cloud systems.

A. Replication of Application

How does a cloud customer know whether his data was processed correctly within the cloud? There is no technical way to guarantee that an operation performed in a cloud system was not tampered with or that the cloud system was not compromised by an attacker. The only kind of guarantee is based on the level of trust between the cloud customer and the cloud provider and on the contractual regulations made between them such as SLAs, applicable laws and regulations of the involved jurisdictional domains. But even if the relation and agreements are perfectly respected by all participants, there still remains a residual risk of getting compromised by third parties.

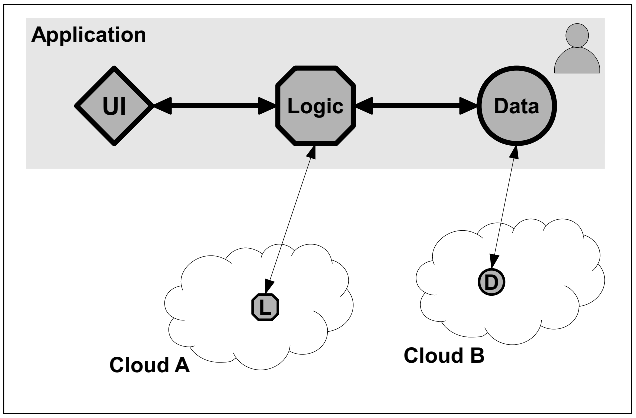

Figure 1. Replication of Application Systems

In order to solve this intrinsic problem, the use of multiple distinct clouds is proposed (see Figure 1). Instead of executing a particular application on one specific cloud, the same operation is executed by distinct clouds. By comparing the obtained results, the cloud user gets evidence on the integrity of the result. In such a setting, the required trust towards the cloud service provider can be lowered dramatically. Instead of trusting one cloud service provider totally, the cloud user only needs to rely on the assumption, that the cloud providers do not collaborate maliciously against herself.

Assume that $n > 1$ clouds are available (like e.g. Cloud A, B, and C in Figure 1). All of the n adopted clouds perform the same task. Assume further that f denotes the number of malicious clouds and that $n - f > f$ the majority of the clouds are honest. The correct result can then be obtained by the cloud user by comparing the results and taking the majority as the correct one. There are other methods of deriving the correct result, for instance using the TurpinCoan algorithm [24] for solving the General Byzantine Agreement problem.

Instead of having the cloud user performing the verification task, another viable approach consists in having one cloud monitoring the execution of the other clouds. For instance, Cloud A may announce intermediate results of its computations to a associated monitoring process running at Cloud B. This way, Cloud B can verify that Cloud A makes progress and sticks to the computation intended by the cloud user. As an extension of this approach, Cloud B may run a model checker service that verifies the execution path taken by Cloud A on-the-fly, allowing for immediate detection of irregularities.

This architecture enables to verify the integrity of results obtained from tasks deployed to the cloud. On the other hand it needs to be noted that it does not provide any protection in respect to the confidentiality of data or processes. On the contrary, this approach might have a negative impact on the confidentiality since—due to the deployment of multiple clouds—the risk rises that one of them is malicious or compromised. To implement protection against an unauthorized access to data and logic this architecture needs to be combined with the architecture described in Section III-C.

The idea of resource replication can be found in many other disciplines. In the design of dependable systems for example it is used to increase the robustness of the system especially against system failures [25]. In economic business processes—and especially in the management of supply chains—single-source suppliers are avoided in order to lower the dependency on suppliers and increase the flexibility of the business process [26]. In all these cases the additional overhead introduced by doing things multiple times is accepted in favor of other goals resulting from this replication.

This architectural concept can be applied to all three cloud layers. A case study at the SaaS-layer is discussed in Section IV-A.

B. Partition of Application System into Tiers

The architectural pattern described in the previous Section III-A enables the cloud user to get some evidence on the integrity of the computations performed on the third-party resources or services.

The architecture introduced in this section targets the risk of undesired data leakage. It gives an answer to the following question: how can a cloud user be sure, that data access is implemented and henceforth enforced effectively and that errors in the application logic do not affect the user's data?

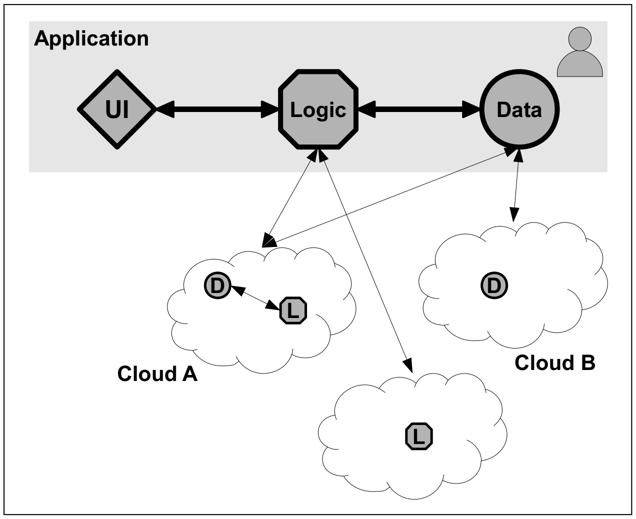

Figure 2. Partition of Application System into Tiers

To limit the risk of undesired data leakage due to application logic flaws, the separation of the application system's tiers and their delegation to distinct clouds is proposed (see Figure 2). In case of an application failure the data is not immediately at risk since it is physically separated and protected by an independent access control scheme. Moreover the cloud user has the choice to select a particular—probably specially trusted—cloud provider for data storage services and a different cloud provider for applications.

It needs to be noted, that the security services provided by this architecture can only be fully exploited if the execution of the application logic on the data is performed on the cloud user's system. Only in this case, the application provider does not learn anything on the users' data. Thus, the SaaS-based delivery of an application to the user side in conjunction with the controlled access to the user's data performed from the same user's system is the most far-reaching instantiation.

Beside the introduced overhead due to the additionally involved cloud, this architecture requires moreover standardized interfaces in order to couple applications with data services provided by distinct parties. Also generic data services might serve for a wide range of applications there will be the need for application specific services as well.

This architectural concept can be applied to all three cloud layers. A case study at the SaaS-layer is discussed in Section IV-B.

C. Partition of Application Tiers into Fragments

The partitioning of application systems into tiers and distributing the tiers to distinct clouds provides some coarse-grained protection against data leakage in the presence of flaws in application design or implementation.

The next architecture variant targets the disclosure of processing logic and data. It gives an answer to the following question: how can a cloud user avoid the full revealing of processing logic and data to the cloud provider?

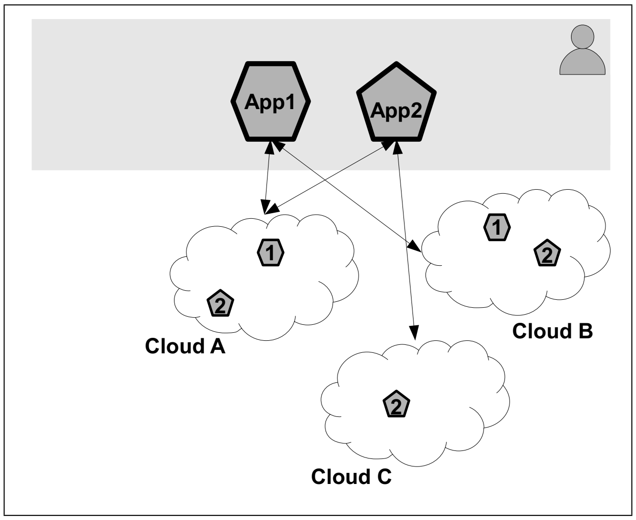

Figure 3. Partition of Application Tiers into Fragments

The idea is, that the application logic and data needs to be partitioned into fine-grained parts and to distribute these parts to distinct clouds (see Figure 2). This approach can be instantiated in different ways depending on how the partitioning is performed.

1) Obfuscating Splitting: By this approach, data and/or application parts are distributed to different clouds in such a way, that every single cloud gains only a limited knowledge and only the final result or the combined data at the user's side must be classified as confidential. For *application splitting*, a first approach is using the existing sequential or parallel logic separation. Thus, depending on the application, every cloud provider just performs subtasks on a subset of data (see below for examples). A typical way of *data splitting* is pseudonymization: one provider receives the data with some key fields (typical personal identification data like name, address etc.) replaced by a random identifier and the second provider receives the mapping of the identifier to the original information. This approach is for example used in a commercial cloud security gateway [27].

The problem with the idea of *obfuscating splitting* is the fact that there is no general pattern for realization. Careful analysis of the splitted data and application must be performed regarding its confidentiality, i.e. checking if the information a single cloud provider receives really is "useless".

2) Multi-party Computation: With multi-party computation a number of participants can compute functions on their input values without revealing any information on their individual inputs. This approach considers a multi-party computation between several clouds. Two distinct scenarios can be imagined: an application that intrinsically requires multi-party computation is outsourced to the multi-party cloud, or a single cloud user make use of a multi-party cloud for better protection of the secrecy of his data.

The idea of secure multi-party computation was first presented in [28] as a solution to the millionaires problem: Two millionaires want to find out who is richer without disclosing any further information about their wealth. Very closely related problems do indeed appear in today's business environment. One scenario is that multiple corporations want to do a statistical analysis of their business and market data. The result is expected to help all of them, however, for obvious reasons no corporation wants to disclose their data to each other. If the stakeholders cannot identify a single third party trusted by all, this scenario requires multi-party computation. Thus, when outsourcing this computation to a cloud, there will not be a single cloud that is trusted by all participants. However, if every participant finds their own cloud they trust, e.g. because some corporations have their data already outsourced to a cloud, a multi-party computation between these clouds will be possible.

Besides transferring problems that require in any case a multi-party computation into the cloud, using multiple cloud services doing a multi-party computation can also be beneficial for protecting the secrecy of data of a single user. Multi-party computation in this case will work as follows: the user will compute shares of his data using a secret sharing scheme such as Shamir's [29] and distribute the shares to the different clouds. The clouds will jointly compute the function of interest on these shares, communicating with each other when necessary. In the end, the clouds hold shares of the result which is sent back to the user who can reconstruct the result. In this case, using multiple cloud computation guarantees the secrecy of the input data, unless the cloud providers collude to open shares. Assuming that the cloud provider itself is not malicious, but might be compromised by attacks or have single malicious employees, this collusion is hard to establish so that a good protection is given.

A multi-party computation between clouds makes it possible to compute a function on data in a way that no cloud provider learns anything about the input or output data. As secret sharing rather than encryption is used, a collusion of all clouds would be able to reconstruct the secrets.

For applications that are already solved by multi-party computation, no further overhead is introduced by outsourcing this computation to several clouds. While addition and multiplications come with a small overhead, more complex operations have a significant overhead. It is ongoing research to reduce the overhead by multi-party computation and recent improvements, e.g. on equality and comparison of values, has lead to frameworks which can already be considered practical [30], [31].

IV. Case Studies

To illustrate the proposed approach, this section presents and discusses scenarios that are based on realistic case studies to show the advancements and conditions that must be considered for using the multi-cloud security approach.

A. Replicating of Application Tasks

Imagine a cloud provider named InstantReporting that provides the service of creating annual accounting reports automatically out of a given set of business data. This is a very typical scenario of cloud usage, since such a report has to be published by all commercial entities once a year. Hence, the resources required to create such reports are only necessary for a small period of time every year. Thus, by using a third-party cloud service for this, in-house resources can be omitted which would run idle most of the year. On the other side, by sharing its service capabilities among a large set of companies—all of which have to create their reports at different times of the year—a cloud service provider gains large benefits from providing such a shared service "on the cloud".

However, as promising as this scenario seems to be in terms of using the cloud computing paradigm, it contains a fundamental flaw: the cloud customers can not verify that the annual report created by the cloud service is correct. There might have been accidental or intentional modifications of the source data for the report, or the processing logic that creates the reports from the source data might contain errors. In the worst case, the cloud system itself was compromised (e.g. by a malicious competitor) and all reports are slightly modified so that they look conclusive but contain slightly reduced profit margins, intended to make a competing company look bad—or even insolvent.

1) Dual Execution: In such a situation, a first and trivial approach for verification might be that a cloud customer triggers the creation of its annual accounting report more than once. For instance, instead of giving the same request to one cloud provider only (called *Cloud A* hereafter), a second cloud provider (called *Cloud B*) that offers an equivalent type of service is invoked in parallel. By placing the same request at Cloud A and Cloud B, a cloud user can immediately identify whether his request was processed differently in Cloud A and Cloud B. Hence, this way, a secret exploitation of either side's service implementation would be detected. However, besides the doubled costs of placing the same request twice, this approach additionally relies on the existence of at least two different cloud providers with equivalent service offerings and comparable type of result. Depending on the type of cloud resources used, this is either easily the case—as even today there already exist many different cloud providers offering equivalent services (see Section I)—or difficult in cases in which very specific resources are demanded.

2) n Clouds Approach: A more advanced, but also more complex approach comes from the distributed algorithms discipline: the *Byzantine Agreement Protocol*. Assume the existence of n cloud providers, of which f collaborate maliciously against the cloud user, with $n > 3f$. In that case, each of the n clouds performs the computational task given by the cloud user. Then, all cloud providers collaboratively run a distributed algorithm that solves the *General Byzantine Agreement* problem (e.g. the *TurpinCoan* [24] or *Exponential Information Gathering* [32, 6.2.3] algorithms). After that it is guaranteed that all non-malicious cloud providers know the correct result of the computation. Hence, in the final step, the result is communicated back to the cloud user via a Secure Broadcast algorithm (e.g. plain flooding, with the cloud user taking the majority as the result). Hence, the cloud user can determine the correct result even in presence of f malicious clouds.

3) Processor and Verifier: Instead of having Cloud A and Cloud B perform the very same request, another viable approach consists in having one cloud provider "monitor" the execution of the other cloud provider. For instance, A may announce intermediate results of its computations to a monitoring process run at B. This way, B can verify that A makes progress and sticks to the computation intended by the cloud customer. As an extension of this approach, B may run a model checker service that verifies the execution path taken by A on-the-fly, allowing for immediate detection of irregularities.

One of the major benefits of this approach consists in its flexibility. B does not have to know all details of the execution run at A—especially not about the data values processed—but is able to detect and report anomalies to the cloud customer immediately. However, the guarantees given by this approach strongly depend on the type, number, and verifiability of the intermediate results given to B.

B. Splitting of Application Tiers

Assume a SaaS-based service named PhotOrga which allows its users to upload and manage their photos as well as share them with their family, friends and other contacts. For this purpose, PhotOrga provides an adequate access control system. In such a setting, how can the user be sure that this access control system has been implemented correctly and effectively? Since the application logic and the data storage of the PhotOrga system are tightly integrated, a flaw in the application logic might have side-effects on the access control to the photos. This might result in an unwanted data leakage (such as in the Google Docs case mentioned in Section II-B).

The separation of the application logic layer and the data persistency layer with the assignment to two distinct clouds reduces the data leakage risk in the presence of application logic flaws. Since the data is not directly accessible by the application, design or programming errors in the application do not have such a widespread effect as in the integrated scenario.

This scenario can be extended to a lot of other services including email, documents, spreadsheets, and so forth.

C. Splitting of Application Computations

For multi-party computation in the cloud—as mentioned before—there are two types of applications. First, an existing multi-party computation application can be directly outsourced to the cloud. As the secrets are shared between the multiple parties, data confidentiality is ensured. An example for a real-world application of secure multi-party computation is a sugar beet auction in Denmark [33]. This auction is used by farmers selling their sugar beets to the processing company Danisco. The farmers' input to the auction depends on their economic situation and productivity which they do not want to reveal to a competitor or to Danisco. Clearly, Danisco also does not want to give away the auction. As a trusted third party is not easily found, the easiest solution was to set up a multi-party computation between servers of the farmer's union, Danisco, and a supporting university team.

The second application for multi-party computation in the cloud is safeguarding the confidentiality for outsourced data processing. Regarding again the task of creating the annual accounting report from Section IV-A, apart from data integrity the property of data secrecy might be required.

Especially, for highly visible stock corporations the details of the accounting must be kept confidential. Otherwise, insider trading or other effects on the stock market are possible. However, due to the nature of the accounting report creation (only needed once a year, provider offers special software, etc.) it still might be useful to perform this task inside the cloud. In this case, using secret sharing and multi-party computation with two cloud providers offers the required properties. As we assume that different cloud providers do not cooperate, it is not required to use more than two providers. This limits the overhead created by the multi-party computation.

For less strict confidentiality requirements also obfuscating splitting can be used for the report creation task. For example, the calculation of earnings and expenses can be partitioned amongst two different providers. These tasks can be performed independently without any significant overhead. In this case, the amount of loss or profit—which is typically the most confidential value—remains undisclosed to the cloud providers.

V. Conclusion

The cloud enables its adopters to economically enjoy the massive computational power, bandwidth, storage, and software provisioned as a service and shared in a pay-per-use manner. Despite the tremendous benefits that come out of these properties, security is the primary obstacle especially for users consuming and producing confidential assets. Thus, the cloud is an intrinsically insecure platform from the viewpoint of the cloud user. Due to the lack of practicable security mechanisms that allow to protect sensitive information by enabling computations with encrypted data or to protect users from malicious behaviors by enabling the validation of the obtained results, the current usage of the cloud is heavily depending on a strong trust relationship between the cloud service provider and the cloud user.

In this paper a concept is introduced, which aims at reducing the required level of trust and which provides innovative cloud security mechanisms in form of architectural patterns. Each of the three presented architectures provides a framework for implementing practicable security services not available so far. The underlying idea is to deploy and distribute the tasks to multiple distinct cloud systems. The main advantage coming out of the presented architectures are security services which still hold in the presence of malicious or compromised clouds.

This comes obviously at a certain cost. Since multiple clouds are adopted at the same time, the number of clouds used denotes the factor in which the costs increase. Nevertheless, even when adopting one of the introduced approaches, the total cost might still be less than running the service in-house. The question here is, what a user is willing to pay for increased assurance and security. Since there are numerous analogies in other disciplines where e.g. the replication of resources is a common practice despite the fact of additional costs coming with this approach, the provided security benefits might weight-out the additional costs.

Further research and development work has to go into the direction of protocols for the federation of the multiple distinct clouds. Related with this aspect is the development of a specific middle-ware which enables a policy-driven, transparent and seamlessly adoption of multiple clouds for the user. Along these lines, the various possibilities to implement the suggested architectures need to be investigated and evaluated to achieve an understanding on their properties and applicability.

Some instantiations of the architectural patterns can be deployed without the specific support of the cloud providers. Some instantiations do require the specific support by the cloud providers. This raises of course the question on the incentives for the cloud provider to do so. Beside an increased fee for enhanced security services, the cloud provider might find the proposed architectures appealing to show the willingness to protect the customers' assets as much as possible. And since no cloud service provider can absolve oneself from being vulnerable to attacks for some degree, with the adoption and support of the introduced architectures it can be explicitly confirmed, that the duty to take care of the customer's entities is considered with the necessary and trust building responsibility.

References

[1] Amazon Web Services Blog, "Amazon S3 - Bigger and Busier Than Ever." [Online]. Available: http://aws.typepad.com/aws/2011/01/amazon-s3-bigger-and-busier-than-ever.html

[2] Gartner, "Gartner Executive Programs Worldwide Survey of More Than 2,000 CIOs Identifies Cloud Computing as Top Technology Priority for CIOs in 2011." [Online]. Available: http://www.gartner.com/it/page.jsp?id=1526414

[3] P. Mell and T. Grance, "The NIST Definition of Cloud Computing, Version 15," National Institute of Standards and Technology, Information Technology Laboratory, 2010. [Online]. Available: http://csrc.nist.gov/groups/SNS/cloud-computing/

[4] Amazon Web Services, "Amazon Elastic Compute Cloud (Amazon EC2)." [Online]. Available: http://aws.amazon.com/ec2/

[5] ——, "Amazon Simple Storage Service (Amazon S3)." [Online]. Available: http://aws.amazon.com/s3/

[6] Savvis, "Savvis Symphony." [Online]. Available: http://www.savvisknowscloud.com/services/

[7] RackSpace, "RackSpace Cloud." [Online]. Available: http://www.rackspacecloud.com/index.php

[8] Google, "Google App Engine." [Online]. Available: https://appengine.google.com

[9] Microsoft, “Windows Azure.” [Online]. Available: http://www.microsoft.com/windowsazure/

[10] Cloudscale, “The Cloudscale Platform.” [Online]. Available: http://cloudscale.com/

[11] SalesForce, “SalesForce.” [Online]. Available: http://www.salesforce.com/

[12] Google, “Google Docs.” [Online]. Available: https://www.google.com/docs

[13] Zoho, “Zoho Productivity Applications.” [Online]. Available: http://www.zoho.com/

[14] F. Gens, “IT Cloud Services User Survey, pt.2: Top Benefits & Challenges,” Blog post on IDC Survey, 2008. [Online]. Available: http://blogs.idc.com/ie/?p=210

[15] D. Hubbard and M. Sutton, “Top Threats to Cloud Computing V1.0,” Cloud Security Alliance, 2010. [Online]. Available: http://www.cloudsecurityalliance.org/topthreats

[16] M. Jensen, J. Schwenk, N. Gruschka, and L. Lo Iacono, “On technical security issues in cloud computing,” in *Proceedings of the IEEE International Conference on Cloud Computing (CLOUD-II)*, 2009.

[17] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, “Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds,” in *CCS ’09: Proceedings of the 16th ACM conference on Computer and communications security*. New York, NY, USA: ACM, 2009, pp. 199–212.

[18] N. Gruschka and L. Lo Iacono, “Vulnerable Cloud: SOAP Message Security Validation Revisited,” in *ICWS ’09: Proceedings of the IEEE International Conference on Web Services*. Los Angeles, USA: IEEE, 2009.

[19] M. McIntosh and P. Austel, “XML signature element wrapping attacks and countermeasures,” in *SWS ’05: Proceedings of the 2005 workshop on Secure web services*. New York, NY, USA: ACM Press, 2005, pp. 20–27.

[20] J. Kincaid, “Google privacy blunder shares your docs without permission,” TechCrunch, 2009. [Online]. Available: http://techcrunch.com/2009/03/07/huge-google-privacy-blunder-shares-your-docs-without-permission/

[21] J. Somorovsky, M. Heiderich, M. Jensen, J. Schwenk, N. Gruschka, and L. Lo Iacono, “Breaking the clouds – security analysis of cloud management interfaces,” *(in submission)*, 2011.

[22] D. Bernstein, E. Ludvigson, K. Sankar, S. Diamond, and M. Morrow, “Blueprint for the intercloud - protocols and formats for cloud computing interoperability,” *Internet and Web Applications and Services, International Conference on*, vol. 0, pp. 328–336, 2009.

[23] A. Celesti, F. Tusa, M. Villari, and A. Puliafito, “How to enhance cloud architectures to enable cross-federation,” in *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*. Los Alamitos, CA, USA: IEEE Computer Society, 2010, pp. 337–345.

[24] R. Turpin and B. A. Coan, “Extending binary byzantine agreement to multivalued byzantine agreement,” *Inf. Process. Lett.*, vol. 18, no. 2, pp. 73–76, 1984.

[25] I. Koren and C. M. Krishna, *Fault-Tolerant Systems*. Morgan Kaufmann, 2007.

[26] J. D. Wisner, G. K. Leong, and K.-C. Tan, *Principles of Supply Chain Management: A Balanced Approach*. South-Western College Pub, 2011.

[27] J. Vijayan, “Vendors tap into cloud security concerns with new encryption tools.” [Online]. Available: http://www.cio.com.au/article/376252/vendors_tap_into_cloud_security_concerns_new_encryption_tools/

[28] A. C. Yao, “Protocols for secure computations,” in *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science, FOCS’82*. IEEE Computer Society, 1982, pp. 160–164.

[29] A. Shamir, “How to share a secret,” *Commun. ACM*, vol. 22, pp. 612–613, 1979.

[30] I. Damgård, M. Geisler, M. Krøigaard, and J. B. Nielsen, “Asynchronous multiparty computation: Theory and implementation,” in *Public Key Cryptography, PKC ’09*, ser. Lecture Notes in Computer Science. Springer, 2009, pp. 160–179.

[31] M. Burkhart, M. Strasser, D. Many, and X. Dimitropoulos, “SEPIA: Privacy-Preserving Aggregation of Multi-Domain Network Events and Statistics,” in *USENIX Security Symposium*, 2010.

[32] N. A. Lynch, *Distributed Algorithms*. Morgan Kaufmann, 1996.

[33] P. Bogetoft, D. L. Christensen, I. Damgrd, M. Geisler, T. P. Jakobsen, M. Krigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter, M. I. Schwartzbach, and T. Toft, “Secure multiparty computation goes live.” in *Financial Cryptography and Data Security, FC’09*, ser. Lecture Notes in Computer Science, R. Dingledine and P. Golle, Eds., vol. 5628. Springer, 2009, pp. 325–343.