CloudSurfer
A Cloud Broker Application for Security Concerns

Milena Frtunic[1], Filip Jovanovic[1], Mladen Gligorijvic[1], Lazar Dordevic[1], Srecko Janicijevic[1],
Per Håkon Meland[2], Karin Bernsmed[2] and Humberto Castejon[3]
[1]*Norwegian University of Science and Technology, Trondheim, Norway*
[2]*SINTEF ICT, Trondheim, Norway*
[3]*Telenor Research and Future Studies, Trondheim, Norway*

Keywords: Cloud, Security, Broker, Requirements, SLAs.

Abstract: The broker is foreseen to take an important role in the future Cloud ecosystem. A Cloud broker will simplify the relationships between Cloud providers and customers, by aggregating, integrating and customizing services in accordance to the customers' needs. This paper demonstrates how security requirements can be a part of the Cloud brokering model. We present CloudSurfer, which is a prototype implementation of an independent Cloud broker that allows the customer to search for services that fulfill a set of security requirements. The application has been evaluated by representatives from the software industry and academia, and is freely available for further research.

1 INTRODUCTION

Cloud Computing has seen an tremendous growth in recent years. As more providers are entering the field and the competition in the Cloud service market is increasing, new business models for integrating and reselling services are emerging. The Cloud broker represents a promising and ambitious approach. According to NIST (Liu et al., 2011), a Cloud broker is "An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between Cloud Providers and Cloud Consumers". The main role of the Cloud broker is hence to help potential Cloud customers to navigate through the jungle of Cloud service offerings by acting as an intermediate layer between the customer and the different providers.

Existing Cloud brokers are often concerned with functional requirements, i.e. finding a Cloud service that fulfils one or more technical goals. Recent research has pointed out the need for brokering based on non-functional requirements, such as availability, performance, scalability and security. In particular security, which often is cited as a showstopper for the uptake of Cloud computing services, has been identified as one of the main priorities for Cloud brokering.

The purpose of this paper is to demonstrate how security requirements can be a part of Cloud service brokering. Our method is based on design science research (DESRIST) (Hevner and Chatterjee, 2010) and we have created CloudSurfer, a prototype implementation of a broker that allows the customer to search for services and browse through service offerings, based on a set of security requirements. These requirements are selected from a repository tailored for Cloud computing based on standards and guidelines from organisations such as NIST (Jansen and Grance, 2011) Cloud Security Alliance (Cloud Security Alliance, 2012) and Enisa (Hogben and Dekker, 2012).

2 RELATED WORK

Cloud services brokerage has become one of the hottest Cloud topics. This is reflected by the number of Cloud brokering platforms commercially available today. Vendors such as Jamcracker[1], Parallels[2], NEC[3], AppDirect[4] and CE On-demand[5], to name but a few, all offer white-label Cloud brokering solutions to communication and IT service providers. While the amount of features offered by these platforms varies,

[1] http://www.jamcracker.com/

[2] http://www.parallels.com/products/pacm/

[3] http://www.nec.com/en/global/solutions/cloud/index.html

[4] http://www.appdirect.com/

[5] http://www.ceondemand.com/EN/whatwedeliver/product/Pages/default.aspx

Frtunic M., Jovanovic F., Gligorijvic M., Dordevic L., Janicijevic S., Meland P., Bernsmed K. and Castejon H..
CloudSurfer - A Cloud Broker Application for Security Concerns.
DOI: 10.5220/0004375001990206
In *Proceedings of the 3rd International Conference on Cloud Computing and Services Science* (CLOSER-2013), pages 199-206
ISBN: 978-989-8565-52-5

all of them implement a one-stop-shop concept for IaaS, PaaS and SaaS services. End-users of these platforms, normally SMBs or enterprises, are offered a marketplace where they can find and access cloud services fitting their needs. Users can search for services based on the service type, but very few of the platforms allow end-users to find services based on non-functional requirements, such as QoS or security. A notable exception is the Cloud Finder from Intel[6], which allows the user to specify high-level security requirements such as *”Provider has a local datacenter in Europe”* and *“Dedicated private cloud with physical isolation offering available”*. However, SLA management support is also minimum in all platforms, if at all existent.

In the research arena, a number of projects are also working on Cloud brokering solutions. The OPTIMIS project (OPTIMIS Consortium, 2010) aims at defining a framework and toolkit for multi-Cloud architectures, such as broker-based architecture (Ferrer et al, 2012). In OPTIMIS, QoS parameters are used for the selection of Cloud infrastructure providers, with special emphasis on trust, risk, eco-efficiency and cost. Security requirements are however not considered. The mOSAIC project (mOSAIC Consortium, 2012) intends to create an open source Cloud API and platform targeted for developing SLA-aware multi-Cloud oriented applications. From the end-user’s point of view, the main component is the Cloud Agency, a broker that will assist applications in discovering Cloud resource providers, negotiating SLAs with these providers, and monitoring the SLA fulfillment. The platform will use an ontology for Cloud services (Moscato et al., 2011), where security is included as a non-functional requirement, but only on a high level. Another related project is Contrail (Contrail Consortium, 2010). It aims at designing and implementing an open source solution for SLA-aware federation of Clouds, such as allowing Cloud resources from different providers to be aggregated and exploited as if they belong to the same Cloud. In Contrail, SLAs include both traditional QoS requirements, such as availability, as well as Quality of Protection requirements, such as data confinement mechanisms and data location (Jensen, J. et al., 2011). Tordsson et al. (Tordsson et al., 2012) have proposed an IaaS Cloud broker mechanism intended to provide Cloud users with the requested number of virtual machines from multiple providers with the best cost/performance ratio, according to a given budget. Their focus is on the total infrastructure capacity and price, but they do not mention security requirements as possible constraints when selecting Cloud providers. Vaquero et al. (Vaquero et al., 2012) have described a rule-based architecture for managing service behaviour in the Cloud, which is comparable to a broker architecture. A similar approach is described with the Claudia in-between management layer for handling multiple Cloud providers (Rodero-Merino et al., 2010).

3 THE CLOUDSURFER PROTOTYPE

CloudSurfer is a prototype Cloud broker that can be used to search for and browse through Cloud service offerings that fulfill a set of security requirements. This section presents an overview of the development, design and implementation of the prototype.

3.1 Development Method

CloudSurfer was developed as a part of project course at the Norwegian University of Science and Technology (NTNU) during fall 2012[7]. The software was developed by a group of M.Sc. in computer science students in their forth year of study, using the Scrum methodology. The first six weeks of the projects were spent on planning, background research on machine-readable languages and software design. The implementation was conducted through six sprints, each of them lasting one week, including brief evaluations and requirement adjustments between the sprints. The first four sprints were used to implement the user interface and the translation tool, which is used to translate the customer’s security requirements into a machine-readable language. These two tasks were performed in parallel. The last two sprints were used to merge the user interface with the translation tool and to develop a matching system. Finally, the application was subject to functional testing and a more thorough usability evaluation

3.2 Design and Interaction

CloudSurfer was designed as a classical client-server application, and a conceptual overview is shown in Figure 1. The customer, or future Cloud consumer, utilizes a Web-based graphical user interface, which is further described in section 3.3. The business logic resides on the server side with the main main logical components:

- **Select.** Allows the user to select which require-

[6] http://www.intelcloudfinder.com/

[7] http://www.idi.ntnu.no/emner/tdt4290/

ments to consider, and further specify them with attribute values.

- **Translate.** Translates the selected requirements into a machine readable language.
- **Match.** Reasons on which Cloud providers are capable of fulfilling the security requirements.
- **Statistics.** Part of the administrative interface that allows you to see which security requirements are most popular.
- **Requirements Storage.** A repository of typical security requirements for Cloud computing, organized according to service type and category.
- **Template Storage.** Sets of pre-defined requirements for given service types.

The details on the server-side technology is described in section 3.4.

CloudSurfer depends on that the Cloud providers advertise their offered security controls in a machine-readable form, as shown in the lower-most part of Figure 1. There is no prevalent standard for expressing contract requirements and offerings, but based on the recommendations by Meland et al. (Meland et al., 2013), WS-Agreement (Andrieux et al., 2003) was chosen. This language is extensible and allows the use of any service terms, but has no built-in predicates for security. Therefore an XML schema extending WS-Agreement was defined. Note that a provider may very well advertise several service offerings, depending for instance on pricing range.

The sequence diagram, displayed in Figure 2, represents the most typical use case where a customer (service consumer) wants to search for a Cloud provider based on his own choice of security requirements. The sequence starts with a selection of a service type, in order to do an initial filtering of requirements. After this, the customer asks for the interface to specify requirements. The selected requirements are then submitted and the broker transforms these into an XML file which is used for matching against the offered security controls from the Cloud providers. The resulting conformance is sent back to the customer, who has the possibility of modifying the requirements.

3.3 The User Interface

The user interface is based on standard web technology such as HTML[8] and CSS[9], with dynamic features enabled by JavaScript[10]. CloudSurfer makes use of the jQuery[11] library for HTML document traversing, event handling, animating and Ajax interactions.

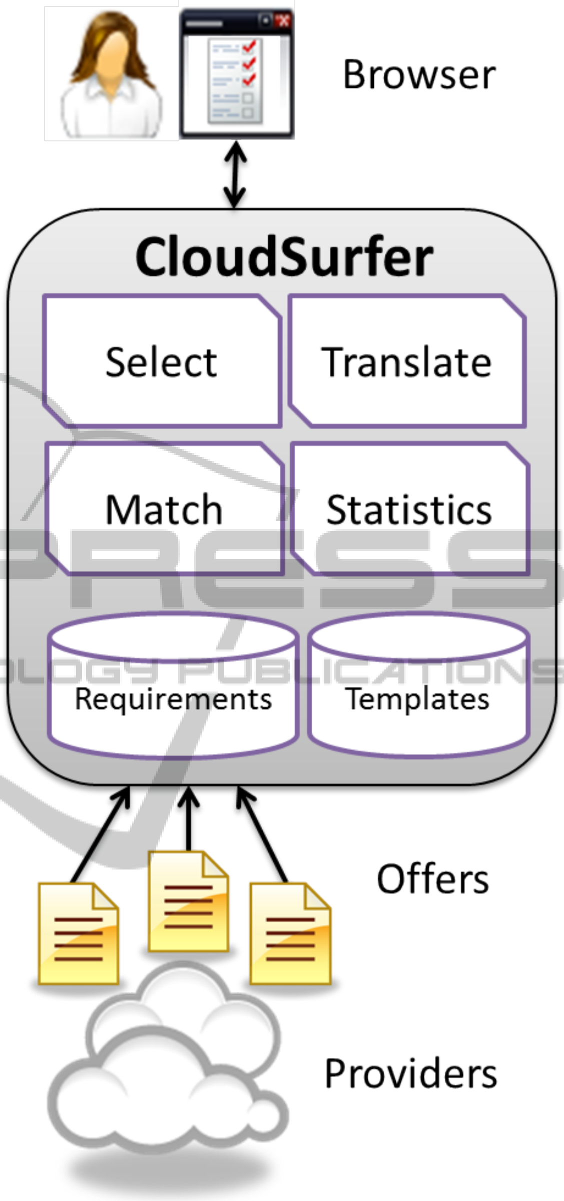


Figure 1: Conceptual design for CloudSurfer.

Usability was one of the main concerns for this prototype. The requirements storage consists of a large number of possible security requirements, and in order to make selection manageable by the customer the user interface has adopted a wizard-based approach. The user will be guided through a series of eight categories of requirements; namely

- Data Storage. This category includes security requirements related to the storage of customer data

[8] http://www.w3.org/html/

[9] http://www.w3.org/Style/CSS/

[10] http://www.ecma-international.org/publications/standards/Ecma-262.htm

[11] http://docs.jquery.com/

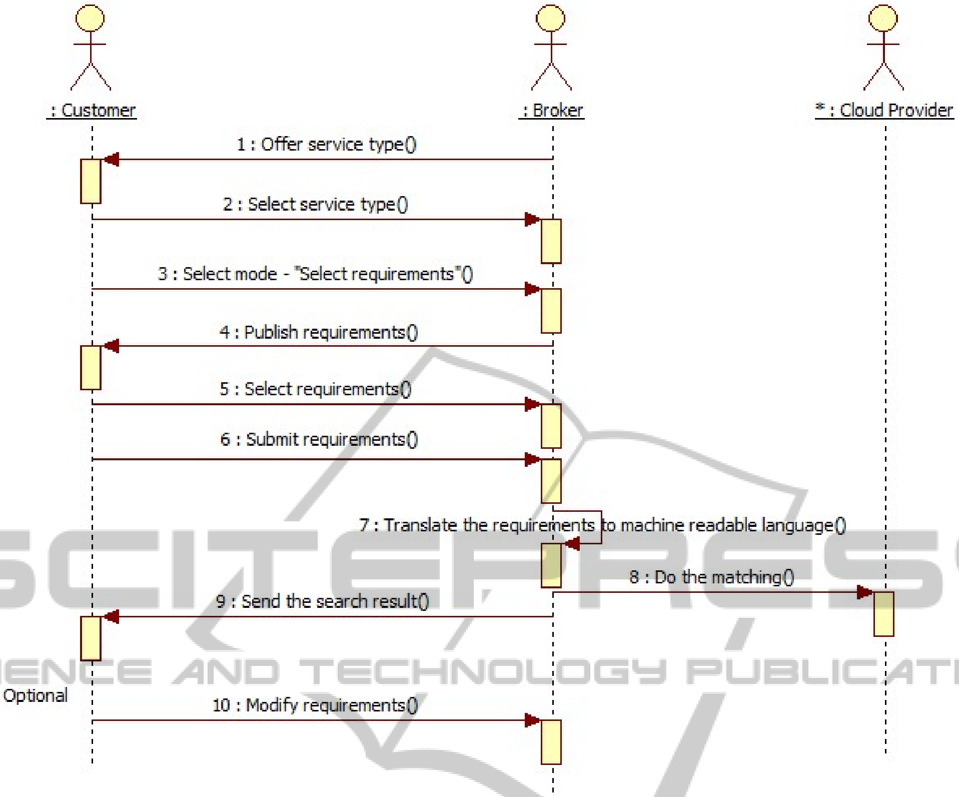
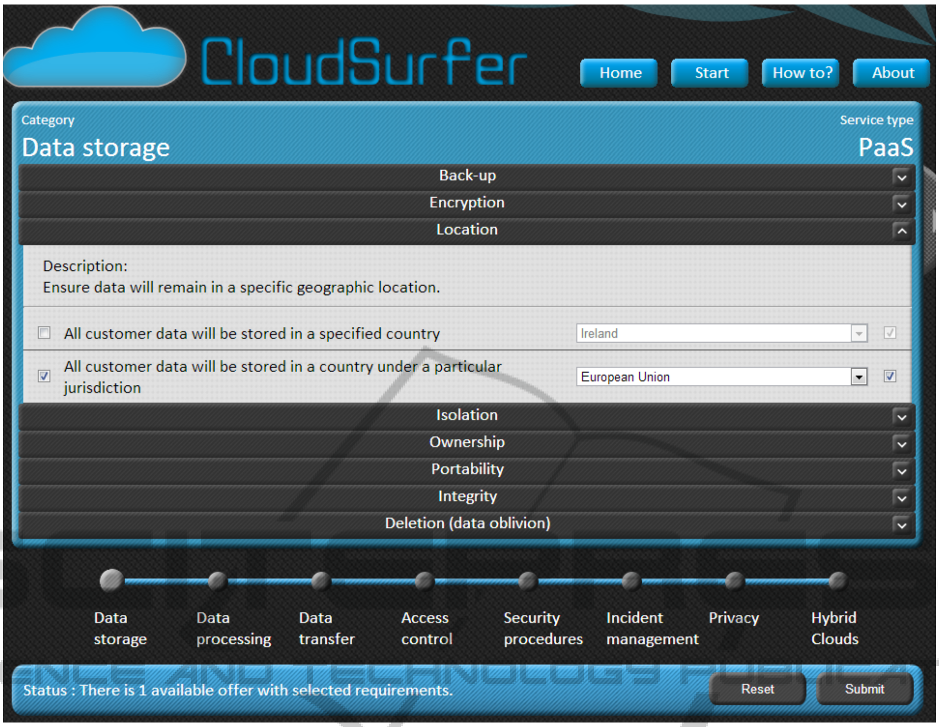

: Customer
: Broker
* : Cloud Provider
1 : Offer service type()
2 : Select service type()
3 : Select mode - "Select requirements"()
4 : Publish requirements()
5 : Select requirements()
6 : Submit requirements()
7 : Translate the requirements to machine readable language()
8 : Do the matching()
9 : Send the search result()
Optional
10 : Modify requirements()

Figure 2: Logical sequence diagram for Cloud brokering.

in the Cloud. Here we include issues related to backup of customer data, encryption, the physical location of the data center where the data is being stored, the isolation of the customer's data from other tenants, the ownership of data, portability, integrity and the secure disposal of customer data.

- Data Processing. This category includes security related to the processing of customer data in the Cloud. Here we include requirements related to isolation, monitoring, location, migration and encryption of customer data that is being processed in the Cloud.
- Data Transfer. This category includes security related to the transferring of customer data; both regarding the upload and download link to the Cloud as well as internally in the Cloud and between different Cloud data centers.
- Access Control. This category includes secure access to the Cloud service management interface, secure access for Cloud service users, physical access control and the availability of APIs for access control.
- Security Procedures. This category includes aspects related to auditing, certification, countermeasures and detection mechanisms, security testing procedures, communication between the provider and customer and key management procedures.
- Incident Management. This category includes requirements for incident response management, logging of incidents, reporting and forensic issues.
- Privacy. This category ensures the privacy and anonymity of the customer and the customer's data.
- Hybrid Clouds. This category includes issues related to the outsourcing of services to 3rd parties, the surveillability of hybrid Clouds and the binding and separation of duties.

These categories are divided into a finer grain of subcategories, which typically contain 3-5 security requirements each. Figure 3 shows an example screenshot where there are two requirements under *Data storage - Location*. The check boxes on the left side must be selected for the requirement to be considered, and the check box on the right side can be selected for requirements that are not mandatory but rather "nice to have". For requirements that need an attribute value, this must be filled in by using input fields or selecting boxes. A status bar at the bottom of the screen



Category
Data storage

Service type
PaaS

Back-up

Encryption

Location

Description:
Ensure data will remain in a specific geographic location.

□ All customer data will be stored in a specified country — Ireland ☑

☑ All customer data will be stored in a country under a particular jurisdiction — European Union ☑

Isolation

Ownership

Portability

Integrity

Deletion (data oblivion)

Data storage — Data processing — Data transfer — Access control — Security procedures — Incident management — Privacy — Hybrid Clouds

Status : There is 1 available offer with selected requirements.

Figure 3: Requirements are selected and detailed.

is constantly updated based on the number of matching Cloud providers.

When the form has been submitted the user will be presented with an overview over the most suitable providers, ranked by how well they match. The user can at any time go back and modify the requirements in order to find more candidates. Figure 4 shows a screenshot where the user can inspect in more detail how the requirements match against the offered security controls.

3.4 Server-side Technology

The server-side part of CloudSurfer consists of an Apache HTTP server[12] that interacts with the browser, and all the business logic is written in PHP. All information about security requirements and service types are stored in a MySQL[13] database. Interaction with the database is always done through SQL queries from the business logic, never from the client directly. The business logic validates and handles the data from the upper layer and hands it over to the data storage layer. The server also has an administrative set of functionalities, e.g. showing statistics and for managing the security requirements.

3.5 License and Availability

CloudSurfer has been designed and implemented as a part of a research project as a mean to gain experience and feedback on Cloud brokering. In order to facilitate further work on this topic the prototype has been made freely availability under a MIT license[14] and the source code is downloadable from SourceForge[15].

4 RECOMMENDATIONS

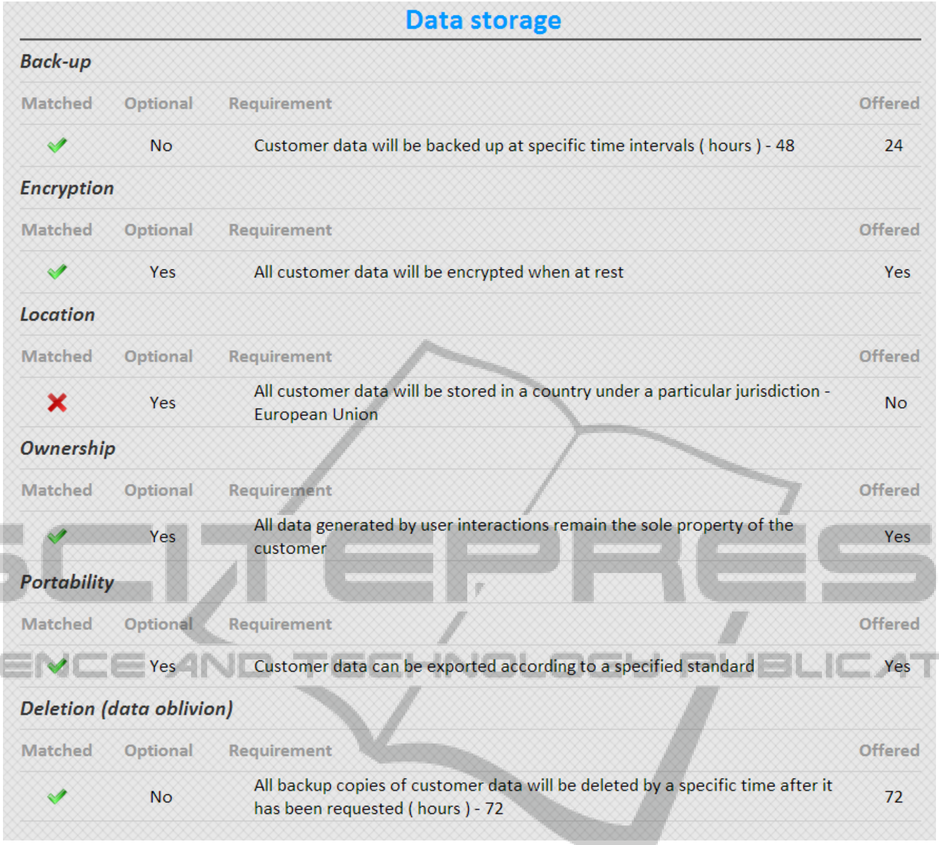
After the last sprint the prototype was evaluated by a group of 15 people both from the software industry and students at the university. This was done through live demonstrations where feedback was given directly, and by filling in an online questionnaire afterwards. The questionnaire covered application-specific topics related to usability, organization of information, presentation, performance and design, but also on the general concept of Cloud brokering and ideas for improvement. More details on the questionnaire and the responses can be found in the project report (Frtunic et al., 2012). This section provides recommendations, which are based on the main findings from the evaluation.

[12] http://httpd.apache.org/

[13] http://www.mysql.com/

[14] http://opensource.org/licenses/MIT

[15] http://sourceforge.net/projects/cloudsurfer/



Data storage

Back-up

Matched	Optional	Requirement	Offered
✓	No	Customer data will be backed up at specific time intervals (hours) - 48	24

Encryption

Matched	Optional	Requirement	Offered
✓	Yes	All customer data will be encrypted when at rest	Yes

Location

Matched	Optional	Requirement	Offered
✗	Yes	All customer data will be stored in a country under a particular jurisdiction - European Union	No

Ownership

Matched	Optional	Requirement	Offered
✓	Yes	All data generated by user interactions remain the sole property of the customer	Yes

Portability

Matched	Optional	Requirement	Offered
✓	Yes	Customer data can be exported according to a specified standard	Yes

Deletion (data oblivion)

Matched	Optional	Requirement	Offered
✓	No	All backup copies of customer data will be deleted by a specific time after it has been requested (hours) - 72	72

Figure 4: Matching results based on category.

4.1 A Personalized Customer Panel

Customers that visit CloudSurfer and specify their desired security requirements may wish to save their set of requirements, or even create multiple sets, for future use. That would save time if the customer is looking for another Cloud service that needs to comply with the same security policy. It would also allow the customer to check if the requirements are still fulfilled if the provider changes the offered security controls.

4.2 Registration of Offers and Authentication of Providers

CloudSurfer currently requires the providers to manually upload XML files describing their offered security controls, but it is an open question if these offers should be stored and managed centralized or made available by the providers in a way that allows the broker to crawl around and find them. If there is any change in the security controls the broker needs to be notified immediately so that it can relay this to the customer. The broker is likely to be held accountable if there is some kind of violation and it has based its matching on an outdated set of security controls. One of the evaluators suggested utilizing natural language processing on the terms that are already published at the providers' web sites. Though these are intended for human readers, it might be possible to extract this information into a deontic form; however, it would be difficult for the broker to give strong guarantees that the offerings are correct.

During the evaluation it was also pointed out that without proper authentication of providers, it would be possible to create false offerings in the name of reputable providers. To mitigate this threat there should be mechanisms that certify the identity of the providers and allow them to digitally sign their offers.

4.3 Negotiation Support

Currently CloudSurfer matches the static offers from Cloud providers with the requirements from the customer. The customer may interactively modify the requirements in order to accommodate an offer, but the provider is not actively involved in this process. The provider can of course make available several offers for the customer to choose from, but we also see the need of having two-sided dynamic negotiations between the customer and the provider, with the broker as the middle-man. SLA negotiations are today considered to be a very resource demanding manual process, often involving lawyers and exchange of ambiguous terms and conditions. Since it often takes so long to settle an agreement, the contracts usually have a long time-span and are seldom re-negotiated. A Cloud broker application where requirements and security controls are clearly stated according to a common vocabulary would simplify and make this a much cheaper process. It is even plausible that the negotiation can be automatically done by software agents for much shorter time frames (e.g. weeks instead of years). Agent-based negotiation is a field that has existed for quite some time, and could potentially be of great benefit for Cloud brokering.

4.4 Useful Statistics

The current support for statistics in CloudSurfer is limited to show the most popular requirements requests. This type of functionality has a lot of potential, and should be specified according to for instance service types. A possible business model for the broker is to provide information about which requirements are of particular concerns for the customers so that the providers will have a market advantage.

It was also suggested that the broker could take the role of gathering feedback from the customers, e.g. whether they experienced that the services were delivered according to the agreed upon terms. This information would be of great value to other customers, but there is also a danger that such reputation systems could be poisoned with false reports by competitors.

4.5 Composite Matching

CloudSurfer only matches single sets of requirements against atomic sets from the providers. However, there should be an option for the customers to match their requirements against Cloud services that are composed by multiple providers. The broker can act as an aggregator of Cloud services, but must make sure that the increased service complexity complies with the security requirements.

4.6 Requirements on the Service Customers

As we have already pointed out, CloudSurfer is designed to help customers find Cloud providers based on requirements from the customer. However, in some cases a Cloud provider may also provide strict requirements that the customer must fulfill, for instance what kind of services that may be deployed on their infrastructure. Service discovery based on the terms of use has so far been out of scope for CloudSurfer, but should definitively be considered for a full-fledged system.

4.7 Requirements Linked to extended, External Sources

The evaluation revealed that some of the requirements fail to be self-explanatory and that further explanations are needed in order to understand what they actually mean and how important they are. Therefore it would be beneficial to the user to link some kind of help for each requirement in the form of question mark next to the requirement. This link could lead to standards documents for security requirements, examples, user stories, etc.

4.8 Easy Comparison of Similar Offers

It sometimes happens that two different offers get the same score from the matching module. For example, let us say that the customer requested the back-up interval for his stored data to be at least 12 hours. The first offer has a 24 hrs. back-up interval and the second offer has back-up interval of one week. Currently these two offers are presented to the customer as being equal, and it would be helpful if the broker could indicate which offer is the closest one to the customer's needs. An alternative would be to replace the simple binary match with some kind of distance metric, in order to better visualize the difference between the different service offerings.

The evaluation also revealed the need for an easier way of comparing offers, for instance by presenting them next to each other on the same screen. This way a user can see exact differences between similar offers. It is not always the case that the offer with the highest number of matching security controls is the one the customer would prioritize.

4.9 Create and Sign Contracts

Our broker is limited to service discovery, but once this is done the next step would be to create and sign an SLA. The broker application should have functionality that automatically generates an agreed-upon contract and provide a mechanism for digitally signing the contract by all the involved parties.

4.10 Templates for Application Types

The initial filtering of requirements is based on a high-level classification of Cloud services types, notably SaaS, IaaS, PaaS and hybrids. We believe it would be very helpful to create a larger set of templates for more specific service types (e.g., e-mail, document sharing, video storage, back-up, etc.) and even within specific domains (e.g., healthcare, education, e-Governance, etc.). Knowing what the right requirements are for your service is considered to be a great challenge by itself, and quality-assured templates can be excellent starting-points for many customers. The broker should over time be able to create such template libraries and could offer them to new customers.

5 CONCLUSIONS

The CloudSurfer prototype implementation has been our vehicle of research for the Cloud brokering paradigm, and we have exploited it to identify future needs and challenges that we will try to address. The evaluation studies of the application itself showed a very good user experience, and there is certainly a need to simplify the process of finding Cloud service providers that fulfill the requirements of a customer. Our focus has been limited to security requirements, since they tend to be among the great show-stoppers for Cloud uptake, but we think that the application itself can easily be extended to support other types of requirements as well, for instance related to cost, functionality, performance and other QoS attributes.

Though we have been able to identify many new needs, it seems obvious to us that the major obstacle today is the lack of a standardized machine-readable contract language that can be used for automatic discovery and reasoning. It is imperative that the Cloud provider industry come to an agreement on what to describe and how to do it. In the end, it is the providers that are able to provide clear and distinct contract terms that will win the customers.

ACKNOWLEDGEMENTS

The research leading to these results has been supported by Telenor through the SINTEF-Telenor research agreement and the European Union Seventh Framework Programme (FP7/2007-2013) under grant no 257930. We would also like to thank the anonymous reviewers that provided us with excellent feedback and suggestions for improvements.

REFERENCES

Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., and Xu, M. (2003). Web Services Agreement Specification (WS-Agreement). https://forge.gridforum.org/projects/graap-wg/.

Cloud Security Alliance (2012). CSA Cloud Controls Matrix. Technical report. https://cloudsecurityalliance.org/research/ccm/.

Contrail Consortium (2010). Contrail FP7 EU Project. http://contrail-project.eu/.

Ferrer et al (2012). OPTIMIS: A holistic approach to cloud service provisioning. *Future Gener. Comput. Syst.*, 28(1):66–77.

Frtunic, M., Jovanovic, F., Gligorijvic, M., Dordevic, L., and Janicijevic, S. (2012). CloudSurfer. Security Requirements for Cloud Brokering. Customer Driven Project, project report, NTNU.

Hevner, A. and Chatterjee, S. (2010). *Design Research in Information Systems: Theory and Practice*. Springer Publishing Company, Incorporated, 1st edition.

Hogben, G. and Dekker, M. (2012). Procure Secure: A guide to monitoring of security service levels in cloud contracts. Technical report. http://www.enisa.europa.eu/activities/Resilience-and-CIIP/cloud-computing/.

Jansen, W. and Grance, T. (2011). Guidelines on Security and Privacy in Public Cloud Computing. NIST Special Publication 800-144.

Jensen, J. et al. (2011). SLA Management Services Terms and Initial Architecture. Contrail deliverable D3.2.

Liu, F., Tong, J., Mao, J., Bohn, R., Messina, J., Badger, L., and Leaf, D. (2011). NIST Cloud Computing Reference Architecture. NIST Special Publication 500-292.

Meland, P., Bernsmed, K., Jaatun, M., Castejon, H., and Undheim, A. (2013). Expressing cloud security requirements for SLAs in deontic contract languages for cloud brokers. *International Journal of Cloud Computing (to appear)*.

mOSAIC Consortium (2012). mOSAIC Cloud. http://www.mosaic-cloud.eu/.

Moscato, F., Aversa, R., Di Martino, B., Fortis, T., and Munteanu, V. (2011). An analysis of mOSAIC ontology for Cloud resources annotation. In *Computer Science and Information Systems (FedCSIS), 2011 Federated Conference on*, pages 973 –980.

OPTIMIS Consortium (2010). OPTIMIS FP7 EU Project. http://www.optimis-project.eu/.

Rodero-Merino, L., Vaquero, L. M., Gil, V., Galán, F., Fontán, J., Montero, R. S., and Llorente, I. M. (2010). From infrastructure delivery to service management in clouds. *Future Gener. Comput. Syst.*, 26(8):1226–1240.

Tordsson, J., Montero, R. S., Moreno-Vozmediano, R., and Llorente, I. M. (2012). Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. *Future Gener. Comput. Syst.*, 28(2):358–367.

Vaquero, L., Morn, D., Galn, F., and Alcaraz-Calero, J. (2012). Towards runtime reconfiguration of application control policies in the cloud. *Journal of Network and Systems Management*, 20:489–512.