Security Risk Optimization for Multi-Cloud Applications

Rudolf Lovrenčić, Domagoj Jakobović, Dejan Škvorc and Stjepan Groš

University of Zagreb, Faculty of Electrical Engineering and Computing

November 2019

Abstract – **Security proved to be a major concern when organizations outsource their data storage and processing. Encryption schemes do not provide solutions as they disable data processing in the cloud. Researchers have used constraint-based data fragmentation to increase security while maintaining availability. We build on this approach by applying fragmentation to the application logic in addition to the data in the database and propose a model for security risk assessment in a multi-cloud environment. By applying a multi-objective optimization algorithm to the proposed model, we determine pareto-optimal distributions of application and data fragments to the available cloud providers.**

1. Introduction

With the increased popularity of cloud computing in the past decade, it is no longer a question whether or not a company will embrace cloud computing. Rather, the question is when the technology will be implemented and which services should be migrated to the cloud. Remote access to a pool of computing resources reduces the up-front IT infrastructure costs and allows companies to meet fluctuating demands.

Current research indicates that the biggest challenges in cloud adoption are related to trust since companies may feel like they are losing control over their data [1]. Numerous data breaches and security vulnerabilities [2, 3] prevent users from trusting *cloud service providers* (CSPs). Compliance with the industry specific regulations and even general information security regulations often make moving to the cloud difficult since regulations may differ from region to region. High flexibility of a cloud service makes exhaustive and continuous security revisions expensive or intractable [4].

Hybrid cloud environments enable users to combine their computing resources with the cloud to retain more control over their data. Confidential data can, for example, be stored or encrypted on premise before leaving the local environment. Such approach makes regulation compliance easier, but burdens the user with key management. Furthermore, encrypted data cannot be used by applications running in the cloud.

Recent trends show that the use of multiple cloud providers simultaneously is increasing to achieve higher service availability and damage reduction in the case of malicious insiders on a single CSP [5]. Such *multi-cloud* environments mitigate reliance on a single cloud provider.

The contribution of this paper is threefold. Firstly, we introduce a multi-cloud application model that assumes multiple application components and multiple data fragments. Secondly, we propose two risk metrics for assessing the risk of a given application deployment to the multi-cloud environment. Lastly, by using multi-objective optimization, we find a set of pareto solutions according to two proposed risk metrics.

1.1. Literature review

Security is a major concern of the cloud platform and is one of the main research directions regarding the cloud computing [6]. Fragmentation has been recognized as a possible solution to improving data security while still enabling query evaluation at the provider side.

Using fragmentation as a method for increasing privacy in data storage has been explored [7]. Sensitive data relations can be broken to decrease information leakage in case of an attack. Confidentiality constraints have been introduced as a means for describing sensitive data relations. Algorithms for finding optimal fragmentation based on the constraints have also been proposed [7, 8], but such algorithms do not touch on finding the optimal distribution of data fragments to the available servers.

Researches have used fragmentation and distribution between multiple cloud providers [9]. Similarly to earlier work [7], user defined constraints are taken into consideration during data

fragmentation and distribution in the cloud. The paper shares motivation and assumptions with our work: cloud providers are non-colluding, usage of multiple CSPs enables better regulation compliance and minimal use of encryption maximizes data availability in the cloud. Work is focused on security in data storage and does not touch on data security during computations performed by the cloud applications.

Information entropy can be used to measure the sensitivity of connections between the data [10]. This can help in deduction of confidentiality constraints. Such approach requires the database to be filled with the actual or representative data which can be an issue when data distribution is not known in advance.

The rest of this paper is structured as follows. Section 2 presents a simple distributed database and distributed application model. Section 3 applies a risk assessment method for the distributed application deployed in a multi-cloud environment. Optimization algorithm and preliminary results of risk optimization are discussed in Section 4. Section 5 concludes the paper and explores possible future work.

2. Distributed model

We assume that both, application logic and the data, are distributed. Since any cloud provider may be malicious, storage and computation are split and distributed among available CSPs. Optimal distribution of data fragments and application components ensures minimal information leak in case of an incident on a single CSP.

The application database is split into N_F data fragments and the application logic is split into N_C application components. *Data fragment* F_i is a portion of data that can be stored to any available CSP. Figure 2.1 illustrates vertical fragmentation on a simple table. The original table is split into two fragments: F_1 containing the name and the surname of a person and F_2 containing the payout amount for each person. Fragmentation aims to decouple the person and the payout amount. Malicious access to only one of the data fragments results in a significantly lower information leak than access to the both fragments. If F_2 is leaked, only payout amounts are known to the attacker. The names of involved persons are

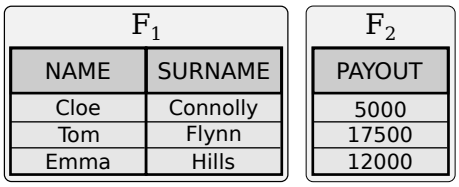

Figure 2.1: Simple relational table split.

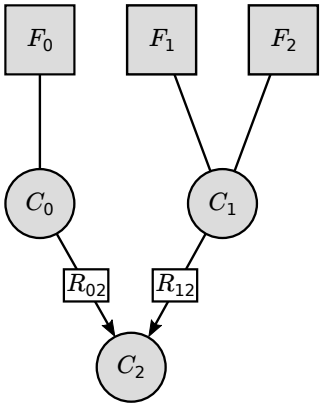

Figure 2.2: Simple application model.

compromised if fragment F_1 has leaked. When such fragments are provided to the deployment optimization process for multi-cloud applications, it will attempt to deploy those fragments to different CSPs to maximize security.

Similar to the data fragment, *application component* C_j is a segment of the application logic that can be deployed to and executed on any available CSP. Each component is able to perform three actions:

1) access data fragments (read or write),
2) receive data from other components,
3) send data to other components.

Result R_{ab} represents the data sent from source component C_a to the destination component C_b. Each result has exactly one source and exactly one destination application component.

Figure 2.2 illustrates a simple application consisting of 3 components: C_0, C_1 and C_2. Components C_0 and C_1 access the data fragments and perform computations. Computation results R_{02} and R_{12} are sent to the component C_2.

Since the application components are treated as black boxes, no assumption can be made for their outputs. Consequently, results exchanged between

the components must be treated as resources that carry information, likewise the data fragments. For example, component C_0 may simply forward input data to its output making the result R_{02} identical to F_0. Therefore, a *resource* ρ_k that the risk assessment process considers is either a data fragment F_i or result R_{ab} exchanged between the application components.

3. Security risk assessment

Fragmentation can be used for increasing security in the data storage by breaking sensitive data relations to decrease information leak in case of attack [7]. Security constraints are used for describing sensitive data relations.

User provides N_K security constraints for the multi-cloud application. Each constraint contains a subset of all resources. A *security constraint* $K_l = \{\rho_1, \ldots, \rho_n\}$ defines a property that sum of information of each resource ρ_k in the constraint is smaller than the information of all resources merged together. This concept is formalized in the expression 3.1 where $I(\rho_1, \ldots, \rho_n)$ is the amount of information leak when resources $\rho_1, \ldots, \rho_n$ leak together.

$$\sum_{k=1}^{n} I(\rho_k) < I(\rho_1, \ldots, \rho_n) \tag{3.1}$$

If all resources contained within a security constraint are present on a single CSP, the constraint is considered violated since more information will leak in case of an incident on that provider.

3.1. Resource reach

A set of fragments and components where a resource ρ_k is available defines its *resource reach* $D(\rho_k)$. Therefore, the reach of a data fragment is a set that contains that data fragment and all components that access that data fragment. For example, resource reach of fragment F_1 shown in Figure 2.2 is a set containing fragment F_1 and component C_1.

Results that components exchange are transferred from a single source component to a single destination component. Consequently, the reach of a component result is a set containing two elements: result source and result destination. Reach of the result R_{12} that can be seen in Figure 2.2 is $D(R_{12}) = \{C_1, C_2\}$.

Resource reach enables efficient check if a constraint can be satisfied in the ideal case where unlimited amount of cloud providers is available. Constraint $K_l = \{\rho_1, \ldots, \rho_n\}$ can be satisfied in the ideal case if and only if:

$$D(\rho_1) \cap \ldots \cap D(\rho_n) = \emptyset \tag{3.2}$$

Evaluation of a distribution of fragments and components in the multi-cloud environment is also made simple with the use of resource reach. Violation of a security constraint $K_l = \{\rho_1, \ldots, \rho_n\}$ is checked in the following way:

1) The reach of each resource in the constraint is calculated: $D(\rho_i)$, $i = 1, \ldots, n$
2) *Cloud resource reach* $D_c(\rho_i)$ is calculated by substituting each fragment and component in $D(\rho_i)$ with a cloud provider where that fragment or component is deployed. $D_c(\rho_i)$ is therefore a set of CSPs where resource ρ_i is available.
3) Multi-cloud distribution satisfies constraint K_l if and only if:

$$D_c(\rho_1) \cap \ldots \cap D_c(\rho_n) = \emptyset \tag{3.3}$$

The procedure is repeated for each security constraint during the risk assessment of a particular deployment of fragments and components. Since the same resources are often part of multiple security constraints, computed cloud resource reach $D_c(\rho_i)$ may be cached to avoid duplicated calculations.

3.2. Risk metrics

We define two metrics for assessing the security risk of a particular multi-cloud deployment:

a) violated security constraints,
b) individual security of resources.

The first metric penalizes when all resources within a security constraint are available on the same CSP. The second metric estimates security of resources individually and is responsible for pushing more important resources towards more trusted CSPs during deployment optimization. Given a set of available CSPs $P = \{\sigma_i, \ldots, \sigma_S\}$, trust estimate function $t\colon P \to \mathbb{R}^+$ must be provided by the user which assigns the trust estimate to each CSP. Higher trust estimate implies that the provider has a lower chance of leaking information. In addition to pushing resources to more trusted CSPs as

much as possible, optimization process attempts to violate unsatisfiable constraints at the most trusted providers.

The risk for breaking security constraint K_l is calculated using the expression:

$$r_{1l} = \frac{p_l}{t(\sigma_l)}, \tag{3.4}$$

where $p_l \in \mathbb{R}^+$ is the penalty for breaking the constraint K_l and σ_l is the cloud provider where the constraint is violated. If multiple CSPs violate the security constraint, CSP with the lowest trust estimate is used.

Expression 3.5 computes individual risk of resource ρ_k with *value* $v_k \in \mathbb{R}^+$ assigned by the user.

$$r_{2k} = \sum_{\sigma \in D_c(\rho_k)} \frac{v_k}{t(\sigma)}, \tag{3.5}$$

More important resources should be assigned a higher value v_k. This enables critical resources to produce higher risks and have higher priorities in the optimization procedure.

4. Experimental setup and results

Metrics for assessing security provided in the previous section can be linearly combined. Total risk of a given deployment is then measured with a scalar. The issue with this approach is that the importance of each metric has to be expressed with a coefficient before optimization. Metrics can differ in scale rather greatly so choosing coefficients that yield good deployments may be difficult for non-trivial applications.

We use multi-objective optimization algorithm NSGA-III [11, 12] to avoid attributing importance to optimization criteria before actual optimization. Importance is attributed implicitly when one of the suggested deployments is chosen by the user. NSGA-III is a multi-objective evolutionary algorithm which aims to improve fit of a population of candidate solutions to a pareto front constrained by a set of objective functions. NSGA-II introduced elitism to the original NSGA algorithm. NSGA-III further improves the algorithm by using a method that increases solution diversity. This results in even distribution of solutions across the pareto front.

A solution is represented as a mapping of fragments and components to the cloud providers.

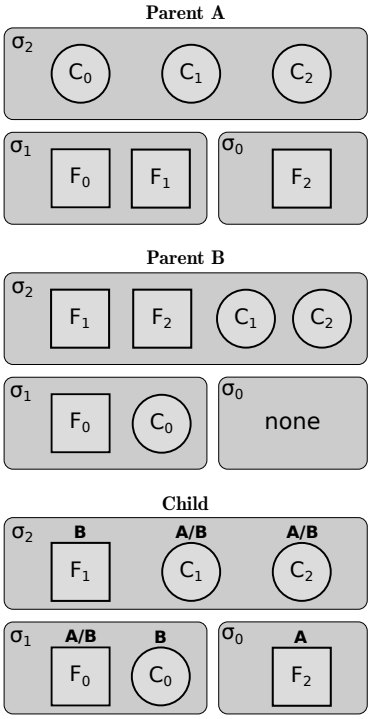

Figure 4.1: Crossover example.

In a scenario where three CSPs $(\sigma_0, \sigma_1, \sigma_2)$ are available, three possible solutions for simple application (Figure 2.2) are shown in Figure 4.1 where crossover operation is visualized. In the solution that represents parent B, CSP σ_0 is unused, two data fragments and two application components are deployed to σ_2, and fragment F_0 and component C_0 are deployed to CSP σ_1. Genotype is implemented as a hash table that maps fragments and components to the CSPs.

Crossover operator uniformly selects a cloud provider for each component from one of the parents. Child solution illustrated in Figure 4.1 is constructed by selecting CSPs for fragment F_1 and component C_0 from parent B. F_2 is deployed to the same CSP as in parent A, while CSPs for F_0, C_1 and C_2 could have been selected from either parent since they map them to the same cloud providers.

Two mutation operators are used with equal probability:

a) picks a random fragment or component and assigns it to a random cloud,
b) all fragments and components are assigned to random clouds (solution is reconstructed).

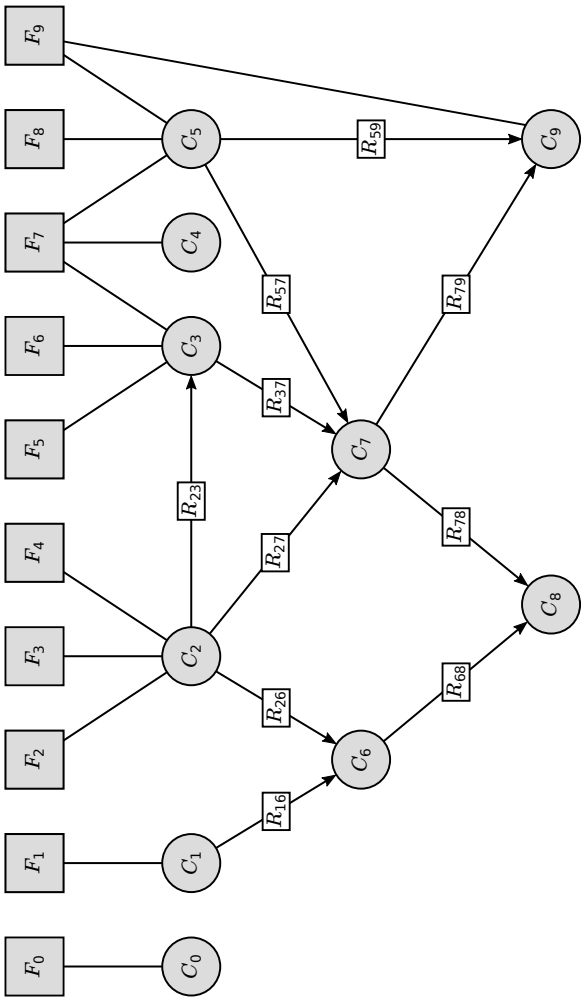

Figure 4.2: Test application.

It has been found that the enterprise applications often consist of many distinct business logic and backend components [13]. Fortune 100 companies have applications with dozens, sometimes even reaching hundred components.

Our test application (Figure 4.2) consists of 10 data fragments and 10 application components. We assume 7 available CSPs, σ_0 to σ_6, assigned with increasing trust values (σ_6 being the most credible). Such environment provides 7^{20} possible cloud deployments. The following set of security constraints is used: $\{F_0, R_{23}\}$, $\{F_2, F_6\}$, $\{F_3, F_4\}$, $\{F_3, F_6\}$, $\{F_3, R_{79}\}$, $\{F_7, R_{59}\}$, $\{F_9, R_{68}\}$, $\{R_{16}, R_{27}\}$, $\{R_{16}, R_{26}, R_{37}\}$, $\{R_{16}, R_{26}, R_{59}\}$. Penalization for breaking a security constraint is set to $p = 100$ for all security constraints.

Fragment	Value
F_0	100
F_1	200
F_2	300
F_3	400
F_4	500
F_5	600
F_6	700
F_7	800
F_8	900
F_9	1000

Result	Value
R_{16}	35
R_{23}	55
R_{26}	15
R_{27}	65
R_{37}	35
R_{57}	85
R_{59}	95
R_{68}	45
R_{78}	25
R_{79}	15

Table 4.1: Values of test application resources.

Table 4.1 lists values of application resources. Results transferred between components have lower values than pure data fragments since we assume that output is arbitrary transformation of the input that is less useful to the attacker (e.g. aggregation).

We use a population of 500 solutions and set maximum number of generations to 100 which results in 50000 evaluations of each objective function. Risks of solutions shown in Figure 4.3 are achieved by the NSGA-III algorithm with 15% mutation rate. The algorithm consistently provides similar fronts. During our testing, random search never provided solutions below 2000 single resource risk. When presented with a pareto front, the user is able to make better decisions since valuing importance of each risk metric is made easier – trade-offs between possible solutions are visualized.

Table 4.2 lists solutions A and B marked in Figure 4.3. Solution A violates three security constraints: $\{F_2, F_6\}$, $\{F_7, R_{59}\}$, $\{R_{16}, R_{27}\}$ on CSPs σ_5, σ_6 and σ_6 respectively. On the contrary, solution B violates only $\{F_2, F_6\}$ and $\{F_7, R_{59}\}$. Furthermore, both constraints are violated on the most trusted CSP σ_6 which results in significantly lower constraint risk. Solution A provides lower single resource risk as 15 resources are available on the single CSP while in B that is true for for 11 resources. Solution A also makes five most valuable resources only available on the most trusted CSP σ_6 which contributes to lower single resource risk when compared to the solution A.

Deterministic strategies for finding optimal deployments can be successful when only database fragmentation is considered [9], but do not scale to

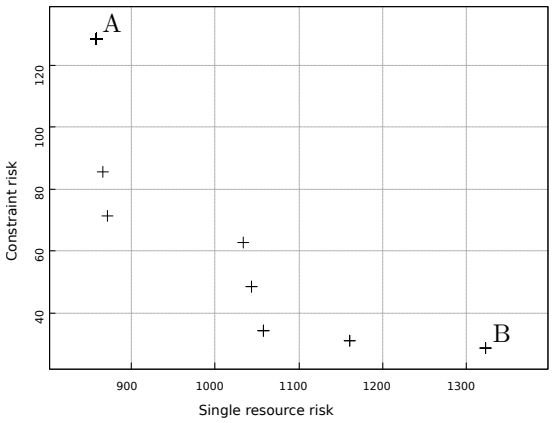

Figure 4.3: Risks of deployments proposed by NSGA-III.

Solution A	
σ_4	F_0, C_8, C_6, C_0
σ_5	F_4, F_3, F_2, C_2
σ_6	$F_9, F_8, F_7, F_6, F_5, F_1, C_9, C_7, C_5, C_4, C_3, C_1$

Solution B	
σ_2	C_8, C_6
σ_3	F_0, C_0
σ_4	F_6, F_5, C_7, C_3
σ_5	F_1, C_9, C_1
σ_6	$F_9, F_8, F_7, F_4, F_3, F_2, C_5, C_4, C_2$

Table 4.2: Notable solutions from Figure 4.3.

applications with many logic components and data fragments. While solutions provided by heuristics might not be optimal, performance they offer opens up possibilities for real-time recalculation of security risk and redeployment of multi-cloud applications.

5. Conclusion and future work

In this paper, we introduced a model for assessing the risk of distributed application and distributed database deployed in a multi-cloud environment. Using the model, we performed multi-objective optimization which provided us with pareto set of deployments with regards to different security criteria. The approach is not exclusive to the multi-cloud setting. It can be applied wherever a multitude of deployment locations for application components and database fragments are available.

The proposed risk assessment model does not take time into consideration – it is completely static. Information about how often and which ratio of certain data flows through each application component would increase the level of detail that the model can describe. Verbosity of such model might become an issue, but supporting tools could be developed to assist with describing real world applications. For example, information entropy can be used to help the user determine security constraints [10].

While applying security risk optimization on existing distributed applications is possible, the best results are obtained when a multi-cloud application is constructed from the ground up with security in mind. Guidelines and patterns for such development should be established so that the right techniques (e.g. cryptographic protections) can be applied in the right situations.

Furthermore, supporting database mechanisms must be established to enable transparent usage of fragmented data while ensuring that constraints imposed on the data are valid (e.g. primary key).

References

[1] S. Subashini and V. Kavitha, **A Survey on Security Issues in Service Delivery Models of Cloud Computing**, 2011.

[2] Chirag Modi et al., **A Survey on Security Issues and Solutions at Different Layers of Cloud computing**, 2013.

[3] Keiko Hashizume et al., **An Analysis of Security Issues for Cloud Computing**, 2013.

[4] Florian Kelbert et al., **SecureCloud: Secure Big Data Processing in Untrusted Clouds**, 2017.

[5] Mohammed Abdullatif AlZain et al., **Cloud Computing Security: From Single to Multi-clouds**, 2012.

[6] Rajkumar Buyya et al., **A Manifesto for Future Generation Cloud Computing: Research Directions for the Next Decade**, 2019.

[7] Valentina Ciriani et al., **Combining Fragmentation and Encryption to Protect Privacy in Data Storage**, 2010.

[8] Sabrina De Capitani Di Vimercati et al., **Fragmentation in Presence of Data Dependencies**, 2014.

[9] Aleksandar Hudic et al., **Data Confidentiality using Fragmentation in Cloud Computing**, 2012.

[10] Tie Hong et al., **A Novel Vertical Fragmentation Method for Privacy Protection Based on Entropy Minimization in a Relational Database**, 2018.

[11] Kalyanmoy Deb and Himanshu Jain, **An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints**, 2013.

[12] Kalyanmoy Deb et al., **A fast and elitist multiobjective genetic algorithm: NSGA-II**, 2002.

[13] Mohammad Hajjat et al., **Cloudward bound: planning for beneficial migration of enterprise applications to the cloud**, 2010.