Dynamic Obfuscation for Secure and Efficient Multi-Cloud Business Processes

Amina Ahmed Nacer[1] and Mohammed Riyadh Abdmeziem[2]
[1]*M'hamed Bougara University, 35000 Boumerdes, Algeria*
[2]*National School of Computer Science, 16000 Oued Smar, Algiers, Algeria*

Keywords: Business Process, Cloud Computing, Data Obfuscation, Obfuscation Methods, Cost-Effective Solution.

Abstract: Organizations increasingly outsource complex business processes to the cloud, but concerns about exposing business strategies persist. While existing solutions split processes across multiple cloud providers, they don't fully address the risk of information leakage. Our approach leverages random obfuscation techniques at each execution to safeguard sensitive data, offering a lightweight alternative to encryption. In multi-cloud environments, where processes are distributed across providers, obfuscation reduces leakage risks with lower computational overhead, making it ideal for resource-constrained scenarios compared to more expensive cryptographic solutions.

1 INTRODUCTION

Cloud computing has become a dominant technology, enabling organizations to minimize infrastructure costs and focus on core activities. This has spurred interest in executing business processes (BP) in the cloud, either by reusing BP fragments or deploying proprietary ones. Business process outsourcing, an advanced form of IT outsourcing, is increasingly popular due to its advantages (Lynn et al., 2014; Yang et al., 2007). However, security and strategic risks remain significant barriers, particularly for BP software, which encodes sensitive company know-how (Aron et al., 2005; ENISA, 2009; Alliance, 2014).

One solution to mitigate these risks is obfuscating BP models before cloud deployment. Works such as (Jensen et al., 2011; Goettelmann et al., 2015; Rekik et al., 2016) propose splitting BP models into fragments, deploying them across multiple clouds, and interconnecting them to form a choreography. While this approach protects the logic of the BP, analyzing input and output data of fragments could still expose sensitive business activities and strategies.

To address this, we consider two cloud scenarios: non-colluding clouds and potentially colluding clouds. Building on (Nacer et al., 2016; Nacer et al., 2018), this work focuses on enhancing data confidentiality by dynamically obfuscating sensitive data. Instead of transmitting unencrypted data, obfuscation techniques are applied, selected randomly for added security. Unlike encryption, obfuscation preserves data utility and is less computationally intensive, making it suitable for multi-cloud environments. This approach ensures data privacy, compliance with regulations, and process efficiency without compromising functionality.

The remainder of this paper is organized as follows: Sections 1.1 and 1.2 present an attacker model and the motivations, along with an overview of our approach. The subsequent section details the proposed data obfuscation method. Section 3 discusses evaluation results, and Section 4 reviews related works. Finally, the paper concludes with future research directions.

1.1 Attacker Model

Different configurations can be used by a malicious server or a coalition of providers (sharing their local information) for mining valuable information about the BP. In any case, we consider in this paper that the malicious clouds cannot control the network, but behave as *honest but curious* about the organization deploying the process (see figure 1).

Assume that we have in this problematic a data set D with n entries $|D| = n$. This data set is represented by a set of labels $L(D)$ and a set of values $V(D)$. Each data d $\in$ D has a label $L(d)$ and a value $V(d)$.

We define F as a collection of fragments formed

Nacer, A. A. and Abdmeziem, M. R.
Dynamic Obfuscation for Secure and Efficient Multi-Cloud Business Processes.
DOI: 10.5220/0013192100003944
In *Proceedings of the 10th International Conference on Internet of Things, Big Data and Security (IoTBDS 2025)*, pages 175-182
ISBN: 978-989-758-750-4; ISSN: 2184-4976

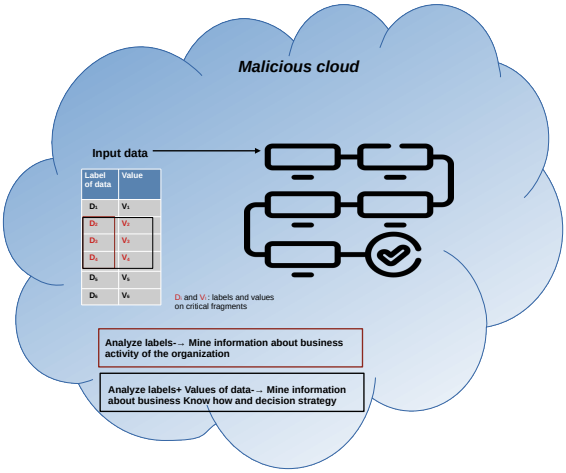

Figure 1: Attacker model.

by the division of the business process, where $|F| = m$ components. Each fragment f ∈ F has access to a subset of $L(D)$ named $L'(f)$ of labels.

We define $S(f_i)$ as a binary function that assigns a value of 1 to sensitive fragment and 0 to non-sensitive fragment

$S(f_i) = \{0,1\}$ where:

$$S(f_i) = \begin{cases} 1 \; if \; f_i \; is \; sensitive \\ 0 \; if \; f_i \; is \; not \; sensitive \end{cases}$$

We consider that two types of attacks can occur:

- **From Labels:** having access to an important amount of valuable labels can allow malicious clouds to mine the business activity of the organization. Valuable labels are considered those labels of data used by critical fragments. This can be formalized as follows:

  We define K as a collection of fragments that are distributed on each cloud in C, where $K \subseteq F$.

  $K(c) = \{f \in F : L'(f) \subseteq L(D), for \; d \in D\}$ for each cloud c ∈ C.

  If any $f \in F / S(f) = 1, L'(f)$ allows mining business activities of the company.

- **From Labels and Values:** we argue that having access to both labels and values of data of critical fragments can allow to mine information about the business know-how and can give indicators on its decision-making strategy. This can be formalized as follows:

  We define K as a collection of fragments that are distributed on each cloud in C, where $K \subseteq F$.

  For each cloud c ∈ C

  $$K(c) = \{f \in F : L'(f) \subseteq L(D) \wedge V'(f) \subseteq V(D), \text{ for } d \in D\}.$$

  If any of the $f \in F / S(f) = 1, L'(f) \wedge V'(f)$ enable mining information about the business know-how and can provide indicators on its decision-making approach.

1.2 Motivating Example

Figure 2 depicts a loan process in a bank using BPMN (von Rosing et al., 2015) with the objective of accepting or rejecting a loan request. The second part of the figure illustrates a collaboration of BP fragments of the same process, resulting from splitting the loan BP logic using the BP Model obfuscator introduced in (Nacer et al., 2018).

While splitting reduces the information managed by each cloud, vulnerabilities persist. A malicious cloud with access to critical data could infer process logic. For example, analyzing labels like *salary* with values such as **salary 2000$**, **personal contribution 100 000$**, and **loan amount 200 000$** might reveal business strategies or decision rules, especially in cases of cloud collusion.

1.3 General Overview of the Approach

Our methodology involves dynamically selecting obfuscation techniques for each execution of an outsourced business process. Using the same obfuscation algorithm repeatedly makes it easier for a malicious cloud to infer process logic through input-output analysis.

To counter this, a random number is generated to determine the obfuscation technique for each execution. The client obfuscates data using the selected technique and sends it, along with the random number, to the required service for execution.

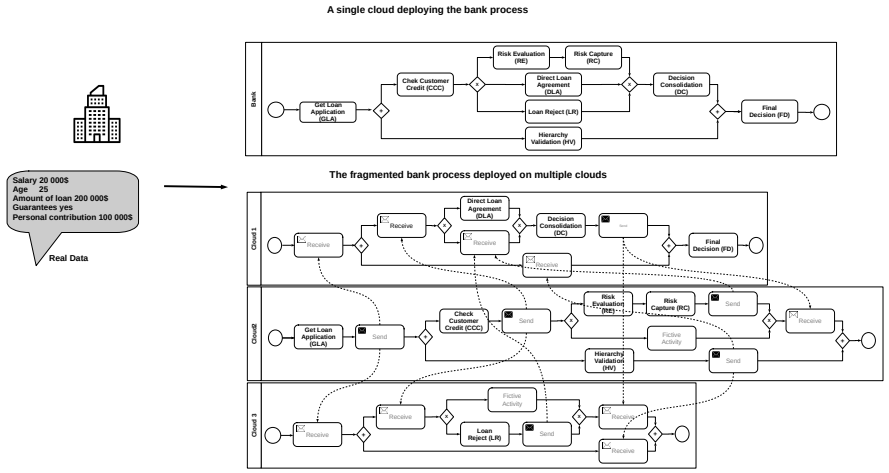

Figure 2: The *Loan* Process before and after cloud deployment.

The steps of the approach are as follows:

- The user generates a random number, selects an obfuscation technique based on it, and sends the obfuscated data and random number to the service.
- The service uses the random number to select the corresponding algorithm, processes the data, and forwards the result and random number to the next service.
- Subsequent services follow the same procedure, ensuring consistent obfuscation handling across the process.

This dynamic obfuscation approach adds an extra layer of security, reducing the risk of reverse-engineering by attackers.

Next, we present the obfuscation techniques used in this methodology.

2 DATA OBFUSCATION APPROACH

This section introduces and explains the different obfuscation techniques used in our approach for labels and values obfuscation.

2.1 Labels and Values Obfuscation

2.1.1 Labels Obfuscation

We introduce in this section the different obfuscation techniques used to obfuscate the labels of data.

- **Randomized Label Obfuscation (RLO):** is an approach to conceal the semantic content of data labels within organizational datasets. Unlike traditional methods, RLO relies on randomization to assign labels to data points, aiming to eliminate patterns and correlations that might compromise data confidentiality. The technique provides an advanced level of label concealment, reducing the risk of reverse engineering and inference attacks.

  Let L represent the set of original labels, and O represent the set of obfuscated labels. The RLO process is denoted by the function $F_{\text{RLO}} : L \to O$, where $F_{\text{RLO}}(l)$ is the obfuscated label corresponding to the original label l. The mapping between original and obfuscated labels is maintained on premise in a secure mapping table, denoted by T, where $T[l] = F_{\text{RLO}}(l)$.

- **Contextual Embedding-Based Obfuscation (CEO):** this method involves transforming labels into high-dimensional contextual embeddings, making it extremely challenging to infer specific details while preserving the richness of semantic information. CEO focuses on transforming labels into high-dimensional contextual embeddings using pre-trained language models, enhancing the confidentiality of label data. The approach aims to provide a highly secure and versatile solution for label obfuscation. Indeed, traditional tokenization or semantic mapping may not provide sufficient security for highly sensitive labels. CEO addresses this by leveraging advanced natural language processing techniques to embed labels into high-dimensional contextual representations.

  Let L represent the set of original labels, and E represent the set of contextual embeddings. The Contextual Embedding-based Obfuscation (CEO) process is denoted by the function $F_{\text{CEO}} : L \to E$,

Table 1: Data order of magnitude modification.

Variables	Range	Max magnitude	Min magnitude
$[n,m]$	$n \leq 10$ and $m \leq 10$	$[2^n, 2^{n+m}]$	/
	$n \geq 10$ and $m < 1000$	$\left[2^{\frac{n}{2}}, 2^{\frac{m}{2}}\right]$	/
	$n > 10000$	/	$\left[\frac{n}{10^{nb-1}} \times 2^{nb-1}, \frac{m}{10^{nb-1}} \times 2^{nb-1}\right]$
n	$n \leq 10$	2^n	/
	$n < 1000$	$2^{\frac{n}{2}}$	/
	$n > 1000$	/	$\frac{n}{10^{nb-1}} \times 2^{nb-1}$

Table 2: Data before and after obfuscation.

Variables	Data	Value	Obfuscated value
V_1	Age	$[20-60]$	$[1024-1073741824]$
V_2	Monthly salary	2000	16
V_3	Amount of the loan	100000	32

where $F_{\text{CEO}}(l) = \mathbf{e}_l$, signifying the transformation of each label l into its corresponding high-dimensional contextual embedding $\mathbf{e}_l$.

It is noteworthy that in this paper, we used BERT models (Devlin et al., 2018), as a pre-trained language models, for the implementation of the Contextual Embedding-based Obfuscation (CEO) method.

2.1.2 Value Obfuscation

We present in this section the proposed approach used to obfuscate values of data.

Data Range Modification. We change the order of magnitude of a value of some sensitive data as much as possible, in order to hide the information which can be extracted from these values. Depending on the original value, the order of magnitude can be maximized or minimized as depicted in table 1.

A malicious cloud analyzing the variables V_1, V_2 and V_3 (see table 2) can mine their meaning from their values and from the service executing it. Therefore to avoid that, we changed their values. A variable having a range of values $[1024-1073741824]$ can hardly be perceived as representing the age of the client. The same reasoning is applied to variables V_2 and V_3 : an amount of 16 for the salary and 32 for the loan.

Encoding Obfuscation. We use encoding obfuscation to modify the representation of decision variables (i.e. invert the logical values TRUE and FALSE). For example, if a decision strategy is that a loan cannot be attributed to a client which salary is not reaching a certain threshold, and the amount of the loan attributed cannot exceed 50 times the salary of the client, the value of a decision variable of a client fulfilling these criteria will be $false$.

Noise Data. We consider that the introduction of noise data only affects sensitive information. Basically, using additional amount of data, we obfuscate (cover, hide) meaningful information used by the service. The additional data includes random values having a magnitude that will be managed according to the random values' standard deviation.

To do so, we have to identify the statistical proprieties of the real data in terms of mean and variance. These proprieties will be used to generate noise data having proprieties that are different from the actual ones.

We define X as the real data and $\mathbb{E}(X) = \mu(X)$, $\text{Var}(X) = \sigma^2(X)$ as the mean and the variance of X respectively.

We define Z as the final set generated from adding noise data Y to X: $Z = X + Y$. We decided to add at the beginning only few data and to adapt it through the different executions according to the risk related to sensitive information. Moreover, the amount of generated data must be adapted to the amount of real ones. In fact, we consider that the more we have sensitive data, the more we need to introduce noise ones.

3 EXPERIMENTATION

In this section, we conduct a comprehensive evaluation of our approach through both security and performance analyses. Firstly, we assess the security aspect by comparing the effectiveness of our obfuscation technique through an analysis of obfuscated and deobfuscated data. Subsequently, we evaluate the performance of our approach by comparing it with established cryptographic methods namely Rijndael (Jamil, 2004) and Twofish (Schneier, 1998). The experiments were conducted using business processes of varying sizes to ensure a comprehensive assessment. All experiments were performed on an Intel(R) Core(TM) i5-2310M 2.10 GHz processor running Windows 10

3.1 Security Analysis

This subsection analyzes the security of our obfuscation techniques by comparing obfuscated and deobfuscated data. The similarity serves as a metric to assess their effectiveness in obscuring sensitive information. Diverse deobfuscation algorithms were used, tailored to the characteristics of the obfuscated data, to evaluate robustness against unauthorized access.

- For label obfuscation, we applied techniques like Randomized Label Obfuscation (RLO) and Contextual Embedding-based Obfuscation (CEO), leveraging machine learning algorithms such as k-Nearest Neighbors (k-NN) (Hastie et al., 2009) and Decision Trees (Breiman, 2017). These algorithms were selected for their ability to handle categorical data and effectively reconstruct original labels.
- For value obfuscation, including data range modification, encoding obfuscation, and noise addition, we used linear regression algorithms (Montgomery et al., 2021) to estimate original values from obfuscated data.

We used the Jaccard similarity measure (McCormick et al., 1992) to evaluate the overlap between original and deobfuscated data. This metric quantifies similarity by comparing the intersection and union of the two sets, providing a quantitative assessment of our obfuscation methods' accuracy. Figure 3 presents the results across varying numbers of business process activities.

The low similarity indices between obfuscated and deobfuscated data (ranging from 0.1 to 0.4) confirm the effectiveness of our obfuscation approach. Scores closer to 0.1 indicate significant divergence, suggesting successful obfuscation, while scores around 0.4 show partial similarity, which may be due to the obfuscation technique's complexity. This implies that the method successfully obscures sensitive data, making it difficult to fully reconstruct the original values, even after deobfuscation attempts.

Our approach further strengthens security by dynamically selecting different obfuscation techniques for each execution based on a random number. This introduces unpredictability, making reverse-engineering more difficult for potential attackers. The variability in similarity scores across different business processes emphasizes the importance of considering each process's specific characteristics when implementing obfuscation techniques.

3.2 Performance Analysis

To evaluate the performance of our approach, we compare it with two well-known cryptographic algorithms, Rijndael and Twofish (128-bit key size), across business processes of varying scales. This comparison provides insights into the efficiency and effectiveness of our approach relative to these established cryptographic methods. The analysis covers key performance metrics, including execution time (in milliseconds), memory usage (in kilobytes), and CPU usage (as a percentage).

Figure 4 shows the execution times for Rijndael, Twofish, and our approach across varying numbers of activities. Both Rijndael and Twofish show increased execution times as the number of activities grows, reflecting their computational complexity. For example, at 7 activities, the times are 10 milliseconds for Rijndael, 8 milliseconds for Twofish, and 4 milliseconds for our approach. At 8 activities, the times are 10.5, 7.25, and 6 milliseconds, respectively. Rijndael generally has slightly higher execution times than Twofish, though the difference is minimal. In contrast, our approach consistently performs faster, suggesting less computationally intensive operations. This likely results from focusing on data obfuscation instead of cryptographic security, providing a quicker alternative for situations where speed is crucial without compromising security.

3.2.1 CPU Usage

Figure 5 shows the CPU usage of the three approaches. Rijndael and Twofish generally use more CPU compared to our obfuscation approach, which consistently demonstrates lower usage across different numbers of activities. For instance, at 7 activities, the CPU usage for Rijndael, Twofish, and our approach is 0.05%, 0.03%, and 0.01%, respectively. At 8 activities, these are 0.06%, 0.05%, and 0.03%,

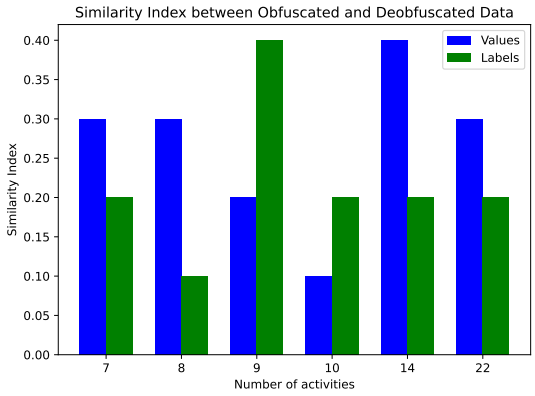

Figure 3: Index of similarity.

Figure 4: Execution time.

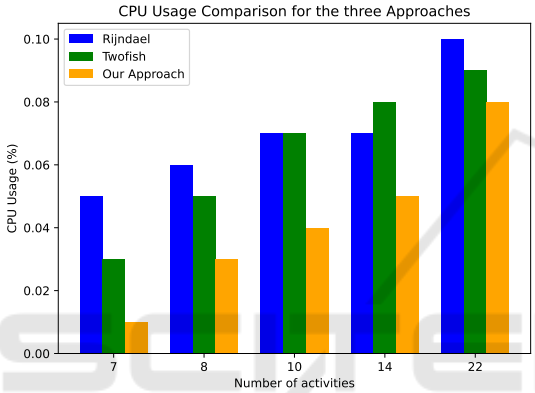

Figure 5: CPU usage.

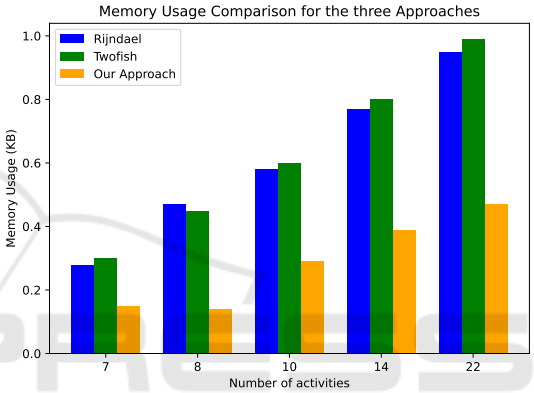

Figure 6: Memory usage.

respectively. This pattern continues as the number of activities increases, with our approach using significantly less CPU. The lower CPU usage can be attributed to the obfuscation technique's focus on protecting data without the complex computations involved in cryptographic algorithms like Rijndael and Twofish. This demonstrates the efficiency of our approach in terms of CPU utilization while maintaining security and performance.

3.2.2 Memory Usage

Figure 6 shows the memory usage of the three approaches. Across varying numbers of activities, our obfuscation approach consistently uses less memory compared to Rijndael and Twofish. For example, at 7 activities, the memory usage for Rijndael, Twofish, and our approach is 0.28 Kb, 0.3 Kb, and 0.15 Kb, respectively. This trend continues as the number of activities increases. The reduced memory usage highlights the efficient memory management of our obfuscation method. As the number of activities grows, our approach's advantage becomes more evident, demonstrating scalability and resource efficiency, which is particularly beneficial in resource-constrained environments. Additionally, the smaller memory footprint may result in cost savings in hardware and infrastructure. This comparative analysis confirms the effectiveness of our obfuscation approach in reducing memory usage while maintaining security and performance.

3.3 Discussion

Based on the comprehensive analysis conducted, our obfuscation approach demonstrates robust security measures and efficient performance metrics. The security analysis reveals the effectiveness of our techniques in obscuring sensitive information within business processes, as evidenced by low Jaccard similarity indices between obfuscated and deobfuscated data. This confirms the resilience of our obfuscation methods against unauthorized access.

In terms of performance, our approach consistently outperforms established cryptographic algorithms like Rijndael and Twofish in execution time, CPU usage, and memory usage. This indicates that

our approach involves less computationally intensive operations while still maintaining a high level of security. Overall, our obfuscation approach offers a balanced solution, prioritizing security without compromising on performance or resource efficiency. This makes it a viable alternative for various deployment scenarios, including resource-constrained environments where minimizing computational and memory overhead is crucial.

4 RELATED WORKS

Several mechanisms for privacy-preserving data before cloud deployment are proposed in the literature, often focusing on data encryption. However, encryption, including searchable and homomorphic encryption, can hinder cloud processing due to its performance limitations and high computational cost. These encryption techniques also require careful key management, which can introduce further complexity.

Alternatively, hybrid methods, such as the combination of AES and FHE (Kumar and Badal, 2019), offer data security while maintaining operational efficiency. Muralidhar et al. (Muralidhar and Sarathy, 2006) introduced data shuffling to mask confidential numerical data, enhancing utility while minimizing disclosure risks. Similarly, V Ciriani et al. (Ciriani et al., 2010) proposed a method that combines data fragmentation and encryption to protect sensitive information.

Other approaches focus on data partitioning and distribution across multiple cloud providers. Jensen et al. (Jensen et al., 2011) suggested using multiple clouds to reduce the risk of data manipulation and tampering. Zhang et al. (Zhang et al., 2013) and Celesti et al. (Celesti et al., 2016) explored splitting data into parts and distributing them across different clouds, improving security by minimizing exposure. However, these approaches often overlook the potential for data reconstruction through conspiracies among cloud providers.

5 CONCLUSION

To ensure the protection of business strategies and sensitive data before cloud deployment, we introduced an innovative approach that combines process fragmentation with dynamic obfuscation. This second level of obfuscation involves randomly selecting techniques at each execution, making it harder for attackers to reverse engineer the data.

Our evaluation demonstrates the effectiveness of this approach in obscuring sensitive information, as indicated by low similarity indices between obfuscated and deobfuscated data. Additionally, our method outperforms traditional cryptographic algorithms in terms of execution time, CPU usage, and memory usage, highlighting its efficiency.

Future work will focus on enhancing the adaptability and resilience of our approach to emerging security threats and refining evaluation metrics to assess obfuscation techniques more comprehensively.

DECLARATIONS

The authors have no competing interests to declare that are relevant to the content of this article.

REFERENCES

Alliance, C. (2014). Csa security, trust & assurance registry (star). *visited on*, pages 08–18.

Aron, R., Clemons, E. K., and Reddi, S. (2005). Just right outsourcing: Understanding and managing risk. *Journal of management information systems*, 22(2):37–55.

Breiman, L. (2017). *Classification and regression trees*. Routledge.

Celesti, A., Fazio, M., Villari, M., and Puliafito, A. (2016). Adding long-term availability, obfuscation, and encryption to multi-cloud storage systems. *Journal of Network and Computer Applications*, 59:208–218.

Ciriani, V., Vimercati, S. D. C. D., Foresti, S., Jajodia, S., Paraboschi, S., and Samarati, P. (2010). Combining fragmentation and encryption to protect privacy in data storage. *ACM Transactions on Information and System Security (TISSEC)*, 13(3):1–33.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

ENISA, C. C. (2009). Benefits, risks and recommendations for information security. *European Network and Information Security*, 23.

Goettelmann, E., Ahmed-Nacer, A., Youcef, S., and Godart, C. (2015). Paving the way towards semi-automatic design-time business process model obfuscation. In *2015 IEEE International Conference on Web Services*, pages 559–566. IEEE.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

Jamil, T. (2004). The rijndael algorithm. *IEEE potentials*, 23(2):36–38.

Jensen, M., Schwenk, J., Bohli, J.-M., Gruschka, N., and Iacono, L. L. (2011). Security prospects through cloud computing by adopting multiple clouds. In *2011*

IEEE 4th international conference on cloud computing, pages 565–572. IEEE.

Kumar, L. and Badal, N. (2019). A review on hybrid encryption in cloud computing. In *2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, pages 1–6. IEEE.

Lynn, T., O'carroll, N., Mooney, J., Helfert, M., Corcoran, D., Hunt, G., Van Der Werff, L., Morrison, J., and Healy, P. . (2014). Towards a framework for defining and categorising business process-as-a-service (bpaas). (2014). In *Proceedings of the 21st International Product Development Management Conference*, page 65.

McCormick, W., Lyons, N., and Hutcheson, K. (1992). Distributional properties of jaccard's index of similarity. *Communications in Statistics-Theory and Methods*, 21(1):51–68.

Montgomery, D. C., Peck, E. A., and Vining, G. G. (2021). *Introduction to linear regression analysis*. John Wiley & Sons.

Muralidhar, K. and Sarathy, R. (2006). Data shuffling—a new masking approach for numerical data. *Management Science*, 52(5):658–670.

Nacer, A. A., Goettelmann, E., Youcef, S., Tari, A., and Godart, C. (2016). Obfuscating a business process by splitting its logic with fake fragments for securing a multi-cloud deployment. In *2016 IEEE World Congress on Services (SERVICES)*, pages 18–25. IEEE.

Nacer, A. A., Goettelmann, E., Youcef, S., Tari, A., and Godart, C. (2018). A design-time semi-automatic approach for obfuscating a business process model in a trusted multi-cloud deployment: A design-time approach for bp obfuscation. *International Journal of Web Services Research (IJWSR)*, 15(4):61–81.

Rekik, M., Boukadi, K., and Ben-Abdallah, H. (2016). A comprehensive framework for business process outsourcing to the cloud. In *2016 IEEE international conference on services computing (SCC)*, pages 179–186. IEEE.

Schneier, B. (1998). The twofish encryption algorithm. *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, 23(12):30–34.

von Rosing, M., White, S., Cummins, F., and de Man, H. (2015). Business process model and notation-bpmn.

Yang, D.-H., Kim, S., Nam, C., and Min, J.-W. (2007). Developing a decision model for business process outsourcing. *Computers & Operations Research*, 34(12):3769–3778.

Zhang, W., Sun, X., and Xu, T. (2013). Data privacy protection using multiple cloud storages. In *Proceedings 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC)*, pages 1768–1772. IEEE.