Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Machine learning based deep job exploration and secure transactions in virtual private cloud systems ☆

S. Rajasoundaran[a], *A.V. Prabu*[b], *Sidheswar Routray*[b,*], *S.V.N. Santhosh Kumar*[c], *Prince Priya Malla*[b], *Suman Maloji*[b], *Amrit Mukherjee*[d,*], *Uttam Ghosh*[e]

[a] *School of Computing Science and Engineering, VIT Bhopal, India*
[b] *Department of Electronics and Communication Engineering, Koneru Lakshmaiah Education Foundation, Guntur, India*
[c] *School of Information Technology and Engineering, VIT Vellore, India*
[d] *School of Electronics and Information Engineering, Anhui University, P.R. China*
[e] *Electrical Engineering and Computer Science, Vanderbilt University, Tennessee, US, United States*

ARTICLE INFO

Article history:
Received 11 December 2020
Revised 15 May 2021
Accepted 15 June 2021
Available online 22 June 2021

Keywords:
Cloud system
Security
Zero-trust environment
Job transactions
Machine learning and job analysis

ABSTRACT

Virtual Private Cloud (VPC) is an emerging cloud environment used to provide more secure data communication. VPC provides authentic communication channel for secure communication between the cloud participants. The cloud jobs and the description of the runtime cloud events must be evaluated to provide flawless VPC service. Although VPC provides security in network services, it has to be enriched with internal and external platform level security features. In this regard, secure job service schemes ensure elimination of attacks, unauthorized jobs, improper accesses and intrusions in VPC. These irrelevant tasks (activities) can be isolated before initiating job scheduling process. Particularly, providing security for zero-trust cloud environment is more challenging task. Zero-trust cloud environment has completely vulnerable trust model on both internal and external circumstances. The proposed Machine Learning Based Secure Cloud Job Services (MLSCS) is implemented to provide multi-level security in this zero-trust cloud job servicing system. The proposed MLSCS develops Multi-Server Queue Management techniques, Reinforcement Learning based deep Q Matrix (RL-Q Matrix) techniques, Authentic VPC configuration and VPC Genetic Algorithm Network (VGAN) for establishing security practices in complex job handling system. MLSCS applies effective techniques for eliminating irrelevant cloud jobs to reduce the scheduler complexity and processor utilization. In this work, irrelevant jobs are considered as the jobs that are not appropriate for particular VPC scheduling policies and security principles (attacks). These jobs are identified through various key validation procedures and VPC policy determination procedures. These jobs are eliminated and prohibited in to job scheduler. Consequently, the legitimate jobs are securely forwarded in to job scheduler through multi-server queues. In the experimental setup, the proposed MLSCS is compared with existing schemes such as Reinforcement Learning Based Distributed Heterogeneous Servicing technique (RLDH), Cat Swarm Optimization Based Job Servicing technique (CSOS) and Fuzzy Based Security-Driven Servicing technique (FSDS). The results show the MLSCS delivers 5% to 8% optimal results than existing schemes.

https://doi.org/10.1016/j.cose.2021.102379
0167-4048/

1. Introduction

VPC is a demand basis configurable cloud section, which provides secure resource sharing in public cloud environment. VPC offers certain range of isolation between VPC users (customers) and generic cloud users. Practically, both types of users are involving under the same cloud environment. However, they are restricted and protected under VPC policies. VPC communication can be processes through a secure link (tunnel) using Virtual Private Networks (VPNs). VPN ensures security needs such as confidentiality, integrity and authentication etc. This network security may go insufficient for VPC due to its virtual deployment under public environment. The interactions between public cloud and VPC may cause platform level or service level insecurity. Apart from network security techniques, the specific platform level and service level security implementations are inevitable. VPC platform and services are completely dealing with users' jobs and job initiative processes. Though, VPC is protected from public cloud system using VPN perimeters, the need for VPC core security is always expected to protect the network from internal and external attacks (zero-trust conditions). Many types of researches were carried out to protect cloud environment. This proposed MLSCS concentrates on isolating unauthorized jobs and providing secure job scheduling under zero-trust VPC platforms.

Researches related to secure scheduling and attack detection techniques resolve many real-time issues to protect cloud environment policies (Muthurajkumar et al., 2018, Gunasekaran et al., 2016, Nancy et al., 2020). Most of the recent researches are establishing cryptography based secure scheduling techniques and Machine Learning (ML) based scheduling techniques. On the other hand, they never produced hybrid secure ML based job scheduling activities. The existing works concentrated on public cloud domains but not VPC and cloud interaction issues. The proposed MLSCS finds this issue as a research problem that to be resolved at runtime. In addition, MLSCS tries to resolve internal and external attacker jobs at queuing levels (Liu and Jiang, 2018, Flavel et al., March 26, 2013).

This proposed system deals with protected job transactions on unsecure cloud environment. The unsecure environment is considered as completely vulnerable environment or zero-trust environment. Notably, zero-trust environment is defined as the network environment without any trust policies. It states that the trust is completely vulnerable in the network. This zero-trust environment is defined for verifying all network participants (insiders and outsiders) without believing any trust policies. The proposed system is mainly focusing on both insider attackers and outsider attackers. The conventional security mechanisms assume that the organization network participants and the insiders are normally considered as trustable parties. However, this assumption is not acceptable under any real-time cloud security.

In this regard, the proposed design has been developed on the basis of zero-trust cloud conditions. The main goal of this proposed MLSCS model is providing highly secure job analysis frameworks and scheduling policies under zero-trust virtual cloud environment. This work hardly extracts each cloud job entities and verifies their security measures for initiating secure scheduling processes. This helps to protect the real-time cloud transactions from any malicious attackers of virtual cloud environment. Particularly, the technical constructions are carried out for initiating the secured VPC transactions such as secure multi-server queues, authentic VPC principles, RL-Q matrices and VGAN structures to make the network transactions more secured under zero-trust cloud environment.

In this work, a new system called MLSCS is proposed and implemented using multi-server queuing model, secure poisson job distribution, Authentic VPC management, RL-Q matrix construction, Triple Key based Direct Acyclic Graph (TDAG) construction, and VGAN construction procedures. In each phase of MLSCS, jobs are validated using their descriptors, properties, dependencies, triple key validations and genetic properties. According to the proposed techniques, novel and secure job handling algorithms are illustrated in section 3. In this system, multi-server queuing models are implemented for managing job arrivals in distributed manner. As illustrated in section 4, the proposed system develops multiple servers for handling numerous client machines or Virtual Machines (VMs). In this setup, each server receives multiple job instances from various clients.

In this distributed cloud structure, the need for multi-level queues in each VPC server is mandatory. Under each VPC server, the organized queues maintain job arrival rate and job departing rate. This phase monitors all job entries to conduct smooth on-server job handlings. In addition, the customer jobs are securely validated and distributed to different job queues using secure poisson distribution model. This helps to handle random job arrivals, and validate customer requests at the queuing points. The effective job queuing procedures lead the proposed design to build successive VPC policy mappings for arrived jobs in multi-server queues. In this phase, each job in a queue (level 1) is mapped with VPC policies and assigned a triple key (job code key, VPC key and session key). This authentic VPC management and secure policy management procedures ensure the validity between job and respective VPC policies. Under this VPC, the TDAG determines cloud job dependencies and order of executions on behalf of valid triple keys. This phase ensures that the cloud jobs' legitimate

☆ Bibliographical Sketch:**S. Rajasoundaran** is working as an Assistant Professor in the School of Computing Science and Engineering, VIT University, Bhopal, Madhya Pradesh, India. His research interests include computer networks, image processing, wireless networks and network security. **A.V. Prabu** is working as an Associate Professor in Electronics and Communication Engineering, Koneru Lakshmaiah Education Foundation (Deemed to be University), Vaddeswaram, Andhra Pradesh, India. His research interests include wireless sensor networks, wireless communication and IOT. **Sidheswar Routray** is working as an Associate Professor in Electronics and Communication Engineering, Koneru Lakshmaiah Education Foundation (Deemed to be University), Vaddeswaram, Andhra Pradesh, India. His research interests include speech processing, image processing, computer vision and wireless sensor networks. **Prince Priya Malla** is pursuing Ph.D. in School of Electronics Engineering, KIIT University, Bhubaneswar, India. Her research interests include image processing, computer vision and wireless sensor networks. **Amrit Mukherjee** is working as an Associate Professor in School of Electronics and Information Engineering, Anhui University, P.R. China. His research interests include wireless sensor networks, IOT and Wireless Communication.

* Corresponding authors.
E-mail addresses: sidheswar69@gmail.com (S. Routray), amrit1460@ieee.org (A. Mukherjee).

properties and truthfulness. Moreover, these valid jobs are departed in to the next level of multi-server queues (Level 2) and other are dropped from initial level queues. The validated jobs are located at Level 2 multi-queues of VPC server.

This proposed MLSCS extends the significant contribution for computing RL-Q matrix for acquiring triple keys of jobs, determining the deep Q values for each job, returning job rewards and predicting the job sequences using appropriate samples. This phase is associated with the jobs in Level 2 queues, TDAG transitions and the job sequences. The distributed cloud environment requires multi-level job validation procedures and dependency validation procedures for each VPC. This proposed system makes RL-Q matrix based learning technique as Deep Learning (DL) procedure. This technique helps to learn the legitimate job sequences and unauthorized job sequences using secure TDAG job transitions, job validity, job sequence patterns and reward based runtime actions. MLSCS is motivated to solve internal and external VPC security issues using both security techniques with the help of enhanced self-learning algorithms (Beach et al., 2019, Hrebicek and Chung, February 25, 2020). In this manner, this technique provides secure scheduling principles and job sequence learning opportunities.

At the end, the MLSCS implements VGAN for gathering natural job properties (genetic properties), job abnormalities and job seed errors (job producer details). This phase learns and isolates any kind of anomaly jobs under each VPC cluster. This phase finalizes that the authorized jobs are allowed in to job scheduler. This phase is particularly used for extracting natural cloud properties of any job, job originating sources and job irregularities. Consequently, the job scheduler initiates the scheduling tasks based on respective VPC policies assigned in distributed network environment. The complex nature of proposed MLSCS helps to construct more protected job analysis models and scheduling procedures under zero-trust VPC environment. In this regard, multi-level queues are collecting and departing the valid jobs to TDAG constructions, RL-Q evaluation procedures and VGAN procedures. The multi-level and sequential job evolution confirms that the ongoing scheduler events are legitimate. Additionally, this mechanism predicts the legitimate job sequences in future to reduce run time scheduler complexity. The technical details and implementation details of proposed MLSCS techniques are illustrated in section 3 and section 4 respectively.

This article is organized as follow from the analysis of related works and contribution of proposed system (Section 2). In section 3, the complete technical details of proposed MLSCS are described using appropriate algorithms. They are listed as secure queuing model development, RL-Q matrix, VGAN development, VPC authentication and secure scheduling procedures. In section 4, the proposed MLSCS is compared with recent existing techniques using various experimental parameters. The performance results are provided to show the efficiency of MLSCS over other systems.

2. Related works

This proposed system analyses different research works to identify scientific problems and implements MLSCS procedures (Selvi et al., 2021, Selvi et al., 2019). The works noted for technical analysis cover VPC management, cloud resource management, job scheduling techniques, cryptography techniques, and ML and DL techniques. Secured cloud scheduling is an important research challenge to be addressed in virtual private clouds. Liu et al (Liu and Jiang, 2018) discussed about ML based resource scheduling techniques in cloud computing environment. In this work, ML techniques such as regression, decision tree, support vector machine and Naïve Bayes are used for cloud resource scheduling procedures. In addition, this survey compared different types ML techniques under cloud environment.

Flavel et al. (Flavel et al., March 26, 2013) and Beach et al. (Beach et al., 2019) analyzed network connectivity and generic issues of VPC. The first work proposed an auto configuration technique with network connectivity issues. This method explained single line and dual line connectivity systems of VPC. The connectivity made VPC physically present in the cloud environment. The later work talked about VPC structure, user interactions, and problems with VPC communications, public cloud and security issues. These two works delivered common information and issues of VPC. In the same way, Hrebicek et al. (Hrebicek and Chung, February 25, 2020) explained about enterprise network functionalities and communication possibilities using VPC.

Sinha et al. (Sinha et al., June 11, 2013) and Mizik et al. (Mizik et al., July 24, 2018) provided vast domain concepts of distributed system principles and VPC's Domain Name System (DNS) functions. Wang et al. (Wang et al., 2017) and Farzi et al. (Farzi, 2009) analyzed the application of ML techniques in network communications. The later work described fish swarm techniques for effective job scheduling procedures. This technique worked better than generic scheduling approach conventionally. Many such works used intelligence in cloud scheduling were found in the literature (Thanikaivel et al., March 2021, Roshni Thanka et al., 2019). Boag et al. (Boag et al., 2017) proposed a novel technique for establishing multi-level reliable life cycle for DL preparation jobs. In their work, the life cycle of job handling procedures were customized with additionally included deep layers. This work increased learning rate of job handling programs. At the same time, it provided multi-level training to a platform to classify job properties. Yet, this work created more complexity and computation overhead in processor units.

Bhatia et al. (Bhatia et al., 2013) proposed a grid scheduling model based on secure priority provisions. This work enabled security practices such as secret keys and authentication procedures in conventional priority scheduling procedures. In addition, the secure mechanism was used to analyze requirements of various jobs of grid network. It achieved nominal security against basic types of attacks but lacks against multiple attacks. Orhean et al. (Sumathi and Poongodi, 2015) proposed new job scheduling technique using RL based rewards. The RL based rewards were given for individual jobs queued. This work analyzed the jobs of dissimilar distributed systems. In this distributed environment, the jobs coming from all nodes varied in their properties and arrival times. This work handled the challenges and provides distributed scheduling technique.

This work supported for the operation of heterogeneous networks effectively but lacked in security aspects. Gabi et al.

(Phung-Duc, 2019) proposed a cat swarm optimization technique for dynamic job scheduling in cloud environment. Angela et al. (Niño-Mora, 2019) implemented fuzzy models for attaining secure job scheduling in cloud computing platforms. Among these works, the later achieved both security and scheduling efficiency. Still, these works produced complexity at scheduler points.

The proposed MLSCS is intensely implementing both cloud security principles and secure cloud job scheduling procedures. In this regard, this proposed system finds notable existing techniques on multi-queue cloud job scheduling (Karthick et al., 2014) and multi-queue cloud job scheduling based on task classification procedures (Zuo et al., 2016). These two techniques were motivated to schedule the cloud jobs under distributed multi-server cloud environment.

Karthick et al. (Karthick et al., 2014) proposed cloud job scheduling and resource sharing benefits for improving the client fulfillment. In this work, the cloud jobs are gathered and scheduled using global scheduler. At this point, clustering techniques were used to classify the jobs based on their burst time. In this regard, the scheduling policies such as first come first served, shortest job first and others were implemented to reduce job starvation. However, burst time based clustering is conventional technique that is not effective under vulnerable and distributed cloud environment. Evaluating and exploring complete job characteristics (including security) are the essential needs of secure cloud scheduling framework. The proposed MLSCS work considers various job characteristics (burst time, priority, legitimacy, secret keys, etc.) to create job sequences under protected VPC cluster units. Additionally, this proposed work enriches the determination of job dependencies for the benefit of successful job executions. This reduces job failures, anonymous jobs and attacks on respective jobs.

Related to this contribution, Zuo et al. (Zuo et al., 2016) developed multi-queue peak job scheduling technique based on task classification procedures. In this work, cloud jobs were classified and scheduled. According to this framework, this work concentrated on CPU utilization, Input Output (I/O) utilization and memory utilization to manage the loads in the entire cloud. The system implemented classification techniques and scheduling techniques. This helped for reducing the response time and work load in distributed cloud network. Besides, this work never considered any security parameters and the vulnerability issues that are notable practical impacts. The distributed cloud environment cannot be implemented without providing security principles when the jobs are scheduled and executed. The nature of both security determinations and scheduling determinations help the entire cloud system work perfectly under vulnerable situations. However, the proposed work takes CPU overhead and security measures as significant metrics to evaluate the performance of proposed MLSCS techniques.

From the deep analysis of related research works, the proposed MLSCS is modeled to resolve secure job scheduling issues before the scheduler section. In addition, this proposed work uses light weight encryption and ML techniques to train the VPC platform for understanding legitimate jobs. In addition, the proposed MLSCS creates more simplified key management techniques and authentication techniques at queuing points. This approach reduces complexity in scheduler units.

The next section describes the procedural details of proposed MLSCS. The major limitations of the existing systems include the lack of security, increasing waiting time due to single job queue construction, non-grouping of jobs and lack of flow optimization. On the other hand, the proposed work increases VPC security by applying security constraints, key based security mechanism and hashing based security with job dependency features. The proposed work uses a rule based approach for making intelligent decisions and it uses genetic algorithms for search optimization. The major advantages of the proposed work over the existing scheduling algorithms include the facility for moving from one queue to another queue, intelligent allocation of jobs to the queues, genetic algorithm based search optimization with effective heuristic values. Finally, it gives the application of security constraints in the edges of the graph with migration to other virtual machines in order to enhance the security of the scheduling process.

3. Machine learning based secure cloud job services (MLSCS)

The proposed MLSCS is motivated to organize secure multi-server job scheduling tasks. At the same time, this system ensures protected VPC policies using highly encrypted credentials. VPC is a private cloud network, which allows authorized private users to access the resources. In addition, is provides separate public communities through another credentials. The user and system processes are initially called as jobs in cloud platform. These are maintained in various queues. In cloud system, many servers and clients are deployed in a distributed manner under various clusters (Suresh and Prasad, 2012). The queues or job queues help the scheduler to allot the slots for various jobs (Khan and Tuteja, 2015).

The proposed MLSCS involves in this complicated cloud services to ensure job authentication, data confidentiality, multi-server queue security and scheduling security at run time. To achieve this highly complicated job scheduling process, the following techniques are proposed.

- Multi-Sever Queuing Model Construction
- Authentic VPC Model Construction
- RL-Q Matrix Construction
- VGAN Construction

The four levels of construction models create end to end user security and system security. VPC has separate communication path to ensure safe resource management activities. Also, VPC maintains well-defined policy configurations. However, it has uncertain vulnerable points that can be breached by internal or external attackers (zero-trust environment). This can be resolved using trained ML network construction. The proposed MLSCS enriches job scheduling VPC policy protections through ML-based security layer deployment procedures.

3.1. Multi-server queuing models

Queuing models play important role in consolidating the jobs or instances coming in to any system. In VPC, the jobs and instances elevated from users must be regulated in ordered manner. VPC is equipped with multiple servers with multiple queues. Among those servers, a few are configured under one subnet and others are built under other subnets. However, the need for multi-server queues is inevitable. At the same time, the queues are expected to have parallel blocks (multi-channel or multiple queues in each server) to improve cloud process speed.

Definition A. *VPC consists of s homogeneous parallel servers and there is l Xw waiting lines (Multi-channel queue). Let s is the number of servers currently in active, then*

$$\rho^V = \frac{\tau^V}{\mu s} \tag{1}$$

ρ^V-*Utilization rate of each server in VPC*
τ^V-*Mean arrival rate of customer jobs for each queue*
μ- *Mean service rate of customer jobs for each queue*
s- *Number of VPC servers*

The definition states that the VPC consists of s parallel active servers with l server queues. In each server queue, l Xw customer jobs are waiting to be served. Eq. (1) depicts the queuing actions. So, the total number of active queues in VPC is determined as given in Eq. (2).

$$N\left[Q^{VPC}\right] = s\ X\ l \tag{2}$$

Let assume, $l \geq 4$, with the size of w. Fig. 1 illustrates the queuing scenario in one VPC server. From Fig. 1, let say there are 4 active queues ($l = 4$) with($w = 100$). The random job arrivals are determined by poisson distribution model,

$$P^{VPC} = e^{-\tau^V} \frac{(\tau^V)^N}{N!} \pm \theta \tag{3}$$

N-Number of arrivals
θ-Behaviour of customer arrivals (Balk or Renege) (Orhean et al., 2018)

Customer Job Key,

$$J^Q = \sum_{i=1}^{L} SHA_3^{512}(U^{IP}||U^{ID}\Big|\Big|U^{Ri}\Big|\Big|U^{TS}) \tag{4}$$

These customer jobs are distributed to multiple queues of each server of VPC if and only if they successfully passed their secret key validation process (Kashyap and Vidyarthi, 2013, Singh and Singh, 2016, Zitzler et al., 2000). The secret queue key, J^Q is computed from user login credentials at the time t^i as given in Eq. (4). Level 1 procedures indicate secure poisson distribution through multi-server queues. In this case, jobs and customer request are validated before queued. In addition, $P^{VPC}$is called (Eq. (4)) to distribute the validated jobs in to multi-server queues.

Fig. 1 illustrates jobs and VPC server interactions. There are multiple jobs initiated from customer point. The customers are interacting to this VPC server through secure communication channel. The jobs are getting placed in multiple queues of VPC server. Each VPC server contains varying number of queues. As given in Fig. 1, the jobs are stored in various queues based on secure poisson distribution model, Eq. (3).

Each job represented at each queue has customer job key, J^Q as given in Eq. (4). This helps to compute independent job key, J^Q for different customer accesses. At the same time, the job key plays significant role in secure job distribution.

Let assume the multi-server queuing model M / M / s, where, $> \tau^V$. The determination of multi-server queuing model with finite number of VPC customer jobs is M / M / s / ∞ / N at the rate of(s X l) for multi-channel and multi-server queues.

The VPC queuing model has multiple levels of queues to make security layering procedures. Unlike, conventional system, the proposed MLSCS maintains key enabled multi-server queues. These queues are trained to ensure job keys and other communication procedures. These strategies construct authentic VPC model with protected policies.

3.2. Authentic VPC Model

Definition B. VPC consists of secure shared resources for c customers with J^cjobs. VPC is conFig.d with virtual private policies for all customer jobs to enable VPN link for secure communication at specific job time t^j. As denoted, it has s active servers at the job time t^j. The queues in VPC servers are organized by the queuing model, M / M / s / ∞ / N. This model is customized with job dependent triple key block, T^K(128 Bits).VPC is a system that is logically separated and shared system resources, they are accessible only for valid users through secure communication path (Tunnel).

Level 2 describes the VPC validation procedures and Q^{L1} and Q^{L2} queue management tasks. In this level, both jobs and VPC requirements are mapped securely.

Triple Key, T^K is computed from Job Code Key, K^C, VPC Key, K^{VPC} and Session Key, K^T. This key is attached with each jobs in queues (running mode). Eqs. (5), (6) and (7) denote the computations of three different keys. In this regard, Eq. (5) denotes job code key, K^C computations and Eq. (6) provides VPC key, K^{VPC} computations. Both keys are generated using SHA functions. K^C is computed from time constraints and job key. At the same time, K^{VPC} is generated from nodes and VPN credentials for establishing more secure VPC environment against attackers.

Job Code Key,

$$K^C = \sum_{i=1}^{L} SHA_3^{512}(J^Q\Big|\Big|J^{AT}\Big|\Big|J^{WT}) \tag{5}$$

VPC Key,

$$K^{VPC} = \sum_{i=1}^{L} SHA_3^{512}(VPN^{IP}\Big|\Big|N^{Ri}\Big|\Big|VPC^{ID}||VPC^{TS}) \tag{6}$$

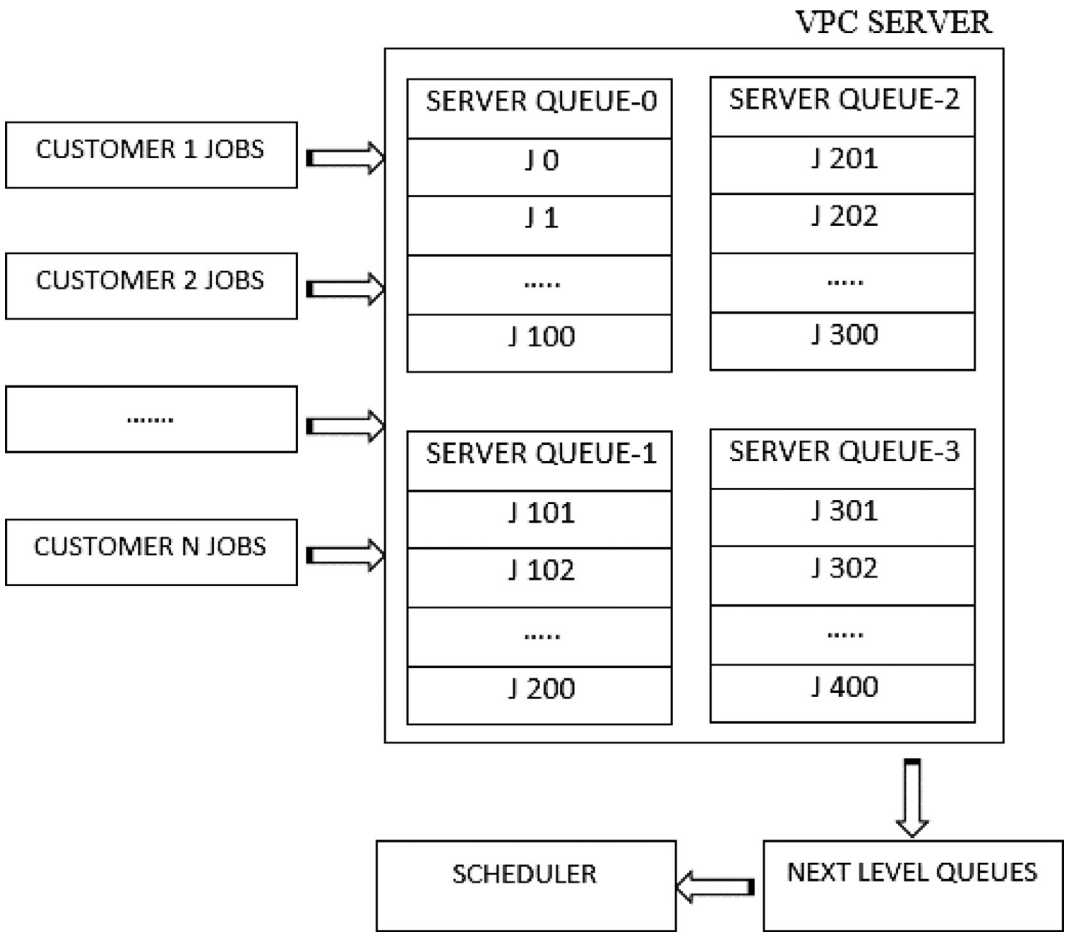

Fig. 1 – Multiple queues in a VPC server.

Session Key,

$$K^T = \sum_{i=1}^{L} SHA_3^{512}(IPSEC^S||CP^K||T||N) \quad (7)$$

In Eq. (5), SHA-3 code is generated as K^C (512 Bits). This key is generated using J^Q, J^{AT}- Job Arrival Time (milliseconds) and J^{WT}- Job Waiting Time (milliseconds) in queue. In this Eq., L is the maximum job arrivals at time't'. In the same way, Eq. (6) creates 512 Bits of K^{VPC} using VPN^{IP}-Internet Protocol Address of VPN, N^{Ri}-Network Request Identifier, VPC^{ID}-VPC identifier and VPC^{TS}-VPC Timestamp values. The session key, K^T is created by IP Security Protocol, CP^K-Pair Key between communicating parties, T- Timestamp and N-Nonce Eq. (7). Once these three keys are generated, the secure triple key is generated that to be attached with all authenticated jobs and batches. This triple key, T^K is used for proposed RL Q Matrix Computations and VGAN scheduling tasks.

In addition, determining job and batch dependencies is important to make consistent execution. The scheduler events can be started only after the job dependencies are successfully mapped. For this operation, T^K enabled Secure Direct Acyclic Graphs (TDAG) are constructed. The dynamic construction of TDAG helps to keep job consistency before scheduling starts.

Fig. 2 illustrates TDAG structure for job dependencies and the flow of job states from one job to another job. The DAG is graphical method that can be used for constructing the job dependency system in any distributed computing environment. In this proposed system, the secure key based DAG is implemented for controlling the job dependency principles under various job keys. This ensures the validity and truthfulness of initiated jobs. Moreover, this secure DAG is responsible for enabling controlled job dependency environment under vulnerable cloud conditions. In this TDAG, each node has valid T^K and linked with another node's T^K. This is represented as triple key pair as given in Fig. 2. Likewise, each node in TDAG represents VPC jobs and job identifier pointers.

3.3. *Secure RL Q matrix construction*

The RL Q matrix construction is a type of DL procedure. The matrix is constructed with numerous Q cells mean individual VPC job. Every Q matrix section jobs are analyzed in all aspects to provide Q values at the final stage. In this proposed MLSCS, each Q section is secured with T^K and T for representing dynamic allocation of VPC jobs. Level 3 procedures given below describe Secure RL Q matrix computations and job validations (Gabi, 2020, Wilczyński and Kołodziej, 2020).

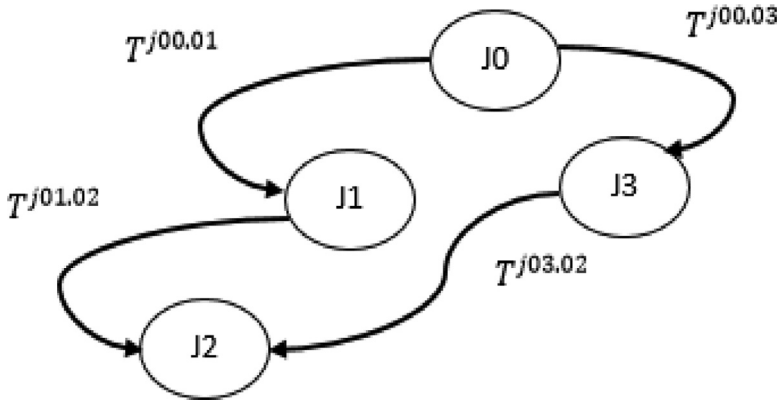

Fig. 2 – Key enabled secure DAG.

This section of MLSCS constructs secure Q matrix cells to denote key enabled jobs. In addition, this secure mechanism delivers protected Q values for each authenticated job. The values given for each job valid for one time. Then the Q value is stored for predicting next job sequences and next Q values. Eq. (8) predicts upcoming job sequence's Q values, $DJ_{i(Q+1)}(J_i, J_\tau)$ from currently allocated Q value, $DJ_{(Q)}(J_i, J_\tau)$. J_i-Job, J_τ-Time. The parameter $\varphi(J_i, J_\tau)$ represents bias in prediction, $max^{J_i}$and min^{JL_i} denote maximum and minimum RL weights given for jobs. In addition, RL_O is RL job reward.

$$DJ_{i(Q+1)}(J_i, J_\tau) = DJ_{(Q)}(J_i, J_\tau) + \varphi(J_i, J_\tau) \times \left\{\left(max^{J_i} + min^{JL_i}\right) + RL_O\right\} \tag{8}$$

The RL_O expression is given in Eq. (9). It is computed using job learning rate, ∂. At the same time, it uses predicted RL reward of current job, $RLW_{(Q+1)}$ and current Q value of job. The exact RL Q value for each job is calculated as given in Eq. (10).

$$RL_O = RLW_{(Q+1)} \pm \partial - DJ_{(Q)}(J_i, J_\tau) \tag{9}$$

$$YQ(J_i) = \sum_{i=1}^{n} f(RL_O \pm \theta) \tag{10}$$

In Eq. (10), YQ(J) denotes RL Q cell value of particular job, J_i. The Q matrix has nxn in size. The Q value is calculated fromRL_O. In this case, θ depicts mean error. Also, the matrix can maintain the Q value with job states, events and triple keys as represented in Eq. (11).

$$J_{(Q)} = Q_0\left(Job\ States,\ Events,\ T^K\right) \tag{11}$$

The appropriate Q value of a job, J_i is given to VGAN construction procedures. This helps to activate completely authenticated scheduler events. This is given in section 3.4.

3.4. *VGAN construction for secure scheduling*

The proposed MLSCS analyses each job in all aspects before it is delivered to job scheduler. In this regard, Level 4 procedures ensure job's genetic properties, batch (job) abnormalities and job producer error. Technically VGAN construction is achieved by analyzing seed process of job producer, $SG^I$and the criticizer value, C^I. C^I is calculated from VPC job policies and scheduling policies. The secure criticizer is given in Eq. (12).

$$C^I = AES^{ENC}(A^{VPOL}||T^K\Big|\Big|A^{SPOL}\Big|\Big|A^T) \tag{12}$$

In this Eq., AES is used to encrypt the criticizer attributes (AES^{ENC}). The attributes such as, Job's VPC policy pointer, A^{VPOL}, T^K, Job's Scheduling policy pointer, A^{SPOL} and Job's arrival time, A^T are given. The Multi-Level GA (MGA) procedure executes following job analysis equations. Eq. (13) denotes Job population at different multi-server queues. J^P is job population, Rd^N is random number, α is populace rate (Angela Jennifa Sujana et al., 2020, You et al., 2020).

$$J^P = \sum_{i=1}^{n^J} f\left(DJ_i\left(Rd^N\right).\alpha.(s\ X\ l\right) \tag{13}$$

Eqs. (14) and (15) represent multi-server job evaluation function, J^E and crossover (dependency) function, J^C respectively. In Eq. (14), $J\omega$ delivers job's genetic weight. In Eq. (15), BJ^i is VPC job batch. This Eq. shows TDAG parameters and dependent job properties for understanding relative executions.

$$J^E = \sum_{i=1}^{n^J} f\left(DJ_i\left(Rd^N\right).\alpha.\ RL_O J\omega\right) \tag{14}$$

$$J^C = \sum_{i=1}^{n^J} f^{CR}\left(BJ^i(\alpha.\ RL_O J\omega)\right) \tag{15}$$

$$J(S^R) = \sum_{i=1}^{n^J} f(DJ.\alpha)\ \sim BJ^i \tag{16}$$

$$J(S^{R*}) = \sum_{i=1}^{n^J} f(DJ(-1).\alpha)\ \sim BJ^{i-1} \tag{17}$$

This VGAN identifies jobs and batches using TDAG representations Wilczyński and Kołodziej, 2020). TDAG is reconstructed every time for new jobs entered in to Q^{L1} and Q^{L2}.

For every job, VGAN determine seed process of job initialization and criticizer. Mainly, VGAN deals with these jobs with the help of genetic properties, GJ^P and other level 3 determinations. Based on these attributes, gradient descent is noted for jobs and batches. Eqs. (16) and ((17) give the details of actual and preceding job sample functions, $J(S^R)$ and $J(S^{R*})$ respectively.

In the same manner, Eqs. (18) and (19) provide discriminative abnormality, $J(D^\theta)$ and job producer process error, J^θ. These Eqs. are used to analyze and extract the run time properties of VPC jobs. Once the properties are validated, the level queues are set.

$$J(D^\theta) = -\frac{1}{2n}\left(\sum_i^n J(RL_0).\log BJ^i + \sum_i^n 1 - (J_i.\log BJ^{i-1})\right) \tag{18}$$

$$J^\theta = -\frac{1}{n}\left(\sum_i^n 1 - (J.\log BJ^{i-1})\right).GJ^P \tag{19}$$

For every job that is passing the validations, the following keys are attached for secure job scheduling process. Each job has a job scheduler key, A^{SCHED} and batch scheduler key, B^{SCHED}. Eqs. (20) and -(21) show the secure scheduler key computations for job and batch respectively.

$$A^{SCHED} = AES^{ENC}(\,RL_0||T^K||A^T) \tag{20}$$

$$B^{SCHED} = AES^{ENC}(A^{SCHED}||T^K\Big|\Big|A^T\Big|\Big|C^I) \tag{21}$$

The jobs evolved in VPC groups have to pass all levels of proposed MLSCS to ensure their identity, property, dependency and authenticated policies at run time. In all the way, proposed MLSCS has given more secure cloud services using these ML and DL techniques (Thanka et al., 2019). The proposed MLSCS is implemented and compared as given in implementation section.

4. Experimental setup and results

The proposed MLSCS environment is created using CloudSim 3.0. This simulation environment produces 150 VPC servers with different types of resources. Each VPC server maintains its private group through secure link. Every VPC local network group has own identity and security policy certificates to communicate with each other. A member in VPC group can communicate within the group and outside the group. The jobs and instances requested from various members or customers are handled by server resources. In addition, the servers are implemented using VMs to activate cloud amenities (Muthurajkumar et al., 2018, Gunasekaran et al., 2016). A particular VM can be activated on demand basis to handle job instances. Table 1 depicts the proposed MLSCS cloud environment.

Table 1 gives the details of VPC servers and VMs used for job executions. It provides number of Central Processing Units (CPUs), physical memory sizes and processor speed in Million Instructions per Second (MIPS). The implemented VPC environment has around 16 CPUs in each server. Each server is equally conFig.d with appropriate processor speed and other parameters. Table 1 indicates the configuration of both VPC servers and client VMs. This configuration table shows maximum processor speed, number of CPUs and physical memory size of both server and other VMs. The considerations of the illustrated configuration parameter are not homogeneous under the proposed cloud environment. This shows the potential impact of real VPC environment that has variety of VMs, varying job arrival rates and customer participations. This system is implemented with an assumption of heterogeneous cloud environment that has dynamic resource allocations and dynamic traffic rates. In this regard, the VMs conFig.d under various servers manage the cloud jobs on demand basis. Particularly this approach works based on available cloud resources and the needs.

In this implementation, jobs are identified from the online cloud job datasets (Google public datasets and Trifacta Job datasets). These datasets are taken in to attention for finding job initialization, job dependencies, job completeness, job failures and missing job data features. The job initialization function helps to initialize the jobs in to the implemented cloud environment. In the experimental setup, both proposed and existing techniques are evaluated using these job datasets and job sequences. These jobs are completely heterogeneous that are suitable for dynamic VPC implementation procedures.

Under this VPC environment, traffic rate is variable to each VPC cluster. Therefore, configuring variable traffic rate for different cloud clusters is an essential task that is carried out in this testbed to ensure the scalability of the network. As the developed VPC environment targets the commercial cloud environment, the number of cloud participants is not determined (scalability issue). Consequently, the network traffic rate is not defined as constant rate in a given time. The proposed VPC network is conFig.d with the traffic rate of 1 Mbps as minimum requirement. This rate is variable based on the number of VPC customer jobs. In the proposed scenario, normal rate of customer or participants is 500 to 600 in each VPC cluster. In this regard, traffic rate is completely dynamic in nature. To meet the scalability requirements, the proposed system works with the variable network traffic rate. This is defined as the traffic rate which is proportional to the number of customers.

The proposed MLSCS is experimented under this VPC cloud environment and technically compared with different existing works. The performances of existing research works like RLDH, CSOS and FSDS are illustrated in this section. RLDH is implemented for distributed scheduling scenario. In this case, the network is considered as heterogeneous distributed nodes. On conventional distributed scheduling technique, RL is applied for reliable and effective job scheduling process. As RL is a reward producing technique, it gives reward to each distributed system job based on job properties. According to this mechanism, jobs are scheduled by distributed job scheduler program. However, this work has lack of security and multilevel job analysis issues. This leads to runtime issues in distributed systems (Sumathi and Poongodi, 2015).

CSOS and FSDS are soft computing based techniques applied for optimized scheduling processes. In these two techniques, CSOS achieved better optimization by reducing aver-

Table 1 – MLSCS testbed environment.

Resources	Total Number of Servers	Processor Speed (MIPS)	Total Number of CPUs	Physical Memory Size (MB)
VPC Servers	150	2650	16	50256
VMs	On Need Basis	1950	2	1150
		2100	2	1800
		600	1	2280
		1200	2	1560
		1500	2	2100

Level 1 – Secure poisson distribution.

Input: *User Identifiers and Jobs*
Output: *Jobs at Multiple Queues*
Begin
Step 1: *Get user requests, USER_REQ* (R^{ID}, U^{Ri}, U^{ID})
R^{ID}*-User Request,* U^{Ri}*- User Request Identifier,* U^{ID}*-Respective User Identifier*
U^{TS}*-User Timestamp*
Step 2: *Compute Customer Job Key,* J^{Q}.
Step 3: *Call procedure* P^{VPC} *for a job* J^{i}, *if* (J^{Q} *is valid)*
Step 4: *Repeat* P^{VPC} *for s X l queues (Level-0).*
End

Level 2 – VPC key validation model.

Input: *Level-1 Jobs and* J^{Q}
Output: *Triple Key attached jobs*
Begin
Step 1: *Get validated user requests, USER_REQ* (R^{ID}, U^{Ri}, U^{ID}, J^{Q})
Step 2: *Call Procedure GET_JOB* (Q^{L1}), Q^{L1}*- Level 1 Job Multi-server Queues*
Step 3: *Call GET_VPCC* (Q^{VPCC}). *VPCC- VPC configuration details and* Q^{VPCC}*-Multi-Server Queues for VPCC management.*
Step 4: *Do security validations for all VPC customer jobs*
If ((Q^{L1}(R^{ID}, U^{Ri}, U^{ID}) && Q^{VPCC}(U^{ID}))=*TRUE)*
Push (Q^{L1}) *entries with* (Q^{L2}), Q^{L2}*- Multi-server Queues with job keys*
Else
Detach (Q^{L1}) *entry*
Set Invalid label for (Q^{L1}) *entry*
Step 5: *Set* Q^{L2}(K^{C}, K^{VPC}, K^{T}) *as Triple Key,* T^{K}, Eqs. (5), (6) *and* (7)
Step 6: *Call Procedure KEYGEN () for each iteration*
Step 7: *Call Procedure GET_JOB* (Q^{L2}),
End

Level 3 – Key attached RL Q matrix for VPC job validation

Input: *Job description samples and Keys*
Output: *Protected Q Values for Jobs*
Begin
Step 1: *Define Job descriptor,* $DJ_{iL} = n,\ n > 1$
Step 2: *Set deep dependent RLQ matrices,* $Q^{J} = \{Q^{1}, Q^{2}, \ldots .Q^{n}\}$
Step 3: *Determine unit memory magnitude for all RL Q cells.*
Step 4: *Calculate RL manager function for dependent job validations,* Eqs. (8) *and* (9).
Step 5: *Trigger RL Q manager function for all cloud jobs synchronously*
Step 6: *Attach Triple keys at each Q cell (Job)*
Step 7: *Determine Job description samples,* J_S*(events) then activate RL reward function,* RL_O *at* τ
Step 8: *Collect* J_S *and* τ *for all event states,* τjs_i.
Step 9: *Determine RL Q network output function (Q-Value),* Eqs. (8) *and* (9).
Step 10: *Execute this RL Q network functions for all cloud servers.*
Step 11: *Call TDAG for dependency verification*
Step 12: *Repeat*
End

Level 4 – VGAN for secure scheduling.

Required: $YQ(J_i)$, RL_0, ∂, *Genetic Job properties,* GJ^{P}
Input: *Seed producer,* SG^{I}*and Criticizer,* C^{I}
Output: *Protected Job Scheduling*
Begin
Step 1: *Call MGA Procedure (Job population, evaluation and crossover) Eqs. (13), (14) and (15).*
Step 2: *Check Job Scheduler in active*
Step 2: *Set seed producer function,* SG^{I}*for reading job seeds*
Step 3: *Define actual and preceding job sample functions,* Eqs. (16) *and* (17).
Step 4: *Activate these function for all cloud server jobs located in queues*
Step 5: *Identify Jobs and Batches*
Step 6: *Define discriminative abnormality,* Eqs. (18)
Step 7: *Calculate model producer error,* Eqs. (19)
Step 8: *Determine gradient descent for job batches*
Step 9: *Set* Q^{L1} *and* Q^{L2}.
Step 10: *Call JOB_SCHED* (Q^{L2})
Step 11: *Determine JOB_SCHED_POL(RR,PRI,FCFS, SJF, MQS)*
RR-Round Robin, PRI -Priority, FCFS-First Come First Serve, SJF-Shortest Job First,
MQS-Multi-Queue Scheduling.
Step 12: *Determine BATCH_SCHED_POL(RR,PRI,FCFS, SJF, MQS)*
Step 13: *Do Scheduling for all jobs and batches and Set Criticizer,* C^{I}.
End

age waiting time of processes. Nonetheless, this scheme has no assumptions on security issues in shared cloud environments (Phung-Duc, 2019). At the same time, FSDS provides solution for both scheduling and security issues. In cloud environment, FSDS creates fuzzy rule engine (Nancy et al., 2020, Selvi et al., 2019, Selvi et al., 2019) and reference system for validating jobs efficiently than CSOS (Niño-Mora, 2019). Nevertheless, these works are not accomplishing stacked security provisions and ML techniques together in scheduling process. Fig. 3 offers the particulars of VPC customers or participants. They are allowed securely to interact with VPC servers. There are five VPC clusters shown in Fig. 3 with different number

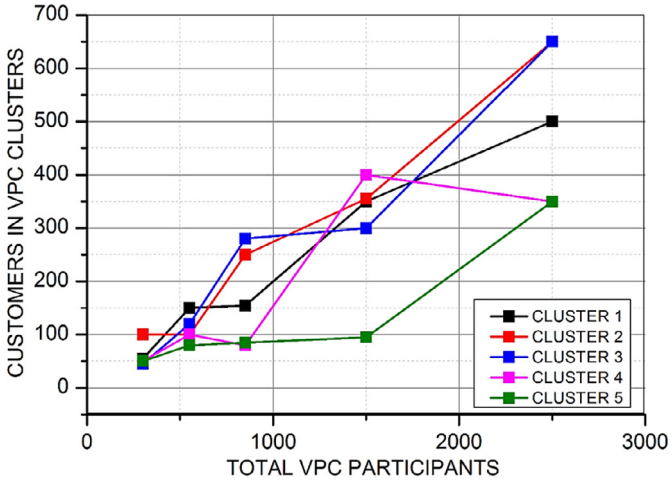

Fig. 3 – VPC clusters and participants.

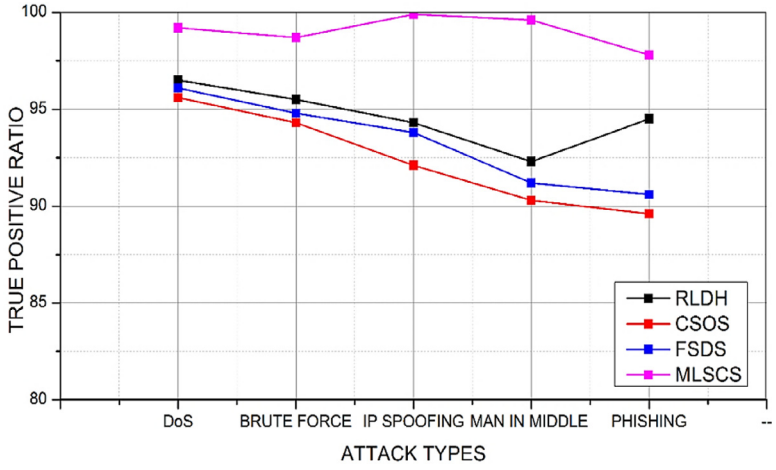

Fig. 4 – Attack detection on VPC.

of customers. The total number of customers in VPC is 2500 at particular time. It varies at random intervals. In this Fig., cluster 1 and cluster 3 show maximum VPC participants as 650. In this proposed work, VPC has dynamically changing cloud participants and their activities. The VPC participants are grouped under different clusters. The clusters are generated for maintaining the participants based on their resource access policies. In this regard, Fig. 3 illustrates the heterogeneous nature of newly developed VPC model.

Fig. 4 illustrates the attack circumstances and the attack detection rates for various VPC attacks. Particularly, VPC is managing service based attacks and identity based attacks using proposed techniques. Fig. 4 and 5 illustrate the performance comparison between the existing and the proposed MLSCS scheme.

Fig. 4 gives true positive rate (%) against Denial of Service (DoS) attack, brute force attack, IP spoofing attack, man in middle attack and phishing attack. Each attack is injected randomly in to VPC from internal nodes (Gąsior and Seredyński, 2019, Arunarani et al., 2019). In this comparison, the existing schemes RLDH, CSOS and FSDS provide around 95% to 96% of true positive rate which is lesser than the proposed MLSCS. MLSCS uses more layers of security credential verification procedures for jobs enter into server queues than other works (Liu and Kang, 2019). MLSCS detects IP spoofing attack as maximum (99.9%) than other attacks.

Fig. 5 provides job scheduler complexity rate (%) against varying job dependency rate in VPC systems. The job scheduler complexity rate deals with the time complexity and space complexity rates for various scheduling tasks. In addition, the scheduler directly depends upon the dependency rates of each job initiated in VPC clusters. The cloud based jobs are directly or indirectly related to each other that create various consequences in process execution (Thanikaivel et al., March 2021, Liu et al., 2021, Konjaang and Xu, 2021). Job dependency is a major criticizer in scheduling task and process management. As job dependency increases, scheduler complexity increases in existing system up to 80%.

In the proposed MLSCS, jobs are pre-validated using TDAG before they reach job scheduler. In this manner, the scheduler complexity of proposed MLSCS mainatains between 55% and 65% on dependency critics.

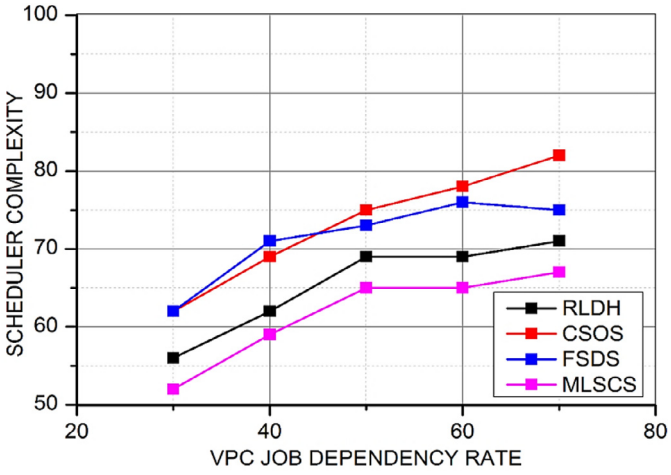

Fig. 5 – Scheduler complexity rate.

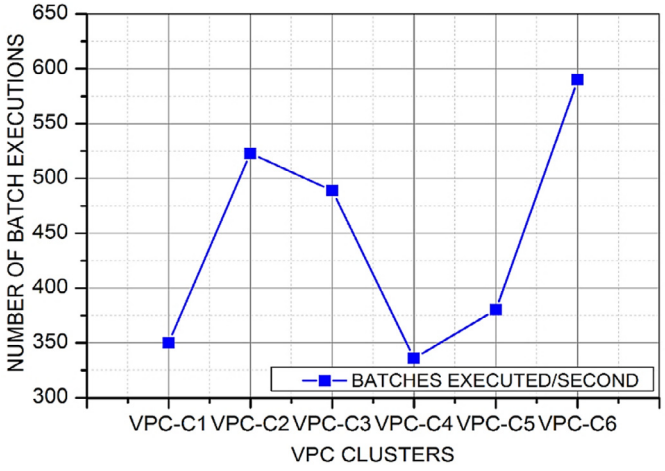

Fig. 6 – VPCs and batch executions.

Fig. 6 delivers the detailed view of VPC batch execution rate with respect to clusters. In each cluster, batch executions vary depends on VPC customer requauest. The batch executions indirectly denote job dependency identification. The scheduler do batch sequaences for CPU execution.

Fig. 7 compares CPU utilization rate of existing and proposed MLSCS with respect to various server units. The number of server units varies from 20 to 150 gradually to calculate overall CPU utilization rates. In this analysis, the proposed MLSCS attains only 0.65 to 0.67 CPU utilization rate compared to other works. In proposed system, unauthorized actions and attack raising jobs are effectively eliminated before they stretch in to scheduler or CPU sections (Xu et al., 2020, Roshni Thanka et al., 2019, Selvi et al., 2021). Consequently, the elimination of insecure events gives optimal CPU utility than other existing systems. In this scenario, VPC has 150 server units for managing cloud participants. The existing techniques such as RLDH, CSOS and FSDS are producing the CPU utilization rates from 0.72 to 0.85 that affects the overall computing performance of VPC environment.

CPU overhead can be termed as additional CPU utilization rate or additional time taken by CPU to complete the process execution. In this regard, generic CPU utilization rate samples (time) are determined to find the overhead. This can be determined in terms of absolute time or MIPS. However, in CloudSim (simulation), this proposed MLSCS creates multiple VMs for managing the jobs. The simulation tool offers special methods (cloudlet) to monitor the CPU utilization rate in terms of MIPS or scale ratings (0 to 1).

The methods used in this testbed are associated to particular VMs which are useful for calculating the additional CPU utilization.

$$CPU\ Overhead = 1 - \frac{CPU\ utilization\ of\ VM\ at\ time't'(MIPS)}{Determined\ Sample\ CPU\ Utilization\ of\ VM\ (MIPS)} \quad (22)$$

In the implementation, each VM is conFig.d with different processor (CPU) speed. According to the configuration, the maximum MIPS rate is denoted in table 1. Each VM is taking the job executions on demand basis with their allotted speed rates (maximum). However, VMs work with various MIPS rates depends on the arrival rates of jobs. In this regard, CPU or processor overhead is calculated on the basis of the formula given in Eq. (22). Eq. (22) determines CPU overhead from the

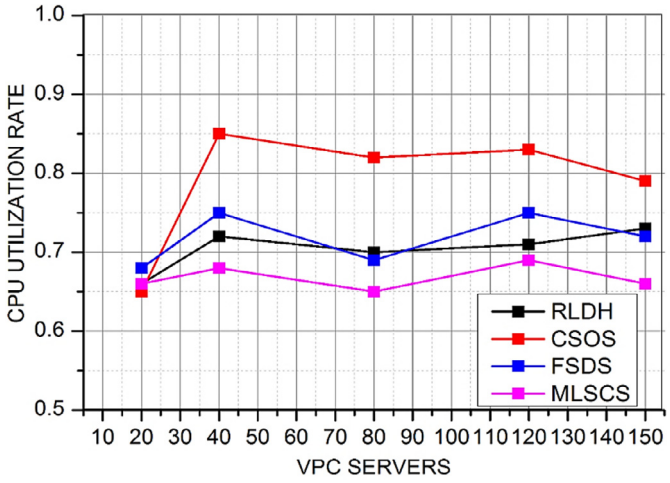

Fig. 7 – VPC CPU utilization.

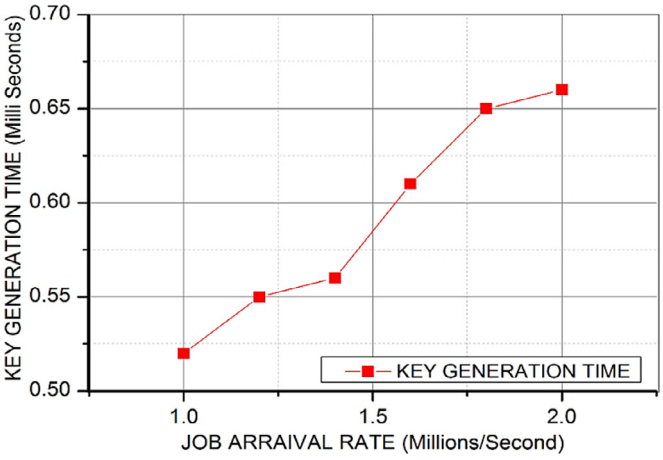

Fig. 8 – Key generation time.

determined sample utilization of particular VM in the allotted duration and the CPU utilization of the same VM at time 't'. The time 't' denotes the moment when the proposed secure scheduling algorithms are initiated for scheduling the jobs. In this situation, the overhead is not persistent but it is related to the number of cloud participants and jobs.

Comparing to other existing systems, the proposed MLSCS produces 10% to 15% of minimal CPU overhead. This is a significant contribution for improving the performance of distributed computing platform. As security is an important factor in MLSCS, finding key generation efficiency is also very important task (Rajendran et al., 2019, Velumadhava et al., 2019, Seenuvasan et al., 2017). Fig. 8 gives relationship between overall key generation time and VPC job arrival rate. It gives little overhead before scheduling process.

However, this is not a critical issue against the need for security reasons. The performance evaluation illustrated above expresses the expanded comparison between the proposed MLSCS and the existing works using various experimental parameters. This work enhances the security benefits in scheduler jobs with optimal overhead.

5. Conclusion

In this proposed MLSCS, secure job scheduling was achieved in zero-trust VPC environment. In order to achieve effective security features in scheduling process, MLSCS implemented multi-server queuing model, authentic VPC model, RL-Q matrix model and VGAN analysis model. These proposed techniques supported end to end VPC protection from various attacks and misbehaviors of any insecure activities. The activities and job events were clearly validated and authenticated after multi-level security analysis. The proposed MLSCS delivered better performance than other compared schemes in various aspects. Finally, it was noted that the proposed system can be used for assisting the scheduler to do secure job scheduling in VPC interactions. However, this method would create little more computation overhead in key generation and encryption phases when more jobs arrived. This work can be extended to resolve these computational issues.

Credit

The individual author's credit are as follows:

S. Rajasoundaran: Mathematical modelling and analysis

A.V. Prabu: Simulation and comparisons

Sidheswar Routray: Mathematical analysis and proof reading

S.V.N. Santhosh Kumar: Simulation and comparisons

Prince Priya Malla: Mathematical modelling and analysis, paper drafting and writing

Suman Maloji: Mathematical modelling and analysis

Amrit Mukherjee: Pseudo-code development and analysis

Uttam Ghosh: Simulation and experimentation; paper drafting and writing

Declaration of Competing Interest

We, the authors state that our article has not been published elsewhere and that it has not been submitted simultaneously for publication elsewhere.

There is no conflict of interest associate with the submission.

REFERNCES

Angela Jennifa Sujana J, Revathi T, Joshua Rajanayagam S. Fuzzy-based security-driven optimistic scheduling of scientific workflows in cloud computing. IETE J. Res. 2020;66(2):224–41.

Arunarani AR, Manjula D, Sugumaran Vijayan. Task scheduling techniques in cloud computing: a literature survey. Future Generat. Comput. Syst. 2019;91:407–15.

Beach Brian, Armentrout Steven, Bozo Rodney, Tsouris Emmanuel. Virtual private cloud. In: In Pro Powershell for Amazon Web Services. Berkeley, CA: Apress; 2019. p. 85–115.

Bhatia Manjot, Kumar Muttoo Sunil, Bhatia Mahinder Pal Singh. Secure requirement prioritized grid scheduling model. IJ Network Secur. 2013;15(6):478–83.

Boag Scott, Dube Parijat, Herta Benjamin, Hummer Waldemar, Ishakian Vatche, K JAYARAM, Kalantar Michael, Muthusamy Vinod, Nagpurkar Priya, Rosenberg Florian. In: Workshop on ML Systems. Scalable multi-framework multi-tenant lifecycle management of deep learning training jobs. NIPS; 2017.

Farzi Saeed. Efficient job scheduling in grid computing with modified artificial fish swarm algorithm. Int. J. Comput. Theory Eng. 2009;1(1):13.

Flavel Ashley, Lund Carsten, Nguyen Han. Network connectivity wizard to support automated creation of customized configurations for virtual private cloud computing networks. U.S. Patent March 26, 2013;8(323) 407issued.

Gąsior Jakub, Seredyński Franciszek. Security-aware distributed job scheduling in cloud computing systems: a game-theoretic cellular automata-based approach. In: International Conference on Computational Science. Cham: Springer; 2019. p. 449–62.

Gabi Danlami. Hybrid cat swarm optimization and simulated annealing for dynamic task scheduling on cloud computing environment. J. Inf. Commun. Technol. 2020;17(3):435–67.

Gunasekaran S, Sai Ramesh L, Sabena S, Selvakumar K, Ganapathy Sannasi, Kannan Arputharaj. Dynamic Scheduling Algorithm for Reducing Start Time in Hadoop. ICIA; 2016. p. 123.

Hrebicek Ondrej, Chung Leonard. Virtual private cloud that provides enterprise grade functionality and compliance. U.S. Patent February 25, 2020;10(453) 572issued.

Karthick AV, Ramaraj E, Subramanian RGanapathy. An efficient multi queue job scheduling for cloud computing. In: 2014 World Congress on Computing and Communication Technologies. IEEE; 2014. p. 164–6.

Kashyap Rekha, Vidyarthi Deo Prakash. Security driven scheduling model for computational grid using NSGA-II. J. Grid Comput. 2013;11(4):721–34.

Khan Shakeeba S, Tuteja RR. Security in cloud computing using cryptographic algorithms. Int. J. Innov. Res. Comput. Commun. Eng. 2015;3(1):148–55.

Konjaang JK, Xu L. Multi-objective workflow optimization strategy (MOWOS) for cloud computing. J. Cloud Comput. 2021;10(11). doi:10.1186/s13677-020-00219-1.

Liu Qi, Jiang YingHang. A survey of machine learning-based resource scheduling algorithms in cloud computing environment. In: International Conference on Cloud Computing and Security. Cham: Springer; 2018. p. 243–52.

Liu Jingchen, Kang Hyeon-Ah. Q-matrix learning via latent variable selection and identifiability. In: Handbook of Diagnostic Classification Models. Cham: Springer; 2019. p. 247–63.

Liu Z, Zhao A, Liang MA. Port-based forwarding load-balancing scheduling approach for cloud datacenter networks. J. Cloud Comping 2021;10(13). doi:10.1186/s13677-021-00226-w.

Mizik Andrey, Zen Lee-Ming, McCullagh Gavin Derek, Santoso Yohanes, Meleshuk Vadim, Gu Yu, Lai Minli, et al. Adaptive resolution of domain name requests in virtual private cloud network environments. U.S. Patent July 24, 2018;10(691) 033issued.

Muthurajkumar S, Vijayalakshmi M, Ganapathy S, Kannan A. Optimal and energy efficient scheduling techniques for resource management in public cloud networks. In: National Academy Science Letters. Springer; 2018. p. 219–23 vol 41.

Nancy Periasamy, Muthurajkumar Sannasy, Ganapathy Sannasi, Santhosh Kumar SVN, Selvi M, Arputharaj Kannan. Intrusion detection using dynamic feature selection and fuzzy temporal decision tree classification for wireless sensor networks. IET Commun. 2020;14(5):88–895.

Niño-Mora José. Resource allocation and routing in parallel multi-server queues with abandonments for cloud profit maximization. Comput. Operat. Res. 2019;103:221–36.

Orhean Alexandru Iulian, Pop Florin, Raicu Ioan. New scheduling approach using reinforcement learning for heterogeneous distributed systems. J. Parallel Distrib. Comput. 2018;117:292–302.

Phung-Duc, Tuan. "Retrial queueing models: a survey on theory and applications." arXiv preprint arXiv:1906.09560 (2019).

Rajendran Rakesh, Kumar SVNSanthosh, Palanichamy Yogesh, Arputharaj Kannan. Detection of DoS attacks in cloud networks using intelligent rule based classification system. Cluster Comput. 2019;22(1):423–34 Suppl.

Roshni Thanka M, Uma Maheswari P, Bijolin Edwin E. An improved efficient: artificial Bee Colony algorithm for security and QoS aware scheduling in cloud computing environment. In: Cluster Computing, 22. springer; 2019. p. 10905–13.

Seenuvasan P, Kannan A, Varalakshmi P. Agent-based resource management in a cloud environment. Appl. Math. Inf. Sci. 2017;11(3):777–88.

Selvi M, Velvizhy P, Ganapathy Sannasi, Khanna Nehemiah H, Kannan Arputharaj. A rule based delay constrained energy efficient routing technique for wireless sensor networks. Cluster Comput. 2019;22:10839–48.

Selvi M, Thangaramya K, Sannasi Ganapathy, Kulothungan K, Khannah Nehemiah H, Kannan A. An energy aware trust based secure routing algorithm for effective communication in wireless sensor networks. Wireless Personal Commun. 2019;105:1475–90.

Selvi Munuswamy, Kumar SVN Santhosh, Ganapathy Sannasi, Ayyanar Ayyasamy. Harichandran Khanna Nehemiah, Arputharaj Kannan," An energy efficient clustered gravitational and fuzzy based routing algorithm in WSNs. Wireless Personal Commun. 2021;116:61–90.

Singh Lovejit, Singh Sarbjeet. Score-based genetic algorithm for scheduling workflow applications in clouds. Int. J. Grid Util. Comput. 2016;7(4):272–84.

Sinha Amit, Devarajan Srikanth, Foxhoven Patrick. Distributed, multi-tenant virtual private network cloud systems and methods for mobile security and policy enforcement. U.S. Patent June 11, 2013;8(335) 464issued.

Sumathi D, Poongodi P. An improved scheduling strategy in cloud using trust based mechanism. Int. J. Comput. Electr. Autom. Control Inf. Eng. 2015;9(2):637–41.

Suresh KS, Prasad KV. Security issues and security algorithms in cloud computing. Internat. J. Adv. Res. Comput. Sci. Software Eng. 2012;2(10).

Thanikaivel B, Venkatalakshmi K, Kannan A. Optimized mobile cloud resource discovery architecture based on dynamic cognitive and intelligent technique. Microprocess. Microsyst. March 2021;81:1–11 103716.

Thanka MRoshni, Uma Maheswari P, Bijolin Edwin E. An improved efficient: Artificial Bee Colony algorithm for security and QoS aware scheduling in cloud computing environment. Cluster Comput. 2019;22(5):10905–13.

Velumadhava Rajasekaran, Selvamani Kadirvelu, Kanimozhi Sakthivel, Arputharaj Kannan. Hierarchical group key management for secure data sharing in a cloud-based environment. Concurrency Computation - Practice Experience 2019;31(12).

Wang Mowei, Cui Yong, Wang Xin, Xiao Shihan, Jiang Junchen. Machine learning for networking: Workflow, advances and opportunities. IEEE Network 2017;32(2):92–9.

Wilczyński Andrzej, Kołodziej Joanna. Modelling and simulation of security-aware task scheduling in cloud computing based on Blockchain technology. Simul. Modell. Pract. Theory 2020;99.

Xu Xiaolong, Chen Yi, Yuan Yuan, Huang Tao. Blockchain-based cloudlet management for multimedia workflow in mobile cloud computing. In: Multimedia Tools and Applications, 79. springer; 2020. p. 9819–44.

You Wencong, Jiao Lei, Li Jun, Zhou Ruiting. In: IEEE International Conference on Computer Communications (INFOCOM). Scheduling DDoS cloud scrubbing in ISP networks via randomized online auctions; 2020.

Zitzler Eckart, Deb Kalyanmoy, Thiele Lothar. Comparison of multiobjective evolutionary algorithms: Empirical results. Evol. Comput. 2000;8(2):173–95.

Zuo Liyun, Dong Shoubin, Shu Lei, Zhu Chunsheng, Han Guangjie. A multiqueue interlacing peak scheduling method based on tasks' classification in cloud computing. IEEE Syst. J. 2016;12(2):1518–30.