


Security-Aware Autonomic Allocation of Cloud Resources: A Model, Research Trends, and Future Directions

Sukhpal Singh Gill, Queen Mary University of London, UK

 <https://orcid.org/0000-0002-3913-0369>

Arash Shaghaghi, Deakin University, Australia

ABSTRACT

Cloud computing has emerged as a dominant platform for computing for the foreseeable future. A key factor in the adoption of this technology is its security and reliability. Here, this article addresses a key challenge which is the secure allocation of resources. The authors propose a security-based resource allocation model for execution of cloud workloads called STARK. The solution is designed to ensure security against probing, User to Root (U2R), Remote to Local (R2L) and Denial of Service (DoS) attacks whilst the execution of heterogeneous cloud workloads. Further, this paper highlights the promising directions for future research.

KEYWORDS

Autonomic Computing, Cloud Computing, Cloud Security, Cyber Attacks, Security, Self-Protection

1. INTRODUCTION

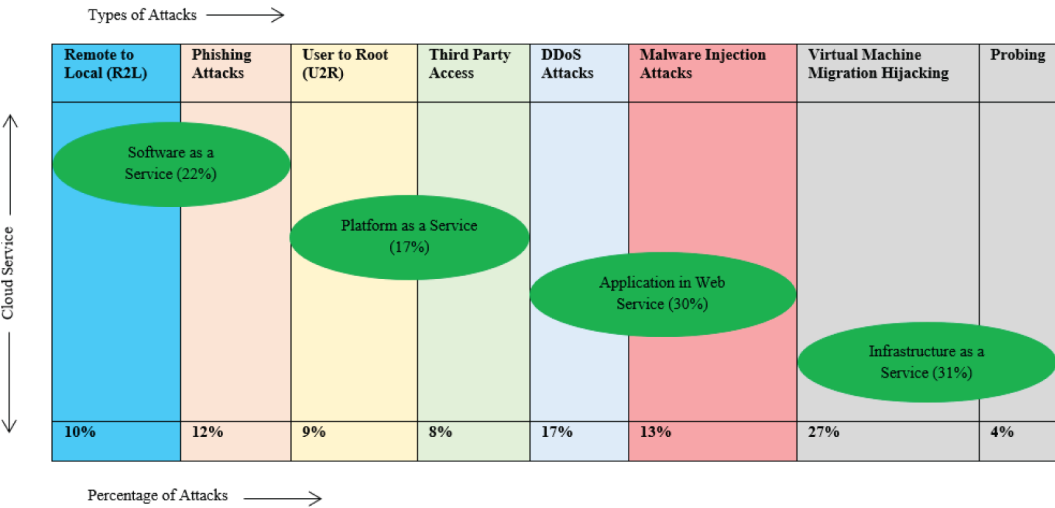
The fast developments in Information and Communication Technology (ICT) have enabled the emerging of the “cloud” as a successful paradigm for conveniently storing, accessing, processing, and sharing information (Varghese & Buyya 2017). With its significant benefits of scalability and elasticity, the cloud paradigm has appealed companies as well as individuals, which are more and more resorting to the multitude of available providers for storing and processing data. Unfortunately, such convenience comes at the price of owners’ loss of control over their data, and consequent security threats, which can limit the potential widespread adoption and acceptance of the cloud computing paradigm (Singh et al., 2017). Figure 1 shows the different types of attacks for cloud services such as software, platform, application and infrastructure.

On the one hand, cloud providers can be assumed to employ basic security mechanisms for protecting data in storage, processing, and communication, devoting resources to ensure security that many individuals and companies may not be able to afford. On the other hand, data owners, as well as users of the cloud, lose control over data and their processing. Currently, cloud services are provisioned and scheduled according to resources’ availability without ensuring the required security

DOI: 10.4018/JOEUC.2020070102

This article, originally published under IGI Global’s copyright on July 1, 2020 will proceed with publication as an Open Access article starting on January 21, 2021 in the gold Open Access journal, Journal of Organizational and End User Computing (converted to gold Open Access January 1, 2021), and will be distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

Figure 1. Different types of attacks for cloud services



(Singh et al., 2017). The cloud provider should evolve its ecosystem to meet security requirements of each cloud component (Pietro et al., 2016) and specifically, deploy mechanisms to ensure the secure and autonomic management of resources.

In this research paper, we have proposed a conceptual model for security-aware allocation of cloud resources called STARK, which provides protection against security threats during resource management for workload execution. This work (STARK) is an extension of our previous research work i.e. STAR (Singh et al., 2017). The rest of paper is structured as follows. Section 2 presents proposed security-aware resource management model. Thereafter, we present a set of future research directions and conclude the paper in Section 3.

2. STARK: SECURITY BASED RESOURCE ALLOCATION MODEL FOR CLUSTERED WORKLOADS

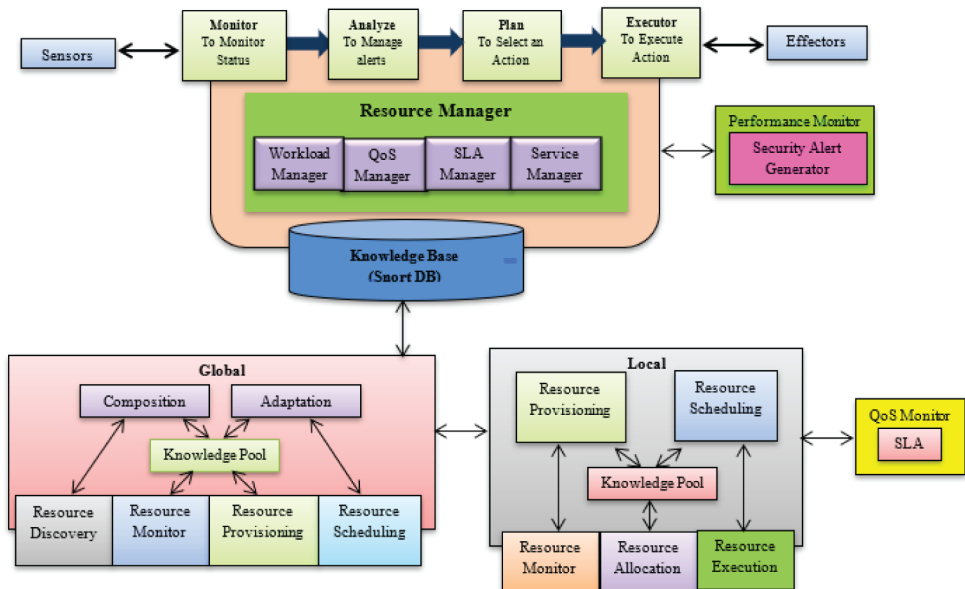
As the scale of mobile networks and the population of mobile users increases, the applications of cloud computing have emerged where social users can use their mobile devices to exchange and share contents with each other (Qu et al., 2010). The security resource is needed to protect mobile social big data during the delivery. However, due to the limited security resource, how to allocate the security resource becomes a new challenge. Existing security-based resource management techniques only focuses on homogenous cloud workloads and provide protection against Denial of Service (DoS) attack (Wailly et al., 2012; Singh & Chana 2013; Bittencourt et al., 2017). To solve this problem, there is the need for a new solution, which can protect the cloud against different types of security attacks during the autonomic execution of cloud workloads.

We propose “Security-based resource allocation model for clustered workloads in cloud environment” (STARK). Our proposed solution provides protection against the following four main types of security attacks including probing, User to Root (U2R), Remote to Local (R2L) and Denial of Service (DoS), which occur whilst the heterogenous cloud workloads are executed. STARK works through four main stages including monitoring, analysis, planning and execution inspired by of an IBM autonomic model. In fact, STARK continuously monitors for security attacks and analyses any alert triggered due to an attack, it then plans an action to handle the identified threat and executes a set of actions to ensure the security of the cloud. This section discusses the architecture of a proposed conceptual model, which offers self-protection against security attacks with minimum

human intervention. The architecture of STARK is shown in Figure 2 and its main components are discussed below:

- Resource management in STARK is done at two levels: global and local. At global level, workloads or applications are submitted for execution along with Service Level Agreement (SLA) pertaining QoS requirements. Execution of workload or application is divided into sub tasks or small levels called local levels. The actual execution of workload or application is performed at local level after verification of availability of resources;

Figure 2. STARK architecture (Singh et al., 2017)



- The knowledge pool stores the predefined rules defined by system administrator and rules will be updated time to time based on new policies of resource allocation;
- Aim of resource allocation is to allocate appropriate resources to the suitable workloads on time, so that applications can utilize the resources effectively;
- Resource discovery is able to find the adequate (with minimum cost and execution time) resource for workload or application based on Quality of Service (QoS) requirements through composition;
- Composition is able to determine the best resource workload pair for execution;
- Resource monitor monitors whether the conditions are being fulfilled as specified in policy and ensures that all the resources are provided. All the inputs received from monitors are analyzed and actions are taken according to the alert generated by QoS monitor;
- QoS monitor is used to verify whether all the QoS attributes defined in SLA are fulfilled or not by using Adaptation. If not, then QoS monitor generates an alert to provide more resources to fulfill the current demand of application;
- Adaptation function is able to maintain the effective execution in case of sudden change in QoS conditions. Based on QoS requirements and policy of system, resources are provisioned to workload or application and resource provisioning information is sent to user for verification;

- After successful verification by user, the autonomic security manager protects execution of workloads;
- Resource executor performs the final step of resource execution and completes the execution within specified deadline and system continues for other workloads or applications;
- For QoS assessment, the QoS manager calculates the execution time of workload and finds the approximate workload completion time;
- Resource manager maintains resource details which includes the number of CPU used, size of memory, cost of resources, type of resources and number of resources. It contains the information about the available resources and reserved resource along with resource description (resource name, resource type, configuration, availability information, usage information and price of resource) as provided by cloud provider;
- Based on SLA information, the SLA manager prepares SLA document which contains information about SLA Violation Rate (maximum and minimum violation and penalty rate in case of SLA violation) and accordingly urgent cloud workloads would be placed in priority queue (workload with urgent deadline in execution state) for earlier execution;
- The service manager manages the whole service of the system in a controlled manner. Based on SLA information, QoS information, workload information and resource information, the resource workload mapper maps the workloads to the appropriate resource by taking care of both SLA and QoS. Resource scheduler is further used to schedule the workloads after mapping of the workloads with available resources and generates the workload schedule. Resource scheduler uses minimum number of resources to execute the workloads within specified budget and deadline. STARK protects the execution of clustered workloads (Singh et al., 2017) from security attacks.

The main aim of self-protecting in STARK is to protect the system from intentional attacks. In essence, STARK works by tracking the doubtful activities and respond accordingly to maintain the working of system without any disruption. The system should be trained to differentiate between legitimate and malicious behaviour in order to enforce protection properly. As mentioned, STARK is designed to protect a cloud against DoS, R2L, U2R and Probing attacks (Vance & Siponen, 2012). In DoS (Denial of Service) attacks, a large amount traffic is generated by attackers to cause damage by flooding the victim's network (Gill & Buyya, 2019). It includes SMURF (to create denial of service, attackers use ICMP (Internet Control Message Protocol) echo request by pointing packets towards broadcast IP address), LAND (Local Area Network Denial) (when source and destination address is same, then attackers send spoofed SYN packet in TCP/IP network) and SYN Flood (attackers sending IP-spoofed packets which can crash memory). In Remote to Local (R2L), attackers access the system locally without authorization to destroy the network by executing their commands (Benkhelifa & Welsh, 2014). It includes attacks like IMAP (Internet Message Access Protocol), Guess password and SPY (Singh & Buyya, 2019). In User to Root (U2R), attackers get root access of the system to destroy the network. It includes attacks like buffer overflow and rootkits (Gill et al., 2019). In Probing, attackers use programming language to steal the private information. It includes attacks like port sweep and NMAP (Network MAPper).

STARK provides security during execution of user workloads automatically. During the execution of workloads, security is monitored continuously using a sub-unit performance monitor to maintain the efficiency of STARK that generates alert in case of security attack (Paul 2015). STARK provides self-protection and ensures the secure communication among autonomic elements during resource execution (Gill & Buyya 2018). STARK comprises of six components of autonomic model: sensor, monitor, analyze, plan, execute and effector. Sensors get the information about performance of current state nodes in terms of a QoS parameter i.e. Intrusion Detection Rate (Somani et al., 2017). Firstly, the updated information coming from processing nodes is transfer to manager node then manager node transfers this information to monitors. Updated information includes information about security threats.

For the monitoring module, security agents are installed on all the processing nodes, which are used to trace the unknown and known attacks. Based on the existing database in the system, new anomalies are captured in STARK. STARK captures an anomaly by detecting system intrusions and misuse of system by using its monitor and classifying it as either normal or anomalous by comparing its properties with data in existing database. New anomalies are captured by security agent and information about anomalies is stored in database (knowledge base). STARK protects the system from various attacks as discussed earlier such as DoS [Smurf, LAND, SYN Flood and Teardrop], R2L [SPY, Guess password, IMAP], U2R [Rootkits, Buffer Overflow] and Probing [Ports sweep and NMAP]. SNORT anomaly detector (Gai et al., 2017) is used to protect the system from attacks. We have used detection engine to detect the attacks and maintain the log about attack. Detection engine detects the pattern of every packet transferring through the network and compares with the pattern of packets existing in database to find the current value. Alert will be generated if current value is out of range [Range (Min, Max)]. State Vector Machine (Gill et al., 2018) [7] is used in STARK to make a network profile for attack detection, which is designed based on the training data to detect and recognize input (user data) and based on the closed match to the data defined in classes, output is produced.

For analyzing and planning module, the analyzing unit starts analyzing the log information of attacks after alert is generated by security agent to generate signature (Gill et al., 2019). STARK performs the following functions to generate signatures:

- Collect all the new alerts generated by the autonomic manager;
- Use Java utility to perform parsing to get URL, port, and payload detail;
- Categorize data based on URL, port, and payload;
- Find the largest common substring apply longest common subsequence (LCS);
- Construct a new signature by using payload string identified by LCS.

For the executing module, SNORT (Caswell & Beale 2004) can be used to refine the signature received from analyzer for Intrusion Defense Systems (IDS) and Intrusion Prevention Systems (IPS) sensors and compares new signatures with existing signature in SNORT database. If signatures are new, then they are added to SNORT database (knowledge base) and if signatures are existing then they are merged. Effector is acting as an interface between autonomic units to exchange updated information and it is used to transfer the new policies, rules and alerts to other nodes with updated information.

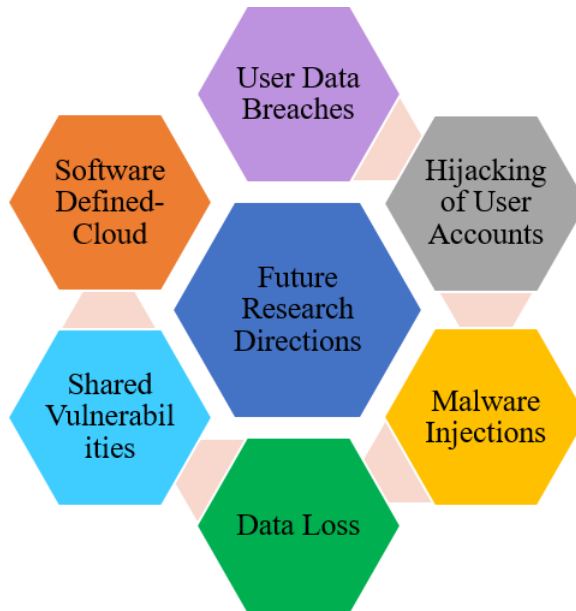
3. CONCLUSION AND FUTURE RESEARCH DIRECTIONS

In this paper, we propose STARK, which is a secure resource allocation model for the execution of cloud. STARK efficiently schedules the provisioned cloud resources for the execution of heterogeneous cloud workloads and maintains the security of cloud services. In the future, STARK will be verified in a cloud environment to evaluate it in practice. The impact of security on QoS will also be analyzed.

Figure 3 shows the future research directions in security-aware cloud resource management, which are discussed below:

- **User Data Breaches:** With sensitive data (online banking transactions) being stored online rather than on premise, is the cloud inherently less safe?
- **Hijacking of User Accounts:** Attackers now have the ability to use cloud user's login information to remotely access sensitive data stored on the cloud; additionally, attackers can falsify and manipulate information through hijacked credentials;

Figure 3. Future research directions in security-aware cloud resource management



- **Malware Injections:** During workload execution, malicious code can be injected into cloud services and viewed as part of the software or service that is running within the cloud servers themselves;
- **Data Loss:** Data on cloud services can be lost through a malicious attack, natural disaster, or a data wipe by the service provider. Losing vital information can be devastating to businesses that don't have a recovery plan;
- **Shared Vulnerabilities:** Cloud security is a shared responsibility between the provider and the client. This partnership between client and provider requires the client to take preventative actions to protect their data. While major providers like Box, Dropbox, Microsoft, and Google do have standardized procedures to secure their side, fine grain control is up to you, the client;
- **Software Defined-Cloud:** Software Defined-Cloud can be used to provide secure communication among cloud nodes. Transport Layer Security (TLS)/Secure Sockets Layer (SSL) encryption techniques can also be used to provide secure communications between controller(s) and OpenFlow switches, the configuration is very complex, and many vendors do not provide support of TLS in their OpenFlow switches by default. Software Defined Network (SDN) security is critical since threats can degrade the availability, performance and integrity of the network.

REFERENCES

- Benkhelifa, E., & Welsh, T. (2014). Towards Malware Inspired Cloud Self-Protection. *Proceedings of the IEEE International Conference on Cloud and Autonomic Computing (ICCAC)* (pp. 1-2). IEEE Press.
- Bittencourt, L. F., Diaz-Montes, J., Buyya, R., Rana, O. F., & Parashar, M. (2017). Mobility-aware application scheduling in fog computing. *IEEE Cloud Computing*, 4(2), 26–35. doi:10.1109/MCC.2017.27
- Carpen-Amarie, A. (2011). Towards a self-adaptive data management system for cloud environments. *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW)* (pp. 2077-2080). IEEE Press. doi:10.1109/IPDPS.2011.381
- Caswell, B., & Beale, J. (2004). Snort 2.1 intrusion detection. Syngress. Retrieved from <https://www.snort.org/>
- Di Pietro, R. D., Lombardi, F., & Signorini, M. A. T. T. E. O. (2016). Secure Management of Virtualized Resources. In *Security in the Private Cloud*. Academic Press.
- Gai, K., Qiu, L., Chen, M., Zhao, H., & Qiu, M. (2017). SA-EAST: Security-aware efficient data transmission for ITS in mobile heterogeneous cloud computing. *ACM Transactions on Embedded Computing Systems*, 16(2), 60. doi:10.1145/2979677
- Gill, S. S., & Buyya, R. (2018). A taxonomy and future directions for sustainable cloud computing: 360 degree view. [CSUR]. *ACM Computing Surveys*, 51(5).
- Gill, S. S., & Buyya, R. (2018, January/February). SECURE: Self-Protection Approach in Cloud Resource Management. *IEEE Cloud Computing*, 5(1), 60–72. doi:10.1109/MCC.2018.011791715
- Gill, S. S., & Buyya, R. (2019). Resource provisioning based scheduling framework for execution of heterogeneous and clustered workloads in Clouds: From fundamental to autonomic offering. *Journal of Grid Computing*, 17(3), 385–417. doi:10.1007/s10723-017-9424-0
- Gill, S. S., Chana, I., Singh, M., & Buyya, R. (2018). CHOPPER: An intelligent QoS-aware autonomic resource management approach for cloud computing. *Cluster Computing*, 21(2), 1203–1241. doi:10.1007/s10586-017-1040-z
- Gill, S. S., Tuli, S., Xu, M., Singh, I., Singh, K. V., Lindsay, D., & Garraghan, P. et al. (2019). Transformative Effects of IoT, Blockchain and Artificial Intelligence on Cloud Computing: Evolution, Vision, Trends and Open Challenges. *Internet of Things*, 8, 100118. doi:10.1016/j.iot.2019.100118
- Manuel, P. (2015). A trust model of cloud computing based on Quality of Service. *Annals of Operations Research*, 233(1), 281-292.
- Qu, G., Rawashdeh, O. A., & Che, D. (2010). Self-Protection against Attacks in an Autonomic Computing Environment. *International Journal of Computers and Applications*, 17(4), 250–256.
- Singh, S., & Chana, I. (2013). Consistency Verification and Quality Assurance (CVQA) Traceability Framework for SaaS. *Proceeding of 3rd IEEE International Advance Computing Conference (IACC-2013)*. Academic Press. doi:10.1109/IAdCC.2013.6506805
- Singh, S., & Chana, I. (2016). QoS-aware autonomic resource management in cloud computing: A systematic review. *ACM Computing Surveys*, 48(3), 1–46. doi:10.1145/2843889
- Singh, S., Chana, I., & Buyya, R. (2017). STAR: SLA-aware Autonomic Management of Cloud Resources. *IEEE Transactions on Cloud Computing*. doi:10.1109/TCC.2017.2648788
- Somani, G., Gaur, M. S., Sanghi, D., Conti, M., & Buyya, R. (2017). DDoS attacks in cloud computing: Issues, taxonomy, and future directions. *Computer Communications*, 107, 30–48. doi:10.1016/j.comcom.2017.03.010
- Vance, A., & Siponen, M. T. (2012). IS security policy violations: A rational choice perspective. *Journal of Organizational and End User Computing*, 24(1), 21–41. doi:10.4018/joeuc.2012010102
- Varghese, B., & Buyya, R. (2017). Next generation cloud computing: New trends and research directions. *Future Generation Computer Systems*, 79, 849–861. doi:10.1016/j.future.2017.09.020
- Wailly, A., Lacoste, M., & Debar, H. (2012). Vespa: Multi-layered self-protection for cloud resources. *Proceedings of the 9th ACM International Conference on Autonomic computing* (pp. 155-160). ACM.

Sukhpal Singh Gill is a lecturer (Assistant Professor) in Cloud Computing at School of Electronic Engineering and Computer Science (EECS), Queen Mary University of London, UK. Prior to this, Dr. Gill has held positions as a Research Associate at the School of Computing and Communications, Lancaster University, UK and also as a Postdoctoral Research Fellow at the Cloud Computing and Distributed Systems (CLOUDS) Laboratory, School of Computing and Information Systems, The University of Melbourne, Australia. Dr. Gill was a research visitor at Monash University, University of Manitoba and Imperial College London. He was a recipient of several awards, including the Distinguished Reviewer Award from Software: Practice and Experience (Wiley), 2018, and served as the PC member for venues such as UCC, SE-CLOUD, ICCCN, ICDICT and SCES. His one review paper has been nominated and selected for the ACM 21st annual Best of Computing Notable Books and Articles as one of the notable items published in computing - 2016. He has published more than 50 papers as a leading author in highly ranked journals and conferences with H-index 20. His research interests include Cloud computing, Fog computing, software engineering, Internet of Things and Big Data. For further information on Dr. Gill, please visit: www.ssgill.me.

Arash Shaghaghi is an Assistant Professor (Lecturer) in Cybersecurity at Deakin University, Melbourne, Australia. He is also a Visiting Fellow at the School of Computer Science and Engineering at The University of New South Wales (UNSW), Sydney, Australia. He received PhD from UNSW Sydney where he was also affiliated with CSIRO, Data61. Previously, he completed an MSc in Information Security at University College London (UCL) and a BSc in Information Technology at Heriot-Watt University of Scotland. Arash has been consistently ranked as a top student in his studies and received multiple awards and scholarships in the past few years including the Australian Government Research Training Program Scholarship. Arash research interests include network and system security, access control systems, Cloud Computing, and applied cryptography.