Lightweight Fine-Grained Access Control Mechanism Based on Zero Trust in CPS

Nakul D. Ghate
NEC Corporation, Japan
Tokyo, Japan
email: te.nak14@nec.com

Shohei Mitani
NEC Corporation, Japan
Tokyo, Japan
email: s.mitani@nec.com

Hirofumi Ueda
NEC Corporation, Japan
Tokyo, Japan
email: h-ueda_cb@nec.com

***Abstract*— The paper explores the trade-off between security and workload when enforcing fine-grained access control in Cyber Physical Systems network. The paper describes a novel approach to select the access control granularity based on dynamic environment conditions by distributing a part of fine-grained application-level policy on a network-level access controller to reduce the workload while ensuring security. Under the desk evaluation, we achieved a workload reduction of over 90% compared to the input policy, with a granularity degrade of just 15%. Although, some mis-control due to denying essential requests can be observed in the distribution-based approach, the presented algorithms are conceptualized to minimize it. The preliminary experimental results show promising improvement in the access control system performance when employing this approach.**

***Keywords- Cyber physical systems; zero trust; fine-grained; workload; distributed access control*.**

I. INTRODUCTION

Beyond 5G / 6G network has highly enabled the integration of physical systems with the cyber world in the form of Cyber Physical Systems (CPS) [1], whose applications range from smart manufacturing, healthcare, power grids, Internet of vehicles, smart homes and so on [2]. Because they are deployed in critical infrastructures, the security of such systems has become ever important. The heterogeneity of the devices utilized in the CPS is one of the fundamental issues in CPS security. Many sensors, and actuators used are constrained IOT devices, on which deploying security functions is a challenge [3]. As CPS integrates many such hardware, along with software used for monitoring and control, etc., every site in CPS network functions as an entry point for malware to intrude into organization's network [2].

Traditional network-perimeter based defense model has become obsolete in the dynamic CPS network due to (1) failure to prevent lateral movement inside the network perimeter as everything inside the perimeter is trusted [4], (2) emergence of cloud services which blur the perimeter boundary by extending the enterprise resource access through third party servers [5]. With cloud services ever evolving, achieving practical security is impossible using perimeter-based defense techniques. Zero Trust (ZT) is the term for an evolving set of cybersecurity paradigms that move defenses from static, network-based perimeters to focus on users, assets, and resources [6]. Access control mechanisms utilizing the ZT principles assume that threats exist everywhere, and no user or device is trusted solely based on its physical or network location. The ZT-based access control continuously performs authentication and authorization to ensure only the authorized entity is permitted to access protected resource(s), adapting to the principle of least privilege to prevent lateral movement. However, to achieve this effectively, it requires fine-grained access control for authorization, where access rules are defined for individual users, devices, resources, applications, and so on. An example is Attribute Based Access Control (ABAC) [7], enforced with mechanisms such as Attribute-Based Encryption (ABE) [8] and with Application-level access policy designed for secure access to resources, when the access risk is associated with attributes such as "device ID", "resource ID", "resource confidentiality", "device behavior", "user-behavior", and so on. The application-level access policy utilizes the influence of these attributes to decide access decision, and such decisions are performed at application-level access controllers which define the fine-grained authorization rules from the application-level policy. The fine-granularity contributes to the large workload of access control mechanism in terms of increased storage of enforceable rules, large computation cost of ABE, higher processing load on the access controller, etc. [9][10][11]. These drawbacks may result in latency in enforcing access decisions and can be a possible target for Distributed Denial of Service attacks (DDOS), hindering enterprise operations [12].

Implementing coarse-grained access control such as network-level access control that defines authorization rules using the network attributes such as "source IP", "destination IP", etc. reduces the access control workload by defining a single access rule for many devices and resources contained within the same IP address. But due to its coarseness, it fails in achieving least privilege security, thereby implying that a trade-off exists between the security and workload when subjected to the granularity of access control policies. Our approach implements a distributed access control mechanism which distributes the access control decisions on sequentially implemented access controllers: network-level access controllers that utilize coarse-grained access policies and application-level access controllers that deal with fine-grained application-level policies. With this, we aim to achieve both high security and low workload to overcome the existing issue. The rest of the paper is organized as follows: Section II

 ISBN: 978-1-68558-092-6

presents the related study, Section III presents the approach to solve the problem, Section IV describes the methodology for the approach. The experimentation and results are described in Section V, while Section VI presents the discussion of our work. Finally, we conclude our work in Section VII.

II. Related Study

The security and workload trade-off exists because the implementation of traditional ZT-based access control in the existing literature is static in terms of access granularity, i.e., it either implements coarse-grained network-level access control [13] or fine-grained application-level access control [14]. For instance, [13] embeds authentication tokens inside TCP packets and first-packet authentication, therefore, enforcing ZT principles with static rules and coarse-grained network-level access control, while [14] only considers Attribute Based Access Control (ABAC) rules to be enforced at the Policy Enforcement Point (PEP) resulting in high workload. Such static approaches to implement ZT-based access control do not result in optimal performance with respect to the workload-security trade-off.

To reduce the workload of application-level policies, existing methods [15][16] focused on controlling the access at network devices such as firewall, based on application-awareness. In [15], the access granularity of the firewall is changed from coarse-grained *packet filtering* to a finer-grained *stateful* TCP or application-level *deep packet inspection*, depending on the application security requirements and static access control policies. In [16], an application-aware network access control for IOT services is proposed based on SDN using mandatory access control (MAC). While they simplify access management of fine-grained access control, they fail to recognize the access risks from the heterogenous devices as well the dynamically changing environment conditions. This may lead to malware infection caused by wrong choice of access granularity by only considering application awareness. [17] proposed a policy-based dynamic network access control by utilizing the real-time feedback from network devices and application servers. However, it is proposed as a conceptual framework only. Several multi-layer access control methods are also provided to enhance security. In [12], the authors proposed cooperation of PEP among network-level and application-level services in the same or remote domains to facilitate defense in depth. However, the solution relies heavily on static application-level policies. In [18], a dual layer ZT architecture is proposed where the policy evaluates user's 5G network layer behavior and industry application layer behavior. In [19] a multi-layer authorization framework of Apache Hadoop is discussed which covers a range of services. Both of them do not consider the workload and scalability aspect.

In a dynamic environment such as CPS, the deployed access control method should be aware of the changing attributes of both the access subject as well as resources, and adaptively respond by enforcing access control rules with choosing appropriate granularity. For instance, enforcing network-level access control when the risk is related to the changing network attributes, such as suspicious activities from a source IP address, and on the other hand, enforcing application-layer access control when the risk is related to the changing attributes of user or device behavior, etc. We propose a novel approach of access granularity selection based on analyzing the dynamic environment.

III. Approach

We argue that the optimal performance is achieved when the security and workloads are balanced by the access control system towards overall business growth by facilitating access continuity. We propose that dynamic selection of access control granularity promises to achieve this balance. A schematic comparison of the existing work [15] and our approach is shown in Figure 1. Due to the fact that the existing work does not take into account for dynamically changing behaviors of access subjects, such as users and (IOT) devices, it would always choose coarse-grained packet filtering for non-confidential resource as shown in Figure 1, which may result in malicious device taking control over the resource. On the other hand, it would always choose fine-grained deep inspection to protect sensitive resources, even for trusted devices, which requires large workload and induce latency in the access.

Our approach lies in dynamically distributing the (fine-grained) application-level access control towards (coarse-grained) network-level access control. Our algorithm dynamically decides which policies are safe to be distributed towards network-level access control for the intention of reducing the workload while not compromising the security. On a high level, we utilize an application-level access policy which intends to be enforced at the application-level access controller in the traditional methods, and distribute it into two sets of policies, the coarse-grained policy enforced at the network-level access controller and the remaining subset of the application-level policies enforced at the application-level access controller as shown in Figure 1.

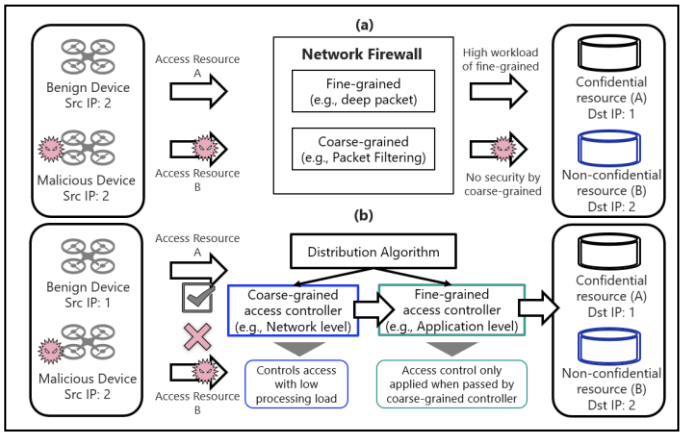

Figure 1. Comparison between (a) existing method and (b) our approach.

We take advantages of the following three properties as long as it doesn't compromise security; 1) Network-level access control requires lower processing load compared to application-level, 2) A network-level access controller deployed in front of an application-level access controller cuts off access requests, 3) The size of network-level access policy is smaller than that of application-level policy if they represent

 ISBN: 978-1-68558-092-6

same access decision rules. In this way, the network-level access control reduces both the workload (as the application-level access controller only controls access to the requests which are passed on by the network-level access controller), as well as cuts of malicious access and attacks such as DDOS early on, securing the network against unnecessary bandwidth consumption.

IV. METHODOLOGY

We describe the methodology to our approach in this section. Assuming an existing application-level policy, the objective of our distributed access control is to reduce the workload and operation cost of directly enforcing the application-level access control policy. Instead, we distribute and enforce it on both network-level and application-level access controller. A trivial solution is to enforce the whole policy as a network-level policy to achieve least workload. However, it has a problem. In a CPS network, many devices may share a common IP address. Assume that for accessing certain resource, one device belonging to a certain IP have "allow" decision in the application-level policy, meanwhile another device have "deny" decision. If the devices share same IP address (and same destination port number), those two devices will be assigned the same decision under the network-level access control. Hence, one of them would be mis-controlled by the decision due to policy differences. Our algorithm evaluates the mis-control rate and decides if some parts of the application-level policy can be distributed and enforced at the network-level access control. Because network-level access controller can deal with (coarse-grained) network-level policy only, we use the aggregation approach where the decisions controlling access of all *devices* from a *source IP* address to all the *resources* in the *destination IP* address can be aggregated into a single coarse-grained network-level policy enforceable at the network access controller as shown in Figure 2.

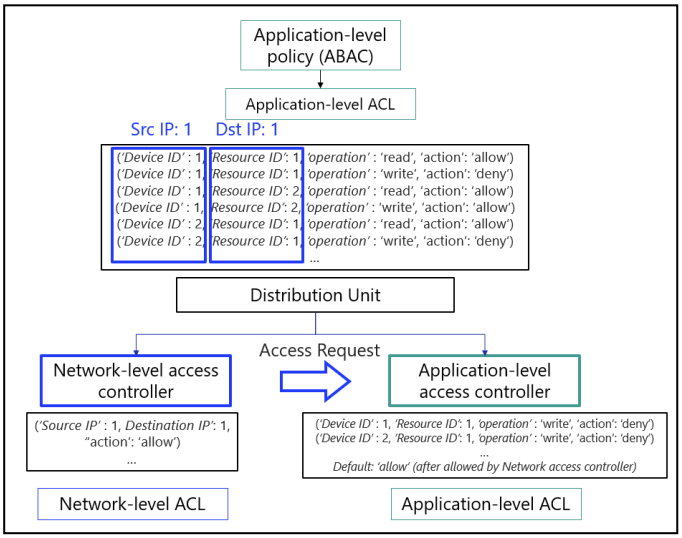

Figure 2. Distribution of application-level access control policies.

Algorithm. In this study, we utilized a manually defined application-level policy (such as ABAC) as the input. We assume that the application-level access policy accurately determines the authorization decisions based on the dynamic environment. For distribution algorithm to work, the policy must be defined in an enforceable form. We picked the Access Control List (ACL) [20] format for the enforceable policies. The application-level policy (ABAC) is first converted to Application-level ACL before distribution as shown in Figure 2. The distribution algorithm is described in Algorithm 1.

Pseudo-code of distribution of access control policies

Input: Application-level policy F = list (Device ID, Resource ID, Operation, Decision)

1. **DO**
2. **Create** all pairs: p = (source IP, destination IP) ∈ P
3. **Create** dictionaries: nw_acl (Network-level ACL), app_acl (Application-level ACL), D (decision), SubP (sub-Policy) with keys p for each p∈ P
4. **Initialize** 'allow count' D[p][ac] = 0 and 'deny count' D[p][dc] = 0 for each p∈ P
5. **FOR** each e = (Device ID, Resource ID, Decision) in F
6. **Find** p for (Device ID, Resource ID) ∈ e
7. If Decision == 'allow'
8. D[p][ac] <- D[p][ac] + 1
9. If Decision == 'deny'
10. D[p][dc] <- D[p][dc] + 1
11. SubP[p] <- append (e)
12. **END FOR**
13. **FOR** each p∈ P
14. **Apply** *policy aggregation*
15. **END FOR**
16. **RETURN** (nw_acl, app_acl)

Policy aggregation

1. Calculate **allow rate** AR = (ac) / (ac + dc)
2. **IF** AR > threshold
3. nw_acl[p] <- {action} = 'allow'
4. app_acl[p] <- SubP[p]
5. **ELSE**
6. nw_acl[p] <- {action} = 'deny'
7. **END FOR**

Algorithm 1. Distribution of application-level policy.

The input application-level policy *F* is defined in an ACL format with attributes '*Device ID*', '*Resource ID*', '*Operation*' (such as 'read', 'write' operations, etc.) and the action decision (such as 'allow' or 'deny'). The algorithm defines the output network-level ACL '*nw_acl*' with attributes 'source IP', 'destination IP', and action decision ('allow' or 'deny') and the output application-level ACL '*app_acl*' defined with same attributes as *F*. The aggregation approach uses the attributes of the network-level ACL, i.e., *source IP*, *destination IP* (if the IP-level access control is enforced). The algorithm proceeds as follows: for each access pattern in *F*, it finds the pair *p* = (*source IP*, *destination IP*) associated with the pair (*device ID*, *resource ID*) using the associated binding between device ID, resource ID and their attributes. Then, for each pair *p* controlled by the network-level access controller, the aggregation algorithm evaluates the 'allow count' *ac* and the 'deny count' *dc* by calculating the total number of 'allowed' and 'denied' access patterns respectively. The decision dictionary *D* stores this value for each p∈P, where P is the set of all pairs (*source IP, destination IP*). All the access patterns for each (*device ID, resource ID*) associated with the pair *p* are

stored as sub-policies of *p* in the dictionary *SubP*. The policy aggregation algorithm calculates the 'allow rate' *AR* of access decisions for all the pairs (*device ID*, *resource ID*) belonging to *p*. The allow rate is compared against a set *threshold*. If the 'allow rate' is greater or less than the set *threshold*, the network-level acl *nw_acl* for each *p* is set to "allow" or "deny" respectively. The policy aggregation operation is shown in algorithm 1. For each "allow" decision in the network-level ACL, an application-level ACL *app_acl* is distributed towards the application-level access controller (by appending sub-Policy *SubP* to *app_acl*). This ensures only legitimate access is permitted to access the resources (at the application-level) while the rest is denied. The value of set *threshold* depends on how strict one wants to set the access control for network access control. Higher the threshold, stricter becomes the network access control. However, denying access at network-level access control may cause limitations, which along with a conceptual solution is discussed in Section VI.

V. Experiment And Evaluation

We performed a desk evaluation in Python to show the effectiveness of our approach. We assume that many devices are assigned a common IP address, e.g., through Network Address Translation (NAT). Likewise, many resources are contained in a single server that has a certain IP address. Our experiment considers two source IP addresses. The number of devices is increased from 10 to 100 in a succession of 10 devices. Similarly, at destination, two resource servers, each assigned with a unique destination IP address and contain 5 resources each. Any device can request any resource and the access is controlled in a similar fashion as Figure 1(b) with two controllers: Network-level access controller and Application-level access controller. In our first evaluation, we considered the effect of applying policy distribution on the size of the ACL (number of entries in ACL). We created the input application-level ACL with *'allow'* probability of approx. 40% (from the ABAC policy). The distribution algorithm distributes this ACL into network-level and remaining set of application-level ACL enforced at network-level access controller and application-level access controller respectively. Figure 3 shows the ACL size comparison before and after applying policy distribution.

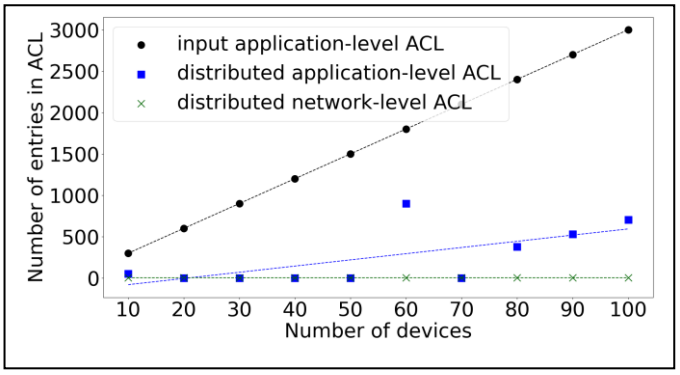

Figure 3. Comparison of the size of the policy before and after distribution.

It can be observed that with the increase in the number of devices, the difference between the size of the input and distributed application-level ACL is increased. This is due to the presence of network-level access controller which cuts the access before reaching application-level access controller. As the ACL is a list of sequentially arranged filters or commands, the throughput is inversely proportional to the size of the ACL [21]. The increase in the difference between the application-level ACL size suggests that the throughput for the distributed application-level ACL will be higher compared to the original application-level ACL, indicating latency reduction.

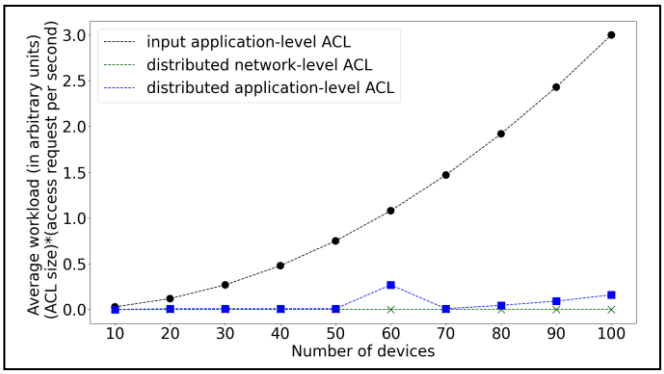

Figure 4. Comparison of the average workload on each controller before and after distribution algorithm.

TABLE I. PERFORMANCE COMPARISION BETWEEN ORIGINAL AND DISTRIBUTED ACL.

Metric	Original Application-level ACL	Network-level ACL	**Proposed method**
Access workload	(100%)	1%	**7%**
Access granularity	(100%)	38%	**85%**

We compare the security and workload trade-off of our proposed method through two metrics: **average access workload** and **access granularity**. The average access workload is described here, as the average time to process all the access requests on the given access controller at an instance, and it is approximated as the product of the size of the ACL used by the controller and the number of access requests falling on it [22]. For this, we artificially created 100 access requests patterns in the format ('device ID, 'resource ID', 'operation') for every device and resource pair. Figure 4 shows the comparison of access workload (defined in arbitrary units) between input policy and distributed policies. Without the distribution algorithm in place, all the access requests will be managed by the application-level access controller. As the number of access requests increase with the increase in number of devices, the access workload on the application-level access controller will keep on increasing. However, when the distribution is applied, the access workload will be divided among both the network-level as well as the application-level access controllers. This, together with smaller ACL size after distribution will result in significant workload reduction after distribution relative to input application-level ACL as shown in Figure 4. For the case of 100 devices, relative to the access workload at original application-level ACL (taken as 100%), the total access workload (network-level ACL + application-level ACL) after distribution is only around **7%**, which is

 ISBN: 978-1-68558-092-6

comparable to simply implementing only network-level ACL as shown in Table 1.

To evaluate the impact on security after distribution, we utilized access granularity metric. We measure the access granularity of a given ACL as the fraction of all access decisions enforced by the ACL which match the input application-level ACL, given the same access patterns. We assume that the input application-level ACL is carefully constructed to provide accurate access decisions with fine granularity. Therefore, any access deviations from the input application-level ACL will result in degrade of access granularity, and thus a degrade in security, as the new access decisions enforced by the access controllers after the policy distribution would not be correct. If we only use network-level ACL for access control, then in case of 100 devices, as expected, the access granularity of a network-level ACL is only 38% relative to the input application-level ACL. However, our proposed method achieves access granularity of **85%**, significantly greater than the network-level ACL. Our method thus, suggests **greater reduction in access control workload** of application-level ACL by distributing the workload among the network-level and application-level access controllers **without degrading the satisfaction of security requirements**.

VI. DISCUSSION

In our method, the ACLs are dynamically distributed. Meaning, once the environmental conditions are changed, i.e., the attribute values belonging to the subject, resource or context change, or when new device join, etc., new ACL rules are distributed, and the previous ones are revoked (by any internal mechanism inside the controllers). The ACL enforced by our distributed mechanism sharply reduce the access control workload of application-level access controller by transferring many of the application-level policies from application-level access controller to the network-level access controller, which controls the access with low processing load. The access requests which are decided to be *"denied"* on the network-level access controller are dropped, and thus cannot reach the application-level access controller. Therefore, those requests results in no workload at the application-level access controller. Such distribution also reduces the size of the application-level ACL which now contains access rules corresponding to only those patterns which are decided to be *'allowed'* by the network-level access controller. As we increase the number of devices, more and more access requests are controlled on the network-level access controller, resulting in significant reduction of workload relative to the input application-level ACL. This is particularly useful in CPS, which contain large number of connected devices accessing data for real-time applications. A large access workload on the application-level access controller may induce latency in the access decisions, thus degrading the application's performance. By employing the distributed control approach, therefore enables to realize the real-time access. Our approach also enhances security, in terms of early rejection of malicious activities. With only using application-level ACL, every request reaches the application-level access controller, usually located close to a resource (such as implemented inside the resource server). This may allow an attacker to compromise availability by launching DDOS. Meanwhile, with our approach, such access requests can be rejected early-on by the network-level access controller, thus minimizing the risk of DDOS and congestion of enterprise bandwidth. This may also improve the CPS performance by allocating the saved bandwidth to mission-critical and other necessary services. Hence, the approach balances the security and workloads towards overall business growth.

In our evaluation, the access granularity after policy distribution was around 15% less relative to the input ACL. Investigating further, we observed that our approach correctly mimicked the input application-level ACL in case of "DENY" decisions, but in some cases failed to mimic the "ALLOW" cases. It means that some access patterns got rejected on the network-level ACL but were originally allowed in the application-level ACL, resulting in additional mis-control and thus causing granularity degradation. The reason of this additional mis-control being that these access patterns belong to a (Source IP, Destination IP) pair which mainly contains access from devices which are intended to be rejected. As our aggregation approach uses a general network allow-rate based mechanism, in such cases, "DENY" decision would be enforced on the network-level access controller when the allow-rate is small. However, it is possible that in some cases, those access patterns which were mis-controlled on the network-access controller may represent critical workflow, such as emergency situations, or mission critical services, which should not be disrupted for maintaining business continuity. Stopping them may result in a loss of availability to those services and may degrade the reliability of the access control system. To overcome this, we conceptualize an algorithm which would intend to balance the workload, security along with the business requirements. Simplistically, to decide the access granularity, the algorithm would evaluate the negative business impacts caused by the general policy aggregation approach, and then utilize several algorithms or techniques to reduce this impact. It may select any one or more techniques depending on the use-case and dynamic environmental conditions.

One example of such an algorithm is the use of attributes of application-level policy defined with attributes such as 'location', 'resource-confidentiality', 'access needs", etc., as an additional method for policy aggregation. It utilizes the impact of these attributes on the access decision for a given access pattern. For instance, consider **confidential** resources such as *employee personal details*. If the access to such resources is mistakenly permitted (mis-controlled), then it causes a large impact of information leakage. On the other hand, resources such as *server monitoring API-calls*, *diagnostics*, *updates*, etc. are essential-workflow resources that have high **access-needs** for business continuity, and it would cause large impact on customer services and revenue, if the access to them is mistakenly denied (mis-controlled). Likewise, the mis-control impact (termed attribute impact) is estimated using access attributes, such as "resource-confidentiality" and "access-needs' respectively. The attribute-impact may dynamically decide access granularity between network-level and application-level access control. For instance, in case of large attribute impact, application-level access control can be chosen for fine-granularity. In the foresight, it is necessary to consider

which attributes lead to optimization between access granularity and workload. The implementation of such concept is left for future work. The evaluation performed in the current study assumes the input application-level ACL to be 100% accurate and the performance objective of the distributed policies is to mimic the original policy as close as possible with low access control workload. Thus, the current results are limited by the accuracy of the application-level ACL itself. As we obtained the accurate (input) ACL through a manually defined application-level ABAC policy, the evaluation of accuracy of the application-level ACL was out of the scope of this study. The current research fulfils the objective to show case a lightweight mechanism to achieve efficient fine-grained access control. The construction and evaluation of the improved method as well as the performance evaluation of the distributed ACLs in a real network scenario is a task for the future work.

VII. CONCLUSION

For optimal performance of any access control mechanism, balancing the security and access control workload is a key challenge which is explored in this research. We proposed a novel approach of achieving a lightweight fine-grained access control mechanism by distributing the application-level access control policy towards coarse-grained access controller to reduce the workload while not compromising the security. Our results show a significant reduction in the access workload compared to the input application-level ACL without degrading the security when evaluated on an artificially created desk evaluation. The study observed the occurrence of mis-control for the cases of essential access requests in the presented algorithm. The second improved method is conceptualized which intends to lower such mis-control occurred in the first method while balancing the workload. The results of our work show a promising direction towards innovative solutions for optimal performance in the field of efficient access control.

ACKNOWLEDGMENT

These research results were (partially) obtained from the commissioned research of National Institute of Information and Communications Technology (NICT) [0120101], JAPAN.

REFERENCES

[1] Z. Wang, W. Xie, B. Wang, J. Tao, and E. Wang, “Survey on recent advanced research of CPS security”, Applied Sciences, vol. 11, no. 9, pp. 3751, 2021.

[2] A. Humayed, J. Lin, F. Li, and B. Luo, “Cyber-physical systems security—A survey”, IEEE Internet of Things Journal, vol 4, no. 6, pp. 1802-1831, 2017.

[3] S.V. Sudarsan, O. Schelén, and U. Bodin, “Survey on delegated and self-contained authorization techniques in CPS and IoT”, IEEE Access, vol. 9, pp. 98169-98184, 2021.

[4] J. Kindervag, and S. Balaouras, “No more chewy centers: Introducing the zero trust model of information security” Forrester Research 3, 2010.

[5] Amoroso, and G. Edward, “From the enterprise perimeter to a mobility-enabled secure cloud”, IEEE Security and Privacy, vol. 11, no. 1, pp. 23-31, 2013.

[6] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, “Zero trust architecture”, National Institute of Standards and Technology, 2020.

[7] V. C. Hu, D. R. Kuhn, D. F. Ferraiolo and J. Voas, “Attribute-based access control”, Computer, vol. 48, no. 2, pp. 85-88, 2015.

[8] J. Bethencourt, A. Sahai, and B. Waters, “Ciphertext-policy attribute-based encryption”, IEEE symposium on security and privacy (SP’07), pp. 321-334, May, 2007

[9] S. Patil. M. Polte, K. Ren, W. tantisiriroj, L. Xiao, J. Lopez, G. Gibson, A. Fuchs, and B. Rinaldi, “Ycsb++ benchmarking and performance debugging advanced features in scalable table stores”, ACM Symposium on Cloud Computing, vol. 2, pp. 1-14, Oct. 2011.

[10] L. Touati, Y. Challal, and A. Bouabdallah, “C-cp-abe: Cooperative ciphertext policy attribute-based encryption for the internet of things”, International Conference on Advanced Networking Distributed Systems and Applications, IEEE, pp. 64-69, June, 2014.

[11] N. Oualha, and K. T. Nguyen, “Lightweight attribute-based encryption for the internet of things”, International Conference on Computer Communication and Networks (ICCCN), vol. 25, pp. 1-6, Aug. 2016.

[12] A. Shaghaghi, M.A. Kaafar, S. Scott-Hayward, S. S. Kanhere, and S. Jha, “Towards policy enforcement point as a service (peps)”, IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp. 50-55, Nov. 2016.

[13] C. DeCusatis, P. Liengtiraphan, A. Sager, and M. Pinelli, “Implementing zero trust cloud networks with transport access control and first packet authentication”, IEEE International Conference on Smart Cloud (SmartCloud), pp. 5-10, Nov. 2016.

[14] T. Ahmad, U. Morelli, and S. Ranise, “Distributed Enforcement of Access Control policies in Intelligent Transportation System (ITS) for Situation Awareness”, In Proceedings of the International Conference on Availability, Reliability and Security, vol. 17, pp. 1-10, August, 2022.

[15] E. Liu, Huawei Technologies Co Ltd, “Firewall control system based on a next generation network service and method thereof”, U.S. Patent No. 7,987,503, Jul. 2011.

[16] B. Alzahrani, , and N. Fotiou, “Enhancing internet of things security using software-defined networking”. Journal of Systems Architecture, 110, 101779, Nov, 2020.

[17] C. Tang, X. Fu, and P. Tamg, “Policy-Based Network Access and Behavior Control Management”, IEEE 20th International Conference on Communication Technology (ICCT), vol. 20, pp. 1102-1106, Oct, 2020.

[18] Z. Feng, P. Zhou, Q. Wang, and W. Qi, “A Dual-layer Zero Trust Architecture for 5G Industry MEC Applications Access Control”, IEEE International Conference on Electronic Information and Communication Technology (ICEICT), vol. 5, pp. 100-105, Aug, 2022.

[19] M. Gupta, F. Patwa, J. Benson, and R. Sandhu, “Multi-layer authorization framework for a representative Hadoop ecosystem deployment”, ACM on Symposium on Access Control Models and Technologies, vol. 22, pp. 183-190, June, 2017.

[20] M. M. Kocatürk, and T. İ. Gündem, “A fine-grained access control system combining MAC and RBACK models for XML”, Informatica, vol. 19, no. 4, pp. 517-534, 2008.

[21] B. A. Khalaf, S. A. Mostafa, A. Mustapha, A. Ismaila, M. A. Mahmoud, M. A. Jubaira, and M. H. Hassan, “A simulation study of syn flood attack in cloud computing environment”, AUS journal, vol. 26, no. 1, pp. 188-197, 2019

[22] D. Suzuki, S. Imai, and T. Katagiri, “new index of hidden workload for firewall rule processing on virtual machine”, International Conference on Computing, Networking and Communications (ICNC), pp. 632-637, Jan. 2017.

 ISBN: 978-1-68558-092-6