A Hybrid Decision-making Approach to Security Metrics Aggregation in Cloud Environments

Ming Lei*, Lianying Zhao*, Makan Pourzandi†, Fereydoun Farrahi Moghaddam†
*Carleton University †Ericsson Research Canada

***Abstract*—In cybersecurity, being able to quantify the level of security has been a long quest so that decisions can be made toward improving security. Various metrics have been proposed and applied, which can usually be computed from collected measurements. However, only certain aspects of the target system are measured corresponding to the purpose the metrics were designed for, be it software vulnerabilities or configuration errors, thus lacking a concise and clear image of the overall security of a system for the practitioners to act on, especially when it comes to large-scale or complex systems.**

We argue that overall security metrics are defined by humans based on specific security goals before they can be computed. Therefore, we propose a hybrid approach to the aggregation of well-established individual security metrics by combining machine computation with human decision making. In particular, we modify the Analytic Hierarchy Process (AHP) to reach a group decision of selected "experts", which can derive the weights of individual metrics for their aggregation. We showcase its feasibility by selecting several common metrics to measure the target systems in our testbed, and conducting an AHP survey with seventeen experts. The resulted overall security score for the target systems shows how our approach enables comparison of the overall security between those systems. By considering cloud-oriented settings, we also showcase how this approach can be applicable to today's virtualized environments.

***Index Terms*—Security Metrics, Metric Aggregation, AHP, CIS Benchmarks**

I. Introduction

In recent years, network/computer security has risen in priority for many organizations [1]. Security metrics [2] can be an effective tool to facilitate quantifying the risks to network environments and individual computers. In particular, well-defined security metrics can also assist in security-related decision making and increase the level of security awareness within the organization.

There have been a wide variety of security metrics proposed from different perspectives [2], e.g., attack surface, human factors, security incidents. Nonetheless, software vulnerabilities are still one main factor considered for security metrics, and intuitively, more vulnerabilities implies a greater risk, therefore numerous existing works [3]–[5] take into account the number of software vulnerabilities. Such works are usually based on vulnerability repositories as exemplified by the National Vulnerability Database (NVD) [6], a public repository of reported software vulnerabilities, and the Common Vulnerabilities and Exposures project (CVEs) [7]. In addition to the quantity of vulnerabilities, their severity matters as well. The Common Vulnerability Scoring System (CVSS) [8] is a widely adopted standard that is used to quantify the severity of vulnerabilities and assign a numerical value to each vulnerability, as a score available in the NVD database. The CVSS score of individual vulnerabilities has become a foundational metric for developing security analysis tools or frameworks.

However, such metrics for individual vulnerabilities do not take into account the *influence between different factors*. For example, CVSS does not provide a way to aggregate network reachability into an overall CVSS score (note that network configuration is per-deployment information, not in software code). In this case, an online system with a few vulnerabilities could be considered less vulnerable than an offline system with many vulnerabilities negatively affecting the remediation order of the two systems. Moreover, for the same security factor, individual metrics cannot reflect the *interactions between different entities* of a complex system in modern cloud-based environments. For example, one vulnerability may satisfy the precondition (e.g., gaining the admin privilege) of another vulnerability, i.e., vulnerability chaining [9]. Hence, a simplistic approach such as averaging up the CVSS scores for multiple vulnerabilities cannot reflect an exact picture of the real overall security of the system. At same time, although there is the paradigm of attack modeling, as represented by the large body of research using attack graphs [10], [11], the multi-factor/multi-entity aggregation is still not addressed[1], or put another way, the graph-based attack modeling is orthogonal to the aggregation problem.

Therefore, what is needed is a way to aggregate existing individual security metrics for multiple factors or entities into one *overall security score*. This is particularly necessary in the era of cloud computing,where massive virtual hosts are co-located with mutual influences, which makes deriving an overall security level more challenging.

The challenge. When a decision maker would like to learn about the security level of an entity, involving multiple factors/entities, how can the individual security metrics (e.g., multiple CVSS scores) of individual entities (e.g., VMs or containers) be aggregated into one overall (relative) score?

We argue that although certain individual security metrics might be widely accepted and do represent a specific risk factor (e.g., the severity of a vulnerability), overall security is still rather defined [12] by security experts (humans) followed by machine computation based on the measurements for the

[1]An attack graph always has a single node/system as the target.

defined metrics. For instance, whether one system is more secure than others depends on what security factors are taken into account (e.g., the organization may consider password policies), and their importance as defined by the experts considering their roles in the overall security. In this paper, we propose a hybrid approach to reflect this reality in the aggregation of individual security metrics. We first rely on modern decision making techniques to capture the security experts' opinions about the importance of the selected metrics (human-determined weights), calculate individual metric scores using collected measurements (machine computation), and then aggregate with weighted scores, hence the hybrid nature of the approach.

For the human-determined weights, we make use of the analytic hierarchy process (AHP) [13] approach to convert human opinions into numerical values, i.e., the weights. In particular, we introduce a new variant of AHP. Unlike the original AHP involving human opinions to determine both the metric weights and the metric scores of the systems (more subjectivity), we only use it for the purpose of weight derivation and rely on machine computation/telemetry for the metric scores. Doing so not only reduces subjectivity, but also improves scalability, enabling automation. More detailed explanation of this AHP adaptation can be found in Sections III and IV-A.

To demonstrate the feasibility of our proposed approach, we apply it to a real scenario by selecting several state-of-the-art security metrics, collecting measurements from pre-configured environments, running open-source cloud-oriented telecom functions, obtaining metric weights via AHP questionnaires and performing the calculation of an overall score accordingly. We consider the following metrics, as examples (explained in Section IV): 1) non-compliance to the Center for Internet Security (CIS) [14] benchmarks to represent container/VM misconfigurations; 2) software vulnerabilities; 3) Internet reachability; and 4) open network ports. Note that these could also be other various metrics of the decision makers' choice.

Contributions:

- We propose a hybrid approach to the aggregation of individual security metrics by combining human opinions and well-established machine-computed scores, into an overall security score, which we believe is the first of its kind.
- We retrofit the AHP approach for our purpose and design questionnaires to obtain the weights from selected human experts, who are typically security specialists.
- To demonstrate feasibility, we use a real scenario to simulate such a decision-making process in an IT environment by choosing typical security metrics and involving participants who have sufficient knowledge, and deriving an overall score with the proposed approach.

II. Background

Security metrics. To better understand the extent of something and make an informed decision about it, we first need a way to measure it, e.g., the temperature outside in $^\circ C$ for a dressing decision or the speed of a vehicle in km/h to enforce the speed limit. Likewise, it is desirable that we can take a measurement of security and represent it in a certain unit. However, the challenge stems from the complexity of security, as it is determined by damages/losses that may only happen in the future and what are considered to be threats. In reality, security is also likely to involve multiple factors instead of a single dimension. Despite various definitions, a security metric is basically a system of related dimensions to quantify the degree of security [15].

While quantifying individual dimensions can still be challenging, there have been established research areas [16], [17]. At the minimum, as Sanders suggests, relative metrics [18] can still be helpful. By contrast, there is no consensus or standard as to what security factors (dimensions) to be taken into account. Such factors can include but are not limited to password strength [19], VM co-residency [20] and misconfigurations/firewall rules [21], as well as software vulnerabilities. It highly depends on how the stakeholders define/assume their security threats. For example, IT environments with a large number of employees may have to prioritize password policies (as a human factor) compared to a small business. Therefore, we argue that security metrics is human-defined at the high level instead of machine-computed. This does not affect the fact that, at the low level, metrics of individual security factors are computed.

The CVSS metric. CVSS captures the characteristics of individual vulnerabilities, as reflected in multiple metrics [8]. Here, we only involve the exploitability metrics of its base score metrics to derive the probability of an exploit. The exploitability metrics measure the relative difficulty to exploit a vulnerability by four sub-metrics: Attack Vector (AV), Attack Complexity (AC), Privileges Required (PR), User Interaction (UI) and Scope (S). UI indicates whether user action is required for the exploit (e.g., clicking on a malicious link). Scope indicates whether an exploited vulnerability can affect other components with different ownership (i.e., "Changed" or otherwise "Unchanged"). The computed CVSS base score is a quantitative representation of the vulnerability's severity. Numerous existing works [3], [4], [17] use the CVSS as a basic metric for computer security, but vulnerabilities merely reflect what is in the software code, not covering security factors such as network configuration [3]. Therefore, the CVSS metric needs to be combined with other security aspects to avoid any one-sided results.

CIS benchmarks. The Center for Internet Security (CIS) [14] is a nonprofit organization that provides configuration baselines and best practices for a range of technology categories including operating systems, virtualization software, network devices, etc. CIS Benchmarks [22] are developed for securely configuring systems, software and networks. Each benchmark item corresponds to a specific configuration (e.g., the critical `docker.service` file's user:group should be `root:root`) so that it can be checked against to ensure compliance.

Multi-criteria decision making. Even though we can accept something to be human-decided as opposed to machine-computed, a decision made by one individual may not be reliable. To minimize potential biases and subjectivity, a common practice is to make the decision collectively and collaboratively. Multi-Criteria Decision Making (MCDM) is a paradigm to enable decision making by collecting a group of people's opinions (i.e., decision-makers) where there are (potentially conflicting) multiple decision criteria and alternatives [23]. There have been various MDCM techniques proposed, such as AHP, Fuzzy AHP, TOPSIS and ELECTRE [24].

Analytic Hierarchy Process (AHP) [13] is a widely-adopted MCDM technique. It aggregates human opinions hierarchically by decomposing complicated problems into sub-problems (i.e., criteria). AHP involves the following steps: 1) Define criteria, e.g., price, functionality and size for a TV purchase decision, where the functionality can be further broken into sub-criteria, e.g., resolution and refresh rate. 2) Capture experts' perspectives on the defined criteria, i.e., to do the pairwise comparison on the criterion importance. 3) Process the responses using a matrix, deriving criterion weights as the normalized geometric means of each row in the matrix. 4) Pairwise compare the alternatives (TVs) with respect to the defined criteria, e.g., assigning a price score to each TV. 5) Rank the alternatives with weighted criteria, as the decision.

III. Methodology

We propose a hybrid decision-making approach to the aggregation of multiple well-established computed security metrics into a single security score. In this section, we first discuss the generic approach at the high level that can be adopted by an IT environment with their customization. Then, in Section IV, we will demonstrate the entire process with a specific setting of our choice. For an IT environment to adopt this approach, the following customization is required: 1) Selection of security factors of concern as individual metrics; 2) Determination of security decision makers; 3) Tooling for the selected individual metrics.

To assess the security posture of an IT environment, the factors affecting security need to be first identified. This cannot be automated and there is no fixed set of such factors. Furthermore, certain seemingly monolithic factors can actually be broken down to sub-factors, e.g., passwords may be measured in terms of either guessability (strength) or management policy (e.g., mandatory update frequency). Correspondingly, the involved tools to collect data for the measurement also vary.

After the security factors (i.e., metrics) of concern have been selected, which would be multiple in most cases, a way to aggregate them is necessary. To achieve aggregation, one needs to understand for what percentage of each selected metric accounts in the total aggregated value, which is the weight. Note that this does not mean the relationship between metrics has to be linear, but just in terms of decision making using the AHP technique. A greater weight implies more importance. An analysis of and comparison between the selected security metrics are to be done, again, by humans.

Decision makers. We consider only personnel with sufficient security knowledge from the concerned organization or a qualified third party to be such human candidates (as opposed to doing an untargeted survey), referred to as "experts" hereafter. Examples of experts include security specialists, security scientists/researchers, senior system administrators, etc. Still, human opinions can be subjective or potentially biased, regardless of their expertise [24]. Therefore, using the AHP technique, we collect experts' options so that the collected data is credible, equivalent to collective/collaborative decision making. This way, the aggregated numeric value can represent the overall security level relatively objectively, as it combines judgements from individual qualified personnel and computed values from individual established metrics.

The workflow of our approach is shown in Figure 1. The first step is to select security metrics of the organization's concern. The rest of the process consists of two tracks: one track is to derive weights for the metrics, which only needs to be done once for a given set of metrics; the other track is to collect measurement from the target environment. In the weight derivation track (blue area), there are three steps: 1) Design AHP questionnaires, where the AHP criteria correspond to the selected security metrics; 2) Conduct AHP surveys. In this process, the experts are asked to systematically evaluate given AHP criteria and pairwise compare them. This is a crucial step since the relative importance of selected metrics is determined here; 3) Calculate metric weights using the AHP matrix [13] based on received responses. On the other hand, the measurement collection track (beige area), includes automated measurement (telemetry) and metric value calculation. In this step, the telemetry data is collected by various means such as vulnerability scanners, network tools and automated scripts. This step does the measurement which is followed by an initial automatic aggregation, e.g., statistics for multiple vulnerabilities. Then the telemetry data is converted into metric values and followed by an initial automatic aggregation, e.g., statistics for multiple vulnerabilities. By the end of the two tracks, we have both the individual computed metric values from the target environment and the derived metric weights. Lastly, they are aggregated to an overall security score. The details of each step are discussed as follows.

A. Metric Selection

Some existing works use security metrics such as vulnerability/CVSS based metrics [3], [4], network reachability [4] and virtualization based metrics [25]. They might be useful for evaluating their specific security factors, but as previously mentioned, not comprehensive. At the high level, we consider metrics from two categories: *pre-deployment* and *post-deployment*. Pre-deployment corresponds to inherent risks when the product (software/hardware) is released, e.g., software bugs, and post-deployment reflects risks introduced after the product has been deployed in an environment, likely due to misconfiguration or non-optimal settings, e.g., insecure firewall policies. While pre-deployment metrics are mostly based on software vulnerabilities, post-deployment metrics could be very diverse. Aside from measuring pass-

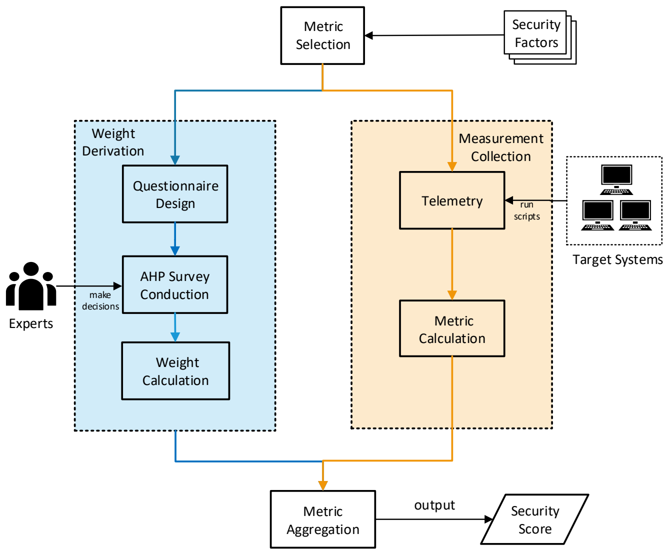


Fig. 1. Workflow of the proposed hybrid approach. Note that Metric selection and Weight Derivation are a one-time effort for a given deployment

word strength/policies, examples of other factors that can be measured include: network exposure [26] (e.g., open ports and Internet reachability), human factors [27], system configuration [21], etc. In addition to the factor selection (what to measure), the other aspect is how to measure, e.g., to measure the probability, impact of compromise, time or effort to compromise, non-compliance to guidelines, attack paths, or even configuration diversity [27]. In the case of CVSS, compromise effort is reflected which can be converted to probability.

B. Metric Weight Derivation

In this track, a survey is conducted through AHP questionnaires to collect expert opinions and derive metric weights. Note that although the derived weight is usually reused for repeated measurements, a weight refresh interval can be considered so that this step is redone to reflect the situation of the target, when necessary.

Questionnaire design. Depending on the targeted participants (experts) the first choice to make is the survey delivery format, for which options may include professional online survey websites, emails, spreadsheets, and even a dedicated mobile app. Each option has its privacy/convenience/effectiveness implications. The next choice will be the question format, which is crucial. Although AHP needs pairwise comparison as the input, the actual question format can vary, convertible to the pair comparison (e.g., from a full ranking).

Survey conduction. Different from untargeted user surveys where participants are recruited, our proposed decision making survey is conducted among internally selected "experts" or contracted third-party consultants. Thus, they are assumed to be qualified to answer the AHP questions. Once the survey delivery format is decided, the questionnaires can be distributed and responses can be collected. Note that identity disclosure is optional depending on whether anonymity is a goal in the organization. Exceptions apply to our evaluation as for research purposes we need to consider the ethical aspects.

Weight calculation. This step aims to convert the comparison results of AHP criteria into their corresponding security metric weights. The pairwise comparison results of one individual questionnaire can be represented by an AHP comparison matrix. Such a comparison matrix can be used to derive the weights of individual metrics but it only reflects one expert's judgement. For the weights to be more objective, we rely on group judgements. This can be achieved by the geometric mean method (GMM) [28], which aggregates individual comparison matrices into a single representative matrix. Therefore, the security metric weights calculated from this aggregated comparison matrix reflect the group decision.

C. Measurement Collection

Note that this track can be iterative, if we want to learn about the up-to-date security posture at certain intervals, e.g., every two weeks. The derived weights from the previous AHP step is a one-time effort and can be reused here.

Telemetry. The purpose of this step is to automatically and preferably remotely take measurements of the target system's deployment by various means such as network tools, vulnerability scanners, and custom scripts. With our cloud emphasis, we measure not only the physical machines but also the virtual machines or containers therein.

Metric value calculation. As the measured raw data could be in various formats and from multiple sources, they need to be processed before aggregation. Furthermore, the extracted information from raw data needs to be converted into numerical metric values. This needs to be automated with a script because a large number of sub-metrics may be involved.

D. Metric Aggregation

Once the weights of individual metrics are ready and the individual metrics have been calculated, we aggregate them with the formula: $\sum m_i \cdot w_i$ (as is done in [13]), where m_i is the computed individual metric value and w_i is its derived weight via group decisions. The result is the aggregated security score that represents the overall security level of a target.

IV. An Application Scenario

In this section, we set up a real test environment with several qualified participants to demonstrate the entire process shown in Figure 1. The test environment is composed of six VMs running on a VMware ESXi 7.0.2 Supermicro server with an Intel Xeon CPU D-1541 and 128GB RAM, and two PCs (Ubuntu 20.04 and Windows 11). We choose to run more telecom-oriented/cloud-native software to demonstrate applicability. For instance, VM1 (Ubuntu) and VM3 (Debian) run network function virtualization (NFV) [29] orchestration software stacks using Open Source MANO (OSM) [30] and OpenBaton [31], repectively. VM2 (Ubuntu) and VM6 (OpenSUSE) run free5GC [32] and Open5GS [33] respectively simulating a 5G mobile core network. To start with, we select certain security factors as the security metrics, and consider the following to be important in our environment.

Software Vulnerabilities (SV). This is one of the most common security metrics, identified by scanning tools based on the CVE/NVD databases with CVSS scores.

Internet Reachability (IR). Apart from a target's inherent vulnerabilities, the extent to which it is exposed to attackers also matters. We consider Internet (where external bad actors reside) reachability as a factor reflecting attacker population, e.g., all attackers in the wild compared to only LAN attackers.

Open Network Ports (OP). Active code listening on a network port indicates that a non-local attacker (including LAN within the organization) can exploit the vulnerabilities. More open ports means wider attack surface.

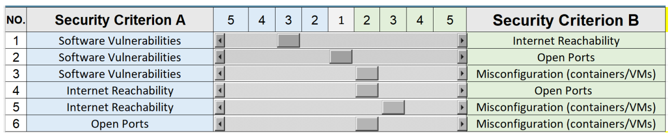
Virtualization/Containerization Misconfigurations (MC). Virtualization and containerization add to the threat model complexity, both technically and semantically. For example, a privileged user inside the container/VM usually is not and should not be privileged outside, effectively creating more privilege levels; each container/VM represents a computer with its own owner. Therefore, the degree of the container/VM configuration's (non-)compliance to certain best practices can serve as a security metric.

A. Conducting AHP Surveys

In a research setting, we demonstrate how the collective decision making using AHP surveys could be conducted in an IT environment.

Participants. The invited participants are divided into 1) Academia: faculty members (professors) or security lab graduate students; 2) Industry: security specialists or senior system administrators in a well-known telecom company. Considering our special requirements that each response must be from a person with sufficient security knowledge regarding the involved technologies, only invited participants are allowed to take this survey. Therefore, we did not use an online survey website and share URLs which could be disseminated (no way to verify eligibility). Given the small number of participants, questionnaires were communicated by email to the experts for pairwise comparison.

Questionnaires. We designed one main questionnaire for the four metrics (SV, IR, OP and MC). We use the CIS Docker/Kubernetes benchmarks [14] to quantify misconfigurations, with sections as sub-metrics (detailed in Section IV-D), so we also created two additional questionnaires for them. A snippet of our designed questionnaires is shown in Figure 2.



NO.	Security Criterion A	5	4	3	2	1	2	3	4	5	Security Criterion B
1	Software Vulnerabilities			●							Internet Reachability
2	Software Vulnerabilities					●					Open Ports
3	Software Vulnerabilities						●				Misconfiguration (containers/VMs)
4	Internet Reachability						●				Open Ports
5	Internet Reachability							●			Misconfiguration (containers/VMs)
6	Open Ports						●				Misconfiguration (containers/VMs)

Fig. 2. The main questionnaire pairwise-comparing SV, IR, OP and MC

We map the AHP terminology [13] to our scenario: there is a *goal* based on which the decision is made, e.g., in our case, security. This goal can be further broken down into multiple *criteria*, e.g., the individual security metrics like CVSS. There are also *alternatives* the decision will pick the best from, e.g., in our case, VM1, VM2, etc. Unlike the original AHP, since pairwise comparison is not needed for the alternatives (e.g., VM1 vs. VM2 in terms of CVSS), we only use AHP to determine the criterion weight by pairwise-comparing the criteria with respect to the goal, e.g., CVSS vs. open network ports in terms of security.

Weight calculation. Each AHP with one participant will result in one comparison matrix for the criterion weights. We demonstrate how to calculate the weights using this matrix.

$$A = \left(\begin{array}{ccccc} & C1 & C2 & C3 & C4 \\ C1 & 1 & 5 & 1 & 1/3 \\ C2 & 1/5 & 1 & 1/3 & 1/5 \\ C3 & 1 & 3 & 1 & 1/3 \\ C4 & 3 & 5 & 3 & 1 \end{array}\right)$$

The comparison matrix A above is converted from one of our questionnaire responses. The headings of rows and columns represent our selected four security metrics (software vulnerabilities—C1, Internet reachability—C2, open network ports—C3 and misconfigurations—C4). The value A_{ij} reflects the relative importance intensity from 1 (equal) to 9 (extreme), downscaled to 1 to 5 in the questionnaire for simplicity (intermediate intensities 2, 4, 6 and 8 are excluded). For instance, $A_{12} = 5$ can be interpreted to mean that software vulnerabilities is strongly more important than Internet reachability. Each criterion weight is calculated by the row's normalized geometric mean method. Compared to arithmetic mean, geometric mean can reduce the effect of extreme values [34]. The geometric mean of the first row is calculated by $\sqrt[4]{1 \times 5 \times 1 \times 1/3} \approx 1.136$. Therefore, the normalized (between 0 and 1) geometric means for each row are the weights: 0.224, 0.067, 0.197 and 0.511.

Note that the calculation result above is only for one individual participant. As we need a relatively unbiased group decision, the final AHP criterion weights are calculated by aggregating all individual comparison matrices. To this end, we use the aggregation of individual judgments (AIJ) method [35]. In the aggregated matrix, the element is also calculated by geometric mean, which is $[\prod_{k=1}^{n} a_{ijk}]^{1/n}$, where a_{ijk} is the pairwise comparison result between criteria i and j for group member k, and n is the total number of members. This achieves the aggregation of individual judgements.

This weight calculation process can be done in multiple ways such as offline/online AHP utilities and scripts. In our experiment, we employ an AHP Excel template created by Goepel [36] to automate the calculation process.

B. Measuring Vulnerabilities

In general, vulnerability scanning is to match deployment information with a database of reported vulnerabilities such as the NVD. Examination of state-of-the-art scanners, e.g., Nessus [37], openscap-scanner [38] and Nmap [39], shows that deployment information can be collected in three ways: 1) software name and version from a local package manager; 2) signature/version extracted from program binary files; and 3) scanning network ports for listening services. Nmap relies on port scanning so is limited to identifying only vulnerabilities exploitable via network. Program binary files may be significantly redundant as the majority of the files are not in use (e.g., backup copies or previous versions). Therefore, we choose to use the CVE Binary Tool [40] to scan for up-to-date vulnerabilities based on the local package manager.

To calculate the software vulnerability metric value, we consider two aspects: vulnerability quantity and exploit probability of each vulnerability. The exploit probability can be computed as proposed by Zhang et al. [41], using the CVSS exploitability metrics shown in Table I and converting them to a probability.

TABLE I. Exploitability Metrics Value for CVE-2021-32036

AV	AC	PR	UI	Scope
Network(0.85)	Low(0.77)	Low(0.62)	None(0.85)	Unchanged

We use CVE-2021-32036 as an example to show the exploit probability calculation from Table I: $P = 2.11 \times AV \times AC \times PR \times UI \approx 0.728$, where 2.11 is a constant and the CVSS vectors are explained in Section II. Thus, the exploit probability of CVE-2021-32036 is 0.728.

Finally, following the derivation of exploit probability from CVSS vectors [41], we aggregate the normalized vulnerability count and the arithmetic mean of exploit probability using multiplication:

$$P_{mean} \times \frac{n_v}{N_{v\_max}} \tag{1}$$

where P_{mean} is the arithmetic mean of exploit probability for all found vulnerabilities on a system; n_v is the number of found vulnerabilities; N_{v_max} is $max(n_v)$ among all systems. The normalized vulnerability count and mean of exploit probability are presented in column SV and column EP of Table III in Section V-A.

C. Measuring Open Ports

Even if a vulnerability exists, it will only be exploitable to local malware already landed on the system. A remote attacker will not be able to exploit it until it is exposed through an open network port on which the vulnerable code listens. Therefore, we consider open ports to be orthogonal to software vulnerabilities and a post-deployment attack vector. Note that firewall rules also determine which ports are exposed and to which network entities.

We use Nmap [39] to scan all IPv4 and IPv6 port numbers for both TCP and UDP. The open port metric score is calculated by the number of open ports over the maximum number of open ports among all tested systems:

$$n_o / N_{o\_max} \tag{2}$$

Although in theory, there are 65,536 ports for TCP and UDP respectively, the number of open ports on an actual system is usually a very small portion of the range and the rest are closed. For the open port metric to be reasonable, we define the maximum number of ports (N_{o_max}) to be the total of unique open ports among all tested systems. Hence, the calculated value is relative, to the deployment-specific max. As PC2 has the maximum number of open ports (46), given that VM1 has 30 open ports, then VM1's open port score is $30/46 \approx 0.653$.

D. Measuring Misconfigurations

To reflect the cloud emphasis in our proposed approach, we choose to quantify the misconfiguration of containerization/clustering aspect of the system (although there exist other forms of misconfiguration to evaluate). To this end, we employ the CIS benchmarks, which are recommended configuration practices, to evaluate the non-compliance of the target system. Our experiment uses CIS Docker Benchmarks v1.3.1 and CIS Kubernetes Benchmarks v1.6.0, listing 116 and 124 security recommendations respectively. The recommendations are grouped into sections based on components or security rules, see Table II.

We use two open-source shell scripts [42], [43] to test the compliance of Docker and Kubernetes configurations respectively. Note that not all recommendations are applicable to a system, e.g., if Docker Swarm is not used in a configured Docker, the corresponding tests are skipped. For each tested recommendation, there are four types of results: 1) *Pass*, indicating compliance; 2) *Fail*, indicating non-compliance; 3) *Not applicable*, e.g., testing if file permission is set to read-only but the file does not exist; 4) *Human action needed*, e.g., recommending to run the Docker daemon as a non-root user. We convert the results to pass or fail: if an item is not applicable, we treat it as pass because an inapplicable item does not have security impact. On the contrary, if an item needs human action, we regard it as fail because the worst-case scenario should be assumed in security (e.g., the user may not take the required action).

Misconfiguration score calculation. The mis-configuration score of Docker/Kubernetes is calculated by summing up all weighted fail rates of individual sections, which can be represented by: $\sum_{i=1}^{n} F_i \cdot W_i$, where F is fail rate, W is section weight and i is the index of individual sections. The fail rate is calculated as the number of failed items divided by the total number of items. The section weights are derived from the two benchmark questionnaires via AHP. Table II shows the test results of VM1's Docker and Kubernetes configurations. The first column lists section names (components) and the second column lists the number of failed items over the total.

V. Security Metrics Aggregation Results

In this section, we briefly report our experiment's results.

A. Measurement Results

It takes approximately 3-5 minutes to scan for vulnerabilities on a system with around 1,800 packages and 1-2 minutes to perform the Docker/Kubernetes tests (tool installation time not included). Other tasks are orders of magnitude faster. Note that timing telemetry data collection is only to have an idea of how long measurement collection can take in our experiment setup and may vary with metrics selected, tools used, software packages installed, etc.

TABLE II. CIS Benchmark results (misconfiguration) of VM1

Docker Sections	Failed / Total
S1. Host Configuration	15 / 20
S2. Docker Daemon Configuration	9 / 18
S3. Docker Daemon Configuration Files	0 / 24
S4. Container Images and Build File Configuration	8 / 11
S5. Container Runtime Configuration	11 / 31
S6. Docker Security Operations	0 / 2
S7. Docker Swarm Configuration	0 / 10
Kubernetes Sections	**Failed / Total**
S1. Control Plane Components	18 / 65
S2. etcd	1 / 7
S3. Control Plane Configuration	1 / 3
S4. Worker Nodes	13 / 23
S5. Policies	0 / 26

TABLE III. Measurement data for the main criteria

	SV[1]	EP[2]	IR[3]	OP[4]	MC sub-metrics[5]
VM1	141 (0.199)	0.692	1	30 (0.652)	Kubernetes + Docker
VM2	358 (0.506)	0.658	1	18 (0.391)	-
VM3	68 (0.096)	0.587	1	17 (0.370)	Docker
VM4	708 (1)	0.586	1	16 (0.348)	Kubernetes
VM5	708 (1)	0.586	1	11 (0.239)	Kubernetes + Docker
VM6	236 (0.333)	0.611	0	14 (0.304)	-
PC1	144 (0.203)	0.700	1	11 (0.239)	Docker
PC2	266 (0.376)	0.514	1	46 (1)	-

[1]Software Vulnerabilities [2]Exploit Probability [3]Internet Reachability [4]Open Ports [5]Cloud/container misconfiguration (details in Table IV)

The measurement data is shown in Table III. Column SV shows the number of vulnerabilities in a system and the normalized value (divided by the max); Column EP shows the mean of exploit probability of all vulnerabilities in a system; Column IR shows Internet reachability (Boolean) and OP shows the number of open ports in a system and its normalized value. Column MC shows whether the system has Docker/Kubernetes configured. Per Formula (1), the vulnerability score of VM1 is $0.199 \times 0.692 \approx 0.138$. Per Formula (2), its open port score is $30/46 \approx 0.652$.

B. Survey Results

We received 17 responses for the main questionnaire, 13 for Docker and 12 for Kubernetes. Among these responses, there are 5 from industry and 12 from academia (5 professors and 7 graduate students). As a known issue of pair-wise comparisons, the response can be potentially inconsistent between pairs [36]. We added a consistency indicator to highlight the top inconsistent pairs and displayed a *consistency ratio* (CR). A higher CR means a higher degree of conflict in the questionnaire. As Docker involved 7 criteria (the more pairs the more difficult to ensure consistency), we chose to accept response with a CR below 0.2 [44]. For the other two questionnaires, we use a 0.1 threshold as recommended by Saaty [13]. Eventually, we have 15 main, 12 Docker and 10 Kubernetes questionnaire results. The calculated weights of individual security metrics and sub-metrics are shown in the second column of Table IV (Sx = Sections).

TABLE IV. Derived metric weights and measured values

	Sx	Weights	VM1	VM2	VM3	VM4	VM5	VM6	PC1	PC2
Docker	S1	11.8%	.75	-	.75	-	.7	-	.7	-
	S2	17.5%	.5	-	.5	-	.5	-	.5	-
	S3	16.0%	0	-	0	-	0	-	0	-
	S4	16.0%	.73	-	.82	-	.64	-	.73	-
	S5	14.8%	.35	-	.42	-	0	-	0	-
	S6	13.0%	0	-	0	-	0	-	0	-
	S7	10.9%	0	-	0	-	0	-	0	-
	Sub-score		.34	0	.37	0	.27	0	.29	0
Kubernetes	S1	21.8%	.28	-	-	.51	Ø	-	-	-
	S2	14.1%	.14	-	-	.71	Ø	-	-	-
	S3	13.8%	.33	-	-	.33	Ø	-	-	-
	S4	23.7%	.57	-	-	Ø	.48	-	-	-
	S5	26.6%	0	-	-	0	Ø	-	-	-
	Sub-score		.26	0	0	.26	.11	0	0	0
Main	SV	34.3%	.14	.33	.06	.59	.59	.20	.14	.19
	IR	22.4%	1	1	1	1	1	0	1	1
	OP	15.2%	.65	.39	.37	.35	.24	.30	.24	1
	MC	28.1%	.3	0	.19	.13	.19	0	.15	0
	Overall Score		.46	.40	.35	.51	.52	.11	.35	.44

All results are rounded to 2 decimal places

C. Aggregation Results

The calculated metric values of individual systems are in columns 3 to 10 of Table IV. The sub-score shows weighted sum of Sx scores for Docker and Kubernetes respectively.

VM1 has both Kubernetes master and worker nodes. VM4 (master node) and VM5 (worker node) form a cluster. VM3 and PC1 have only Docker. To achieve a fair comparison of all 8 systems, we apply the following rule: both Docker and Kubernetes will be assumed present, where absence is naturally assigned a 0 (no risk due to inapplicability). Then the misconfiguration (MC) score of individual systems is the mean value: (Docker sub-score + Kubernetes sub-score)/2. Therefore, the sub-score of a system without Docker/Kubernetes will be 0, e.g., the Docker sub-score of VM2, VM4, VM6 and PC2. The final overall scores are presented in the last row. It can be seen that VM5 is the most insecure (narrowly followed by VM4) due to its high SV scores (heavily weighted), and VM6 is the most secure one unsurprisingly because it has no internet connection and the Docker/Kubernetes is not configured on it.

VI. Related Work

Amid the immense body of reserach of security metrics, works that address metric aggregation are not common. HARMs [45] uses the Attack Graph (AG, with topology/path information) to represent network-level attacks and the Attack Tree (AT, which is the attacker goal's breakdown) to represent host-level attacks. For such aggregation proposals, the missing aspect—which we argued about in Section I—is human decisions. Early uses of AHP in security metrics (e.g., Moeti and Kalema [46], Turskis et al. [47], Sun et al. [4]) either purely rely on human judgements (not using well-established computed metrics) or do not address metric aggregation. There have also been attempts to automate aggregation using machine learning, e.g., Beck and Rass [12] propose to use neural networks to resemble human expert's decision making, but no details of how this could be achieved have been disclosed in particular when "experts" can possess arbitrary opinions.

VII. Concluding Remarks

We presented an approach to aggregate selected security metrics into a single numerical score to represent the overall security level in cloud/virtualized environments. Through conducting the AHP survey and measurement collection with a telecom/cloud-oriented environment we set up, we showed the feasibility of combining human opinions and machine computation to minimize subjectivity. It should be noted that the comparison is possible between systems against the same selected security metrics, or across different time frames of the same system following changes.

Metric selection and weight derivation via AHP are a one-time process, which remains reusable across repeated runs and environments of similar security concerns. As future work, we may also address hierarchical aggregation, e.g., between a VM and its host, where the challenge could be recursive derivation. We hope the current work can shed light on future research on security metric aggregation.

References

[1] Kaseya, "2021 IT Operations Report," Tech. Rep., 2021. [Online]. Available: https://www.kaseya.com//wp-content/uploads/dlm_uploads/2021/06/Kaseya-Whitepaper-2021-IT-Ops-Survey-Report.pdf

[2] G. O. Yee, "Chapter e32 - security metrics: An introduction and literature review," in *Computer and Information Security Handbook (Third Edition)*. Boston: Morgan Kaufmann, 2013, pp. e57–e70.

[3] P. Cheng, L. Wang, S. Jajodia, and A. Singhal, "Aggregating CVSS base scores for semantics-rich network security metrics," in *SRDS'12*, 2012, pp. 31–40.

[4] K. Sun, S. Jajodia, J. Li, Y. Cheng, W. Tang, and A. Singhal, "Automatic security analysis using security metrics," in *MILCOM'11*, Baltimore, Maryland, US, 2011, pp. 1207–1212.

[5] A. Alkussayer and W. H. Allen, "Security risk analysis of software architecture based on AHP," in *7th International Conference on Networked Computing*, 2011, pp. 60–67.

[6] NIST, "National vulnerability database," available at: https://nvd.nist.gov/.

[7] The MITRE Corporation, "Common vulnerabilities and exposures," available at: https://www.cve.org/.

[8] P. Mell, K. Scarfone, and S. Romanosky, "Common vulnerability scoring system," *IEEE Security & Privacy*, vol. 4, no. 6, pp. 85–89, 2006.

[9] S. Jajodia, S. Noel, and B. O'berry, "Topological analysis of network attack vulnerability," in *Managing cyber threats*. Springer, 2005, pp. 247–266.

[10] K. Kaynar, "A taxonomy for attack graph generation and usage in network security," *Journal of Information Security and Applications*, vol. 29, pp. 27–56, 2016.

[11] M. U. Aksu, K. Bicakci, M. H. Dilek, A. M. Ozbayoglu, and E. i. Tatli, "Automated generation of attack graphs using NVD," in *ACM CODASPY'18*, Tempe, AZ, USA, 2018, p. 135–142.

[12] A. Beck and S. Rass, "Using neural networks to aid CVSS risk aggregation—an empirically validated approach," *Journal of Innovation in Digital Ecosystems*, vol. 3, no. 2, pp. 148–154, 2016.

[13] R. Saaty, "The analytic hierarchy process—what it is and how it is used," *Mathematical Modelling*, vol. 9, no. 3, pp. 161–176, 1987.

[14] Center for Internet Security, available at: https://www.cisecurity.org/.

[15] O. S. Saydjari, "Is risk a good security metric?" in *Proceedings of the 2nd ACM Workshop on Quality of Protection*, ser. QoP'06, Alexandria, Virginia, USA, 2006, p. 59–60.

[16] D. W. Woods and R. Böhme, "Sok: Quantifying cyber risk," in *IEEE Symposium on Security & Privacy (SP'21)*, 2021, pp. 211–228.

[17] J. Homer, "A sound and practical approach to quantifying security risk in enterprise networks," 2009.

[18] W. H. Sanders, "Quantitative security metrics: Unattainable holy grail or a vital breakthrough within our reach?" *IEEE Security & Privacy*, vol. 12, no. 2, pp. 67–69, 2014.

[19] X. de Carné de Carnavalet and M. Mannan, "A large-scale evaluation of high-impact password strength meters," *ACM Trans. Inf. Syst. Secur.*, vol. 18, no. 1, pp. 1:1–1:32, 2015.

[20] M. G. M. M. Hasan and M. A. Rahman, "Protection by detection: A signaling game approach to mitigate co-resident attacks in cloud," in *CLOUD'17, Honolulu, HI, USA, June 25-30, 2017*, 2017, pp. 552–559.

[21] F. Cuppens, N. Cuppens-Boulahia, and J. Garcia-Alfaro, "Misconfiguration management of network security components," *arXiv preprint arXiv:1912.07283*, 2019.

[22] Center for Internet Security, "CIS benchmarks," available at: https://www.cisecurity.org/cis-benchmarks/.

[23] A. Jayant and J. Sharma, "A comprehensive literature review of MCDM techniques electre, promethee, vikor and topsis applications in business competitive environment," *International Journal of Current Research*, vol. 10, pp. 65 461–65 477, February 2018.

[24] M. Aruldoss, T. M. Lakshmi, and V. P. Venkatesan, "A survey on multi criteria decision making methods and its applications," *American Journal of Information Systems*, vol. 1, no. 1, pp. 31–43, 2013.

[25] E. Caron, A. D. Le, A. Lefray, and C. Toinard, "Definition of security metrics for the cloud computing and security-aware virtual machine placement algorithms," in *CyberC'13*, 2013, pp. 125–131.

[26] T. Setzler and X. Mountrouidou, "IoT metrics and automation for security evaluation," in *IEEE CCNC'21*, 2021, pp. 1–4.

[27] J.-H. Cho, S. Xu, P. M. Hurley, M. Mackay, T. Benjamin, and M. Beaumont, "STRAM: Measuring the trustworthiness of computer-based systems," *ACM Comput. Surv.*, vol. 51, no. 6, feb 2019.

[28] T. L. Saaty and L. G. Vargas, "Dispersion of group judgments," *Mathematical and Computer Modelling*, vol. 46, no. 7, pp. 918–925, 2007, Decision Making with the Analytic Hierarchy Process and the Analytic Network Process.

[29] I. Giannoulakis, E. Kafetzakis, G. Xylouris, G. Gardikis, and A. Kourtis, "On the applications of efficient NFV management towards 5g networking," in *5GU'14*, Levi, Finland, 2014, pp. 1–5.

[30] ETSI, "Open source MANO," available at: https://osm.etsi.org/.

[31] "Openbaton," available at: https://openbaton.github.io/documentation/.

[32] free5GC.org, "Free 5GC," available at: https://www.free5gc.org/.

[33] S. Lee, "Open 5GS," available at: https://open5gs.org/open5gs/docs/.

[34] R. Aull-Hyde, S. Erdogan, and J. M. Duke, "An experiment on the consistency of aggregated comparison matrices in AHP," *European Journal of Operational Research*, vol. 171, no. 1, pp. 290–295, 2006.

[35] T. L. Saaty, *Group Decision Making and the AHP*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1989, pp. 59–67.

[36] K. D. Goepel, "Implementing the analytic hierarchy process as a standard method for multi-criteria decision making in corporate enterprises – a new AHP excel template with multiple inputs." RSW publications, June 2013.

[37] Nessus, available at: https://www.tenable.com/products/nessus.

[38] OpenSCAPBase, available at: https://www.open-scap.org/tools/openscap-base/.

[39] Nmap, available at: https://nmap.org/.

[40] "CVE-binary-tool," 2022, available at: https://cve-bin-tool.readthedocs.io/en/latest/.

[41] H. Zhang, F. Lou, Y. Fu, and Z. Tian, "A conditional probability computation method for vulnerability exploitation based on CVSS," in *IEEE DSC'17*, 2017, pp. 238–241.

[42] "CIS docker security benchmarks," available at: https://github.com/docker/docker-bench-security.

[43] "CIS kubernetes security benchmarks," available at: https://github.com/neuvector/kubernetes-cis-benchmark.

[44] N. M. Scala, K. Needy, and J. Rajgopal, "Using the analytic hierarchy process in group decision making for nuclear spare parts," 2010.

[45] J. Hong and D.-S. Kim, "Harms: Hierarchical attack representation models for network security analysis," 2012.

[46] M. Moeti and B. M. Kalema, "Analytical hierarchy process approach for the metrics of information security management framework," in *CICSyN'14*, 2014, pp. 89–94.

[47] Z. Turskis, N. Goranin, A. Nurusheva, and S. Boranbayev, "Information security risk assessment in critical infrastructure: A hybrid mcdm approach," *Informatica*, vol. 30, no. 1, pp. 187–211, 2019.