Toward Automated Vulnerability Handling

Takeshi Takahashi*, Hideaki Kanehara*†, Masaki Kubo* Noboru Murata*†, and Daisuke Inoue*
*National Institute of Information and Communications Technology, Tokyo, Japan
†Waseda University
E-mail: takeshi_takahashi@ieee.org

***Abstract*—To maintain acceptable levels of security, organizations must manage their IT assets and related vulnerabilities. However, this can be a considerable burden because their resources are often limited. We have been working on a technique and system architecture that monitor the vulnerability of the IT assets on an organization's administrative networks. It determines identifiers of IT assets to locate vulnerability information pertaining to the assets. When vulnerability information pertaining to the assets is found, it changes the network configuration to mitigate any possible impacts on the organization. However, there are several hurdles that hinder the development of such techniques. This paper discusses the main such issues.**

I. Introduction

To cope with the increasing amount of cyber threats, organizations need to take care of software vulnerabilities on their networks. Considering their constrained resources, it is desirable to automate and streamline IT asset and vulnerability management operations.

We have been working on the automation of IT asset vulnerability management on administrative networks, using open data and standardized tools. Common Platform Enumeration (CPE) and Common Vulnerabilities and Exposures (CVE) [1] provide identifiers, i.e. CPE identifiers (CPE-ID) and CVE identifiers (CVE-ID). A **CPE-ID** identifies classes of applications, operating systems (OSs), and hardware devices present among a company's computing assets. Along with the CPE specifications, a list of CPE-IDs is published online as an official CPE dictionary[2]. We can easily locate CPE-IDs using the dictionary. Likewise, a **CVE-ID** identifies publicly known cybersecurity vulnerabilities. Along with security identifiers, we use online vulnerability information repositories, such as the National Vulnerability Database (NVD) [3] and Japan Vulnerability Notes (JVN) [4].

Based on these open data and standardized tools, we have been working on a mechanism that automatically collects IT asset information, determines its identifiers, identifies relevant vulnerability information, and then implements actuations [5]. This paper describes the overview of the mechanism and discusses the issues that may become wheels for expediting automated vulnerability handlings.

II. Automated Vulnerability Monitoring

Figure 1 shows the typical networks we assume. We define the following five roles. A **terminal** is any terminal used by employees to carry out their duties within an organization. A software module called an "agent" is installed on the terminal.

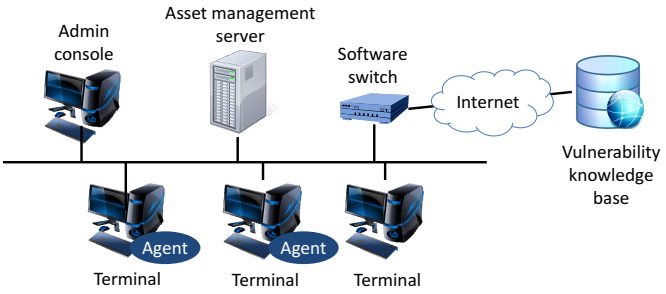

Fig. 1. Typical network

An **asset management server** is a server that stores asset information collected by the agents or collected by the server itself. This role also encompasses assigning an identifier for each piece of asset information and storing the identifier along with the asset information itself. A **vulnerability knowledge base** is a database that stores vulnerability information related to vulnerability [6]. An **administration console** is a terminal used by the system administrator. It receives an alert when a vulnerability is discovered. A **software switch** is a switch that accepts actuation commands.

Figure 2 shows the process flow of our mechanism. The process begins by collecting information on the IT assets within an organization's administrative networks. The asset management server collects information by querying the agents installed on terminals. The types of information collected by the system include the software name, version, and vendor name of the OS and installed applications, as well as IP and MAC addresses.

The server then generates IT asset identifiers, i.e. CPE-IDs, from the IT asset information. In order to determine a CPE-ID, we introduce a parameter called the matching rate, which is calculated as the percentage of the characters that match the query. We choose the CPE-ID that shows the highest matching rate, although we judge that no CPE-ID is found if we do not find a CPE-ID with a matching rate that is higher than a certain threshold value. Upon finding the CPE-ID, we add it to the aforementioned asset information. [7]

The server then queries the vulnerability knowledge base about the identifiers so that it can find vulnerability information pertaining to the IT assets. In our prototype, we retrieve only NVD information through the cybersecurity knowledge base for simplicity. We search for a match in the NVD file, inside the <vuln:product>element, and extract the CVE-IDs

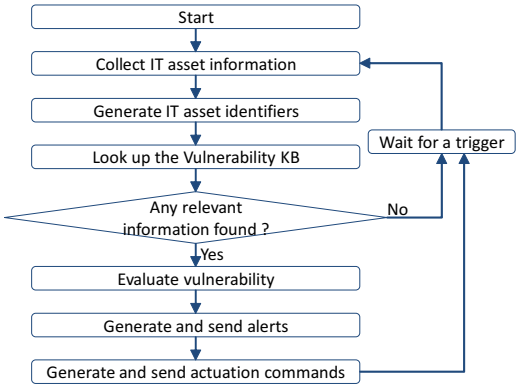

Fig. 2. Process flow

of the matched entries. If a matching NVD is found, it is determined that a vulnerability exists in that particular asset.

If any such information is found, the server evaluates the severity of the information, generates alerts, and sends them to the system administrator if needed. At the same time, the server generates actuation commands and sends them to network devices that need to change their configurations to mitigate any possible consequences of the vulnerabilities[1]. The actuation commands can, for instance, prune the vulnerable device from the network or switch the device's current network to a quarantine network by changing the configurations of network devices. Please note that our current implementation uses Ansible over SSH connections to change the configurations of network devices.

After the actuation commands are sent, the server then waits for the next trigger to run this process again; the trigger could be a time out or any evincive instructions by administrators. This operation will continue so that the server can continuously monitor the vulnerabilities of the IT assets.

III. Issues on automated vulnerability handling

There are several issues that could be better addressed for expediting automated vulnerability handling.

Actuation instructions: Actuation commands are sent to the network appliances through their own interfaces, such as NETCONF and REST APIs. As discussed in [8], those REST APIs differ among the vendors and appliances. Even if NETCONF is supported, its supported data models differ among vendors and thus it still allows room for custom descriptions among the appliances. There are even appliances without useful network APIs, and thus we need to access them through SSH connections.

Automated severity determination: When implementing countermeasures, severe vulnerability needs to be addressed first. For this purpose, we need to know the severity of vulnerabilities. We currently have the Common Vulnerability Scoring System (CVSS) score, but it is calculated manually at present. We have been working on the automated calculation of the score using the descriptions of vulnerability notes, and it is feasible by using machine learning. However, the current description information does not necessarily contain detailed information for automated calculation. More detailed descriptions will help with the more accurate calculation of such scores. There is vulnerability description ontology [9]. By using it, the description becomes more detailed and helps machine learning algorithms to produce accurate scores.

Scoring method: The aforementioned CVSS is widely used these days, but the score is not desirable for automated actuations because many of the scores indicate a critical situation, and most of them will not be exploited. A greater distinction between vulnerabilities likely to be exploited and those not likely to be exploited should be differently scored to avoid unnecessary actuations.

Official CPE dictionary: Our algorithm determines CPE-IDs using the official CPE dictionary, but it does not list all CPE-IDs; indeed many unlisted CPE-IDs are used in NVD. Moreover, the CPE dictionary sometimes contains inaccurate information[2]. Having said that, the CPE-ID is already used even if it violates the naming rule. More CPE-IDs should be accurately listed in the official dictionary so that algorithms can accurately determine CPE-IDs.

IV. Summary

Several issues stand against automated vulnerability handling, including detection and actuation. We believe that reinforcing common interfaces and structured data expedites further developments of security operation automation techniques. Please note that this work was supported by a grant from the Japan Society for the Promotion of Science (JSPS KAKENHI grant number 17K12699).

References

[1] ITU-T. *Common vulnerabilities and exposures*. International Telecommunications Union, Geneva, Switzerland, 2014. ITU-T Recommendation X.1520.
[2] National Institute of Standards and Technology. Official common platform enumeration (cpe) dictionary. https://nvd.nist.gov/cpe.cfm, 2016.
[3] National Institute of Standards and Technology. National Vulnerability Database (NVD). http://nvd.nist.gov/, 2016.
[4] JPCERT/CC and IPA. Japan Vulnerability Notes. http://jvn.jp/, 2014.
[5] T. Takahashi, D. Miyamoto, and K. Nakao. Toward automated vulnerability monitoring using open information and standardized tool. In *IEEE International Conference on Pervasive Computing and Communications*. 2016.
[6] Takahashi T, Panta B, Kadobayashi Y, et al. Web of cybersecurity: Linking, locating, and discovering structured cybersecurity information. *International Journal of Communication Systems*, 2017.
[7] T. Takahashi, and D. Inoue. Generating software identifier dictionaries from vulnerability database. In *14th Annual Conference on Privacy, Security and Trust*, 2016.
[8] Takahashi T, Tsuda Y, Suzuki K, et al. Toward automated threat detection and actuation. In *Coordinating Attack Response at Internet Scale*. 2019.
[9] NIST. *Vulnerability Description Ontology(VDO)*. National Institute of Standards and Technology, Maryland, USA, 2016. NIST Interagency Report 8138.

[1] In practice, we may prepare a step where a human operator reviews the actuation commands and authorizes their execution.

[2] For instance, Hidemaru's CPE-ID violated the rules. The CPE-ID in the official CPE dictionary is "cpe:/a:hidemaru:editor:8.00," but this is against the naming rule of the CPE specification. According to the naming rule, we need to concatenate the vendor name and product name instead of the vendor name and type of product. Accordingly, the correct CPE-ID for Hidemaru should be "cpe:/a:SaitohKikaku:Hidemaru:8.00."