Cyber-Physical Systems Security Based on A Cross-Linked and Correlated Vulnerability Database*

Yuning Jiang[1][0000−0003−4791−8452], Yacine Atif[1][0000−0002−7312−9089], and Jianguo Ding[1][0000−0002−8927−0968]

School of Informatics, University of Skövde, Sweden
{firstname.lastname}@his.se

Abstract. Recent advances in data analytics prompt dynamic data-driven vulnerability assessments whereby data contained from vulnerability-alert repositories as well as from Cyber-physical System (CPS) layer networks and standardised enumerations. Yet, current vulnerability assessment processes are mostly conducted manually. However, the huge volume of scanned data requires substantial information processing and analytical reasoning, which could not be satisfied considering the imprecision of manual vulnerability analysis. In this paper, we propose to employ a cross-linked and correlated database to collect, extract, filter and visualise vulnerability data across multiple existing repositories, whereby CPS vulnerability information is inferred. Based on our locally-updated database, we provide an in-depth case study on gathered CPS vulnerability data, to explore the trends of CPS vulnerability. In doing so, we aim to support a higher level of automation in vulnerability awareness and back risk-analysis exercises in critical infrastructures (CIs) protection.

Keywords: Cyber-Physical System Security · Vulnerability Analysis · Correlated Database Management · SCADA.

1 Introduction

Assessing vulnerabilities supports analytics-based decision-making processes to protect cyber-physical systems (CPSs) such as those prevalent in Critical Infrastructures (CIs), in order to focus on specific risks with varying degrees of impact-severity. The notion of risk remains elusive, as evidenced by the increasing investigations on CI security operations centres (SOCs) where analysts employ various detection, assessment, and defence mechanisms to monitor security events [1]. Normally, SOCs involve multiple automated security tools such as network vulnerability scanners and CVSS[1] calculator, combined with analysis of

* *This research has been supported in part by the EU ISF Project A431.678/2016 ELVIRA (Threat modeling and resilience of critical infrastructures), coordinated by Polismyndigheten/Sweden.

[1] https://www.first.org/cvss/specification-document

data contained and produced by CPS operations as well as alerts retrieved from vulnerability repositories such as Common Vulnerability Exposure (CVE)[2]. The security operators need further to forecast the match between these vulnerabilities and the state of intricate CIs layer networks, while prioritising patching investments using an accurate and a streamlined vulnerability-scoring mechanism [10]. This process is illustrated in Part (a) of Fig. 1, which shows the central role of security operators in SOCs and their need for support to keep pace with dynamically evolving vulnerability-alert repositories.

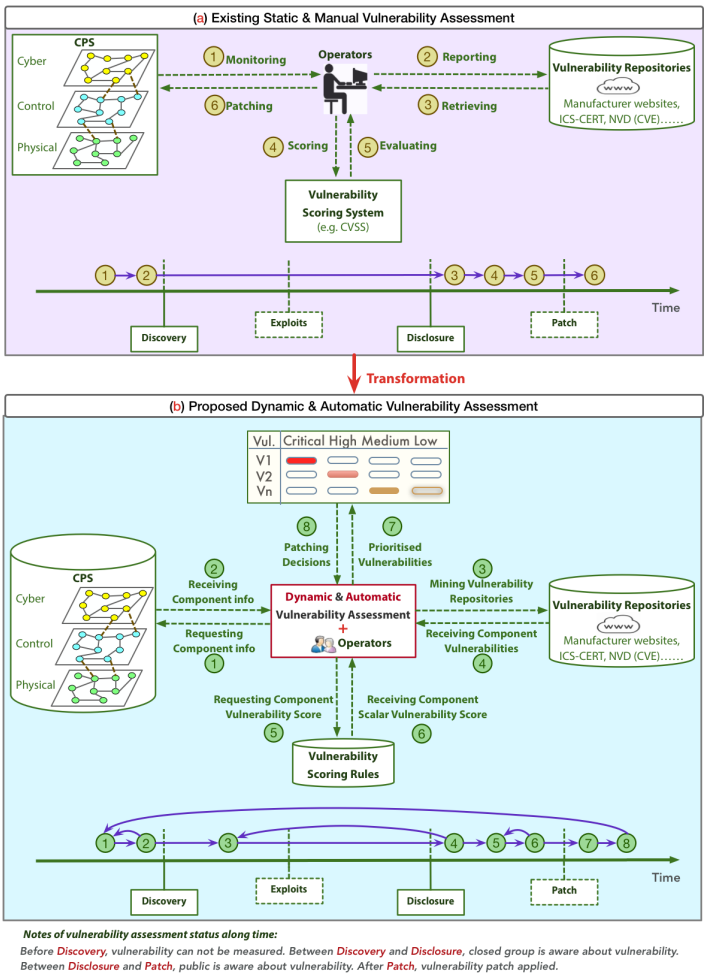

Fig. 1. Vulnerability Analysis Trends for Cyber-Physical System.

[2] https://cve.mitre.org/

However, CPS vulnerability analysis faces two challenging issues:

a) Subjective and Human-Centred Process: Existing vulnerability analysis approaches like CVSS calculator [3] require subjective and manual input, based on qualitative judgments of vulnerability properties such as exploitability, scope and impacts. Relying on individual experts' knowledge could introduce recurrent costs, subjective evaluations and contradicting outcomes. Nevertheless, security operators of CI also need to obey limited budget restrictions, and consider the limited computing resources of CPSs networks.

b) Static Vulnerability Analysis Lifecycle: Static analysis poses a limitation in security management of safety-critical systems such as CPSs, as malicious attempts may use new exploits that occur between successive analysis milestones [4]. On the contrary, dynamic vulnerability analysis allows mitigations to occur within the time interval that span the discovery and disclosure of vulnerabilities, and giving time for vulnerability patches to be available and deployed before the release time of exploits.

In our research, we propose to support a dynamic vulnerability-analysis approach, as shown in Part (b) of Fig. 1. The proposed streamlined approach employs cross-linked database management techniques to analyse data retrieved from multiple vulnerability-alert repositories with emphasis on CPS network components. This data analysis can be further combined with correlation techniques to produce both component-level and system-level vulnerability instances. Subsequently, retrieved vulnerability instances feed a rule-based scoring system using common industrial scoring standards such as CVSS, to derive a vulnerability-severity score automatically. We suggest to substitute offline, costly, error-prone and subjective vulnerability analysis processes with an automatic, accurate and data-evidenced approach, to improve situation awareness and to support security decision-making mechanisms. Based on this novel method, we did an in-depth case study of CPS vulnerability, to provide insights of the trends in CPS vulnerabilities in the context of CIs protection.

2 Related Works

Security assessment automation in CPS systems is evidenced by the plethora of definitions on Cyber-Threat Intelligence (CTI) in the literature, with an objective to overcome some notorious flaws in manual processes [9]. A recent trend in CPS vulnerability analysis uses computational intelligence approaches, such as graph-based and text-driven mining techniques.

Advances in formal semantics and data visualisation models have been widely used in graph-based applications for CPS security [5][2]. Tree structures, directed graphs and logic diagrams fall into this category. Tree-structured analysis and graph-based methods differ in the semantic of nodes and edges. But many of the previous studies mainly focus on certain vulnerabilities exploited by specific threat types, such as DoS [6]. However, risk analysis of a complex CPS may involve various vulnerabilities across highly-interdependent components of CPS. Current security-related data is mostly text-based, and appear in

three types, namely unstructured, semi-structured and structured data. However, graph-based approaches are not suitable to deal with text-based data.

On the other hand, other techniques have been studied and reported to support text-data driven information processing. Tools like cve-search[3] (referring to a Web interface and an API to CVE repository) and Open Vulnerability and Assessment Language (OVAL) are some of the efforts investigated to identify and manage CPS vulnerabilities effectively in a vendor-independent manner, while enabling a certain level of automation. Whereas, it is also desirable to define vulnerability indexes to help with a better categorisation and analysis of vulnerabilities, as well as to automate the analysis process by translating natural language statements found in vulnerability reports into machine-readable formats, as a supplement to automatically generate prevalent topics in vulnerability disclosure [12][10]. A more relevant work employs information-retrieval techniques to facilitate keyword identification, such as malware detection features from researcher papers [7]. However, these models focus on attack-steps and corresponding prerequisites in vulnerabilities instead of the vulnerable nature of the system.

Our method provides near real-time risk-monitoring through an information-fusion approach with multiple input vulnerability-repositories to provide up-to-date information, to extract relevant data for vulnerability analysis. This approach contrasts with traditional rigid management portfolios, which may tolerate security gaps and exploits occurrence beyond disclosure limits.

3 CPS Vulnerability

Typically, a CPS includes a cyber, a control and a physical process layer. The control layer features a network of operational technology (OT) components to coordinate and synchronise operations. The control centre provided by cyber layer incorporates an informational technology (IT) network of workstations and servers, including application and data-store servers. Therefore, each CPS asset confines data from software and physical components, while each component generates data from a list of vulnerability instances.

Considering the power grid as an example, it is a typical CPS empowered by a Supervisory Control and Data Acquisition (SCADA) control and monitoring system to efficiently optimise power generation, transmission and distribution processes [8]. The physical process consists in this case of a power-flow regulated by junction-busbars and power-generation sources. The control layer includes a network of microprocessor-controlled physical objects, such as remote terminal unit devices (RTUs) and programmable logic controllers (PLCs), which interface with physical process sensors. Master Terminal Unit (MTU) is another SCADA control-layer element that concentrates data gathered from RTUs. Thus, the control layer relays measurements from sensors that interact with field devices such as power-transmission lines and transformers, to remote control centres.

[3] https://www.circl.lu/services/cve-search/

Control centre applications process these measurements to support operational power-flow decisions to balance the supplied and demanded power flows.

Systems and softwares that are adopted in the digitalised power industry are interconnected. Meanwhile, more vulnerabilities have emerged due to the interconnections among systems. Considering the nature of CPS, we define a component to be either an application software (e.g. a PLC firmware program), a hardware (e.g. a PLC), a network (e.g. a network based on Modbus protocol), or an operating system (e.g. a Linux-based server). A software is embedded in a hardware, and is physically influenced by the hardware, for instance electricity supply. Different vulnerabilities might contribute to threats that bring different levels of impact, through different levels of losses in confidentiality, integrity and availability (CIA) triad. For example, an outdated software might contain flaws in source code. Such flaws might result in bypass threats materialised by code-injection attack. A hardware is vulnerable by having no physical-access protection, which might be used by an attacker to gain information (e.g. files store in the hardware) through unauthorised USB access, for example. A network protocol without encryption is vulnerable, which might result in threats like information (e.g. data-flows) leakage or man-in-the-middle (MITM). Attackers could trigger a privilege escalation or network reconnaissance attacks by making use of such protocol vulnerability. An operating system (OS) may expose to denial of service (DoS) threats due to resource management errors, which could be exploited by buffer overflow attacks. Most attacks happen in software or network environment. Whereas, a successful attack may also impact the hardware where exploited software is embedded.

4 Cross-Linked and Correlated Vulnerability-Database

We propose correlated database management techniques in vulnerability-data processing to discover CPS vulnerabilities and their attributes, to derive a multi-level vulnerability analysis from both component-perspective and asset-perspective, and to visualise the connection between vulnerability, threat and attack. Our research agenda mainly includes three steps, as further analysed below.

4.1 Step1: Vulnerability Database Preparation

We apply information fusion algorithms to extract attributes of vulnerabilities from multiple repositories into one local database, including vulnerability-instance repositories and security-related standard enumerations. According to the Coordinated Vulnerability Disclosure (CVD) guide [11], vulnerability report documents or vulnerability records could be found typically with some formal identifier, e.g. CVE ID. A published vulnerability report has zero or more associated vendor records, preliminary analysis of reported vulnerability severity using CVSS, and some other pertinent metadata. Using CVE ID as index, we build a database of vulnerability reports containing the base reports from CVE, as well as the cross-references from multiple repositories leading to corresponding manufacturer websites and standardised enumerations.

4.2 Step2: Cyber-Physical System Asset Database Preparation

CPS asset information are extracted from multiple repositories using information retrieval algorithms, including manufacturing websites and system configuration-related enumeration like CPE and Common Configuration Enumeration (CCE)[4]. CVE and NVD have been recognised as valuable resources for large-scale security analysis. Some other available security-related data could be gathered from online forums such as ExploitDB[5] and SecurityFocus[6]. To guide cybersecurity analysis process, a number of open standards are advocated to enumerate system configuration, weakness categorisation and attack categorisation. These standards include product dictionary Common Platform Enumeration (CPE)[7]; weakness taxonomy from Common Weakness Enumeration (CWE)[8]; attack patterns from Common Attack Pattern Enumeration and Classification (CAPEC)[9]. Each of these standards have its own syntax and semantics.

4.3 Step3: Correlation between Asset-Data and Vulnerability-Data

Knowledge-based reasoning approaches are applied to automatically abstract vulnerability attributes for concept-modelling and information-correlation. Features of different vulnerabilities are abstracted and updated with up-to-date vulnerability repositories, and then clustered into vulnerability instances which are stored in a Standardised Vulnerability Database. Meanwhile, component features including component properties, component versions, etc., are also abstracted to be stored in Asset Database. Information from the two databases are queried and correlated, to generate an Asset-based Vulnerability Database.

5 Evaluation and Discussion

In this section, we use our proposed vulnerability-search technique to gain insights about threat landscape in CPS environments via an experimental study conducted using relevant Python APIs.

5.1 Case Study Setup

Following the steps introduced in the previous section, we start by setting-up a vulnerability database that is inherently synchronised with multiple on-line vulnerability-reports repositories. Our database is built on top of cve-search Python API[10], which brings together CVE and enhancing NVD repositories

[4] https://cce.mitre.org/
[5] https://www.exploit-db.com/
[6] https://www.securityfocus.com/
[7] https://cpe.mitre.org/
[8] https://cwe.mitre.org/index.html
[9] https://capec.mitre.org/
[10] https://github.com/cve-search/cve-search

into a local MongoDB system that can handle large unstructured data. When the local MongoDB engine starts running, it is kept synchronised on hourly basis with feeds from repositories data, as shown in Fig. 2.

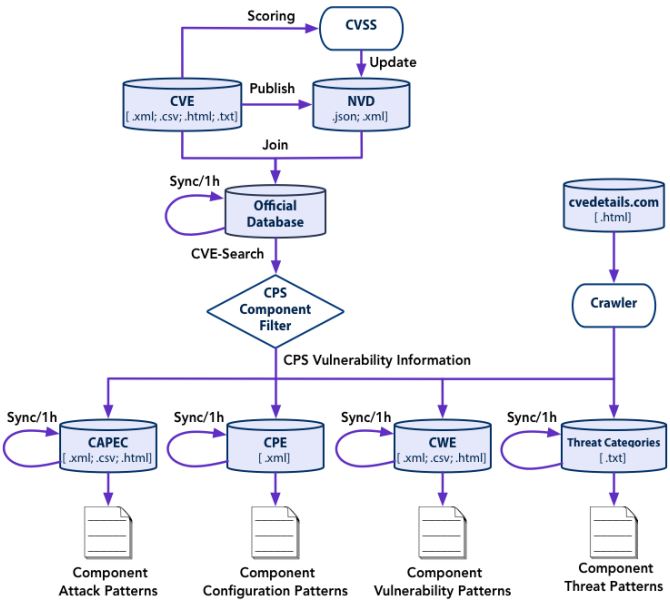

Fig. 2. Security Reports from Synchronised Cross-Linked Vulnerability-Databases

Subsequently, we retrieve CPS-relevant vulnerability instances, that we further cross-reference against a variety of sources including industry-standard CPE used to reveal operating systems, hardware and software information. Other cyber-security sources may also be enclosed in the data fusion process, namely CWE which expand the information-set about the vulnerability regardless the affected product instance, and CAPEC, which provides a dictionary of known attack patterns used by adversaries to exploit the discovered vulnerabilities.

The testing data set we used as a case study are retrieved till the 7th of July, 2019. The CVE database checked on 7th of July 2019 contained 123 687 entries, CPE contained 261 112 entries, CWE entries included 719 elements from CWE, and CAPEC included 463 elements.

We tested our correlated and cross-linked database method with four CPS asset types, namely RTU, MTU, PLC and HMI/SCADA. The four asset types are selected as they are core assets in Critical Infrastructures (CIs) CPS control systems. Next, we report the results obtained from querying our correlated local database to extract CPS assets vulnerability trend, as well as the analysis results about threat types that could exploit these CPS assets.

5.2 Case Study Results

We use our local vulnerability-databases that are kept synchronised on hourly-basis to generate security reports, which reveal CPS vulnerability trends across correlated feeds from third-party sources.

a) CPS Vulnerabilities: We first look at CPS assets' vulnerability instances. Total vulnerability instances amount of each CPS are mapped to the years from 2000 till 2019, considering that CVE discloses vulnerabilities since 1999 till now, as shown in Fig. 3. Compared to MTU and RTU, more vulnerabilities of PLC and HMI/SCADA have been disclosed, especially in the past 6 years.

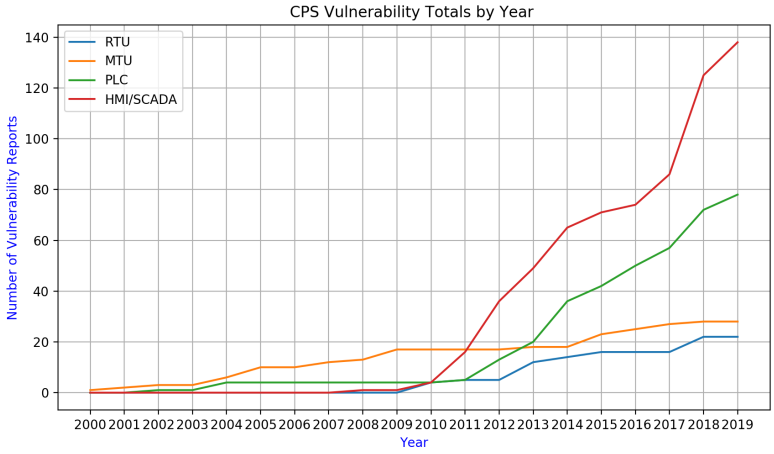

Fig. 3. CPS Vulnerability Totals Trend

Then we look at the severity level of CPS assets' vulnerabilities. Part (a) of Fig. 4 shows vulnerability-instance bar charts and average CVSS-score line, where the reported vulnerability instances related to RTU, MTU, PLC, HMI/SCADA are 22, 28, 78, and 138 instances respectively. Although HMI/SCADA reveal the largest amount of vulnerability reports, their average severity-score is not the highest. This is contrasted against the average CVSS base-scores of reported vulnerability instances in RTU, MTU, PLC, HMI/SCADA, which are 8.15, 6.26, 6.60, and 6.87 respectively. The average vulnerability score for overall CPS is 6.97 (approximately 7.0), which refers to "High" severity[11]. According to the documentation of CVSS v2.0, we mapped each CVSS base-score of vulnerability instances to the qualitative severity rating scale, and concluded that most of these vulnerability instances are evaluated as *Medium* or *High* severity, as illustrated in Part (b) of Fig. 4.

[11] https://www.first.org/cvss/v2/guide

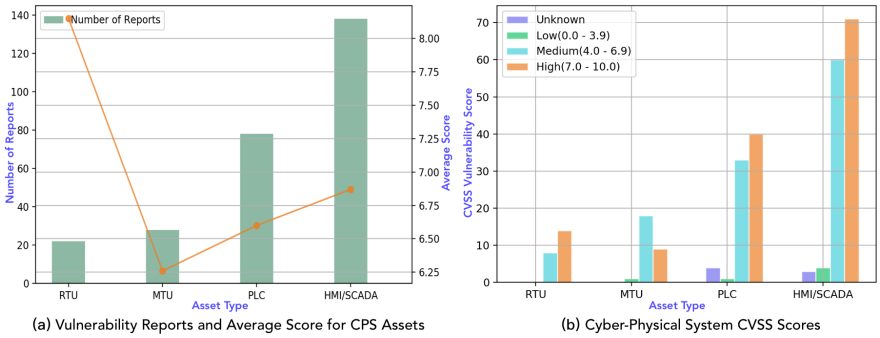

Fig. 4. Vulnerability Reports and CVSS Score for CPS Assets

b) Threat Types and Vulnerability Categories: We correlated CVE-ID against the threat categorisation provided in *www.cvedetails.com* to retrieve the threat types that a vulnerability instance may be exposed to. Note that a vulnerability could be exposed to more than one threat types. Three threat types, namely Execute Code, Denial of Service, and Overflow, appear to be the most typical ones that might materialise into attacks targeting CPS assets, as shown in Fig. 5.

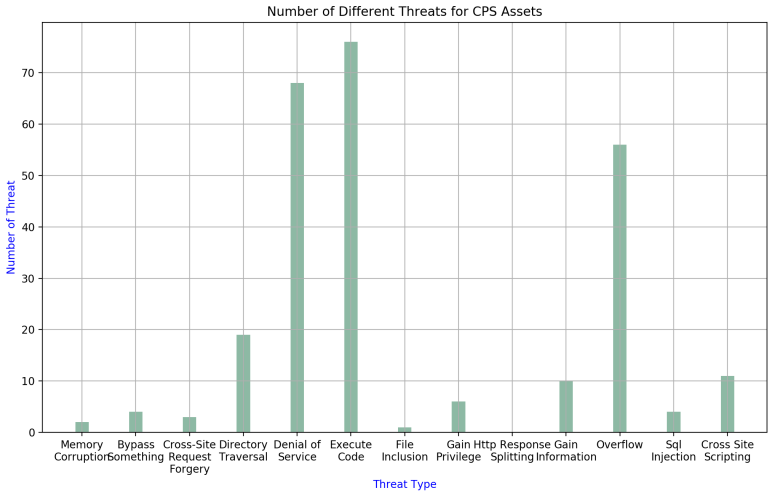

Fig. 5. Threat types Targeting CPS assets

Subsequently, we show the results of crosschecking CVE-ID against the weakness categorisation provided in CWE to obtain a list of weakness types. However, there are multiple ways of weakness categorisation in CWE, and different CWE-

IDs might refer to the same vulnerability type. Therefore, we gather all the related weakness descriptions, and extract common topics from gathered data. As a result, we get six topics to represent the most common weakness types of CPS, including Input Validation weakness, Authentication weakness, Access Control weakness, Resource Management weakness, Code Quality weakness, and Data Exposure weakness. Vulnerability categories of reported instances are shown in Fig. 6. It could be seen that the weakness type Access Control appears with highest frequency in CPS assets.

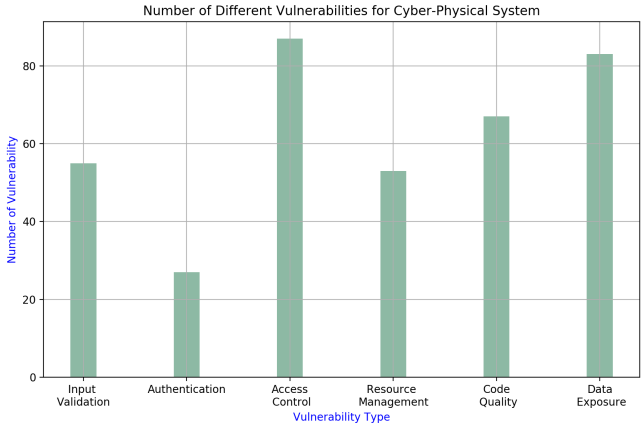

Fig. 6. Vulnerability Categories for CPS Assets

c) Vulnerable CPS Asset Components and Affected Vendors: From asset configurations, we retrieve their component information for each reported vulnerability. An example is the reported vulnerability *CVE-2013-2810*[12] in RTU-vulnerability, from which we obtain six CPE metadata records, such as "*cpe:2.3:o:emerson:dl-8000-remote-terminal-unit-firmware:2.30*" and "*cpe:2.3:h:emerson:dl-8000-remote-terminal-unit*". By default, a vulnerable software or operating system makes the embedding asset also vulnerable. Based on CPE naming specification, we further acquire detailed information of vulnerable components, such as component type (where *"h"* refers to hardware device, *"o"* refers to operating system, *"a"* refers to software application), vendor, component name, component version, etc. We calculate the amount of vulnerable components in two ways. One way is to sum-up only the component instances but ignore the different versions. The other way is to take into consideration the different versions and view them as different vulnerabilities. Adopting both ways leads to some interesting results, referring to asset-level vulnerabilities, as shown in Fig. 7. Although application software and operating systems are the main source of vulnerability, there might be larger number of hardware devices embedding those vulnerable components.

[12] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-2810

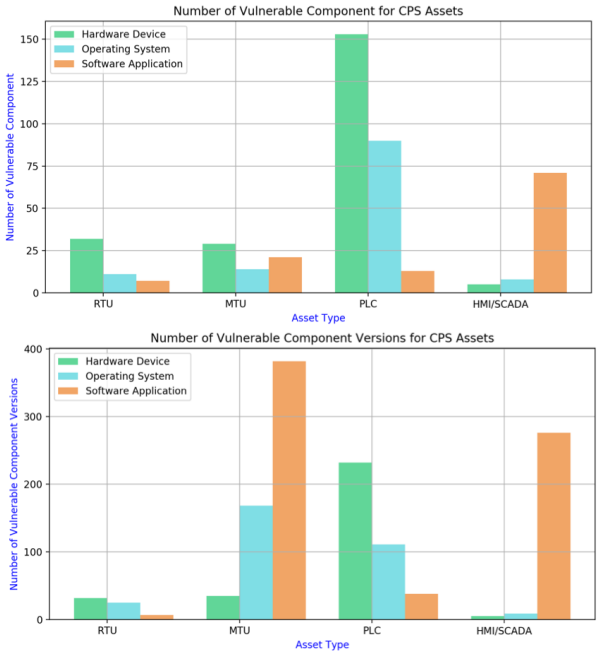

Fig. 7. Number of Vulnerability Components in CPS Assets

We further retrieve the vendor information for each reported vulnerability, and ranked the vendors based on the total amounts of affected products. The most distinct vendors are listed here. RTU vendors *Schneider Electric SE* and *Yokogawa Electric Corporation* have 14 and 8 affected products separately. MTU vendors *Cisco Systems Inc* and *F5 Networks Inc* have 29 and 17 affected products separately. PLC vendors *Schneider Electric SE* and *Siemens AG* have 138 and 32 affected products separately. HMI/SCADA vendors *Schneider Electric SE* and *General Electric Company* have 23 and 14 affected products separately.

6 Conclusion

In this paper, we motivated the need for, and the promises driven by correlated database management approaches to deal with semi-structured vulnerability data and also to address trends in vulnerability-analysis. We introduced a method to cross-check and correlate multiple vulnerability repositories through a step-by-step analysis. We built our local vulnerability database that is inherently synchronised with heterogenous security-related repositories and we employed information fusion techniques to extract relevant information. In doing so, we applied computational intelligence techniques to support classification of vulnerability categories and related threat types. We presented some results of the proposed approach as a case study that investigates vulnerability trends

at both CPS asset-level and component-level. The results we achieved could be obtained through manual processes, but it is more time-consuming and less accurate. By applying our methods, we managed to (a) offer a dynamic CPS-focused vulnerability analysis from several online repositories to assist operators evaluating up-to-date CPS vulnerability trends in order to reduce existing security management gaps, (b) narrow further the risk-window induced by discovered vulnerabilities, and (c) increase the level of automation in vulnerability analysis. In future works, we plan combine other computational intelligence techniques, to improve vulnerability evaluation at various levels of CPS architecture.

References

1. Sundaramurthy, S. C., Bardas, A. G., Case, J., Ou, X., Wesch, M., McHugh, J., & Rajagopalan, S. R. A human capital model for mitigating security analyst burnout. In Eleventh Symposium On Usable Privacy and Security (SOUPS 2015) (pp. 347-359). (2015).
2. Lallie, H. S., Debattista, K., & Bal, J. An empirical evaluation of the effectiveness of attack graphs and fault trees in cyber-attack perception. IEEE Transactions on Information Forensics and Security, 13(5), 1110-1122. (2018).
3. Joh, H., & Malaiya, Y. K. Defining and assessing quantitative security risk measures using vulnerability lifecycle and CVSS metrics. In The 2011 international conference on security and management (sam) (pp. 10-16). (2011).
4. Arbaugh, W. A., Fithen, W. L., & McHugh, J. (2000). Windows of vulnerability: A case study analysis. Computer, 33(12), 52-59.
5. Pasqualetti, F., Dorfler, F., & Bullo, F. Control-theoretic methods for cyberphysical security: Geometric principles for optimal cross-layer resilient control systems. IEEE Control Systems Magazine, 35(1), 110-127. (2015).
6. Mavroeidis, V., & Bromander, S. Cyber threat intelligence model: an evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence. In 2017 European Intelligence and Security Informatics Conference (EISIC) (pp. 91-98). IEEE. (2017).
7. Zhu, Z., & Dumitra, T. Featuresmith: Automatically engineering features for malware detection by mining the security literature. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (pp. 767-778). ACM. (2016, October).
8. Humayed, A., Lin, J., Li, F., & Luo, B. Cyber-physical systems security-A survey. IEEE Internet of Things Journal, 4(6), 1802-1831. (2017).
9. Kordy, B., Pietre-Cambacedes, L., & Schweitzer, P. DAG-based attack and defense modeling: Don't miss the forest for the attack trees. Computer science review, 13, 1-38. (2014).
10. Hafiz, M., & Fang, M. Game of detections: how are security vulnerabilities discovered in the wild?. Empirical Software Engineering, 21(5), 1920-1959. (2016).
11. Householder, A.D.,Wassermann, G.,Manion, A. and King, C. The CERT guide to coordinated vulnerability disclosure (No. CMU/SEI-2017- SR-022). Carnegie-Mellon Univ Pittsburgh Pa Pittsburgh United States. (2017).
12. Bogatinov, D. S., Bogdanoski, M., & Angelevski, S. AI-based cyber defense for more secure cyberspace. In Handbook of research on civil society and national security in the era of cyber warfare (pp. 220-237). IGI Global. (2016).