POSTER: VUDEC – A Framework for Vulnerability Management in Decentralized Communication Networks

Michael Steinke
Universität der Bundeswehr
Werner-Heisenberg-Weg 39
85577 Neubiberg, Germany
michael.steinke@unibw.de

Stefan Metzger
Leibniz Supercomputing Centre
Boltzmannstr. 1
85748 Garching n. Munich, Germany
Stefan.Metzger@lrz.de

Wolfgang Hommel
Universität der Bundeswehr
Werner-Heisenberg-Weg 39
85577 Neubiberg, Germany
wolfgang.hommel@unibw.de

ABSTRACT

Vulnerability management, often used as a generic term for any organizational and technical security controls in the context of identifying, assessing, and mitigating security-relevant software and network weaknesses, has specific challenges in decentralized communication networks such as research and education networks operated by higher education institutions. While many large organizations perform professional vulnerability management and related activities, especially risk management, which are supported by commercial and open source software products, universities and other academic environments still often struggle with ad-hoc and scope-limited approaches due to often unclear responsibilities and a lack of suitable tool support. This poster presents VUDEC, an integrated vulnerability management framework tailored for the requirements of decentrally operated networks; besides organizational aspects of the vulnerability management process, its implementation supports, among other functionality, a highly distributed vulnerability scan architecture and full multi-tenancy capability.

CCS Concepts

•Security and privacy → Malware and its mitigation; Vulnerability management;

Keywords

Decentralized Networks; Vulnerability Management; ISO/IEC 27001

1. MOTIVATION

As an essential part of information security teams' daily routine, vulnerability management (VM) consists of all activities related to the identification, assessment, and mitigation of exploitable weaknesses in systems and networks. Vulnerabilities stem from mistakes that are made in any phase of a system's lifecycle, such as neglected security requirements during design, implementation mistakes, insufficient diligence when configuring systems for production use, or carelessness during usage. As any other information security discipline, VM consists of *technical* as well as *organization* controls. Each implemented control, i. e., each security measure, can also be aligned with the lifecycle of potentially successful attacks and therefore be categorized as *preventing*, *detecting*, or *reacting* to successful attacks. Although many VM activities, such as scanning one's own network for outdated software installations, can intuitively be categorized as {*technical*, *preventive*}, it is obvious that the five other categories are just as basal for successful VM.

Our work focuses on the specifics of VM for communications networks that are operated in a decentralized manner, such as commonly found in research and education networks. Universities typically have a central data center or IT department, which, for example, provides the technical networking infrastructure and Internet uplink for the whole campus, but large parts of the operations and responsibilities are delegated to individual departments, chairs, and organizational units such as library and administration. As a consequence, VM must be performed in a much more distributed manner than in scenarios with a single central IT service provider, and careful orchestration is required to ensure that information flows, task allocation, monitoring, and controlling contribute to an effective solution, especially considering the typical heterogeneity of hardware and software used in academic environments. Given that related work did not yet analyze the specific demands and solution options for such environments in depth, the primary contribution of our work is an adaptable VM framework for decentralized networks. At its core, VUDEC is designed and the overall system architecture along with its centrally and decentrally operated components are specified.

In Section 2, we will outline contemporary practices regarding vulnerability management and present some ongoing research, followed by our framework in Section 3 and a conclusion as well as outlook to future work in Section 4.

2. PREVAILING APPROACHES

Publicly accessible VM concepts for decentralized networks (DN) are factually not available, resulting in research and education networks usually having to use a combination of various tools and having multiple individual approaches. Tools like vulnerability scanners can contribute to security

CCS '16, October 24–28, 2016, Vienna, Austria.

ACM ISBN 978-1-4503-4139-4/16/10.

DOI: http://dx.doi.org/10.1145/2976749.2989049

measures but rarely meet the requirements in DNs, which even holds for more comprehensive systems like OpenVAS. For example, nmap is a very flexible network scanner, but lacks management capability; in contrast, OpenVAS is a comprehensive software system for vulnerability detection and management but still cannot provide a user and asset management that fits a DN environment. On the one hand, available group and role management does affect the specific user's available functionality (e. g., read-only access and vulnerability scanning), whereas its capability regarding defining user permissions related to specific assets is plain and restricted to host and interface access control lists. Thus, a separation of users' responsibilities for different activities for a single asset is not possible but absolutely necessary in a DN. An example for this requirement is granting the data center the right to perform scheduled vulnerability scans to provide users with recurring automated checks (which heavily contributes to progressive vulnerability mitigation) without having access to personal data on devices. On the other hand, OpenVAS' rights management concept cannot take the DN structure itself into account, including self-organized as well as central institutions. It provides the definition of groups and the assignment of users to one or more of them, but they do not affect users' responsibilities. Nevertheless, tools like nmap can be used for vulnerability scanning and should be integrated into the technical VM system.

The remainder of this section focuses on contemporary, conceptual research work. For the sake of brevity, frequently summarized vulnerability description languages (CVE, OVAL) and weakness or resource description languages (CWE, CPE), vulnerability severity rating systems (CVSS v2 and v3), and frameworks for (mis-) configuration description (CCE, XCCDF) are intentionally excluded.

Regarding user-friendly VM, Elliot et al. describe their Software Application Vulnerability Management Dashboard System (SAVMDS) in [2], addressing software vulnerabilities. They provide functions for user management, reporting (e. g., graphs and statistics) and assessing vulnerabilities and brought-along risks but lack especially multi-tenancy capability, asset management, and the assignment of assets as well as users to different organizational domains. The approach also lacks orchestration and control capabilities regarding technical components like vulnerability scanners.

Regarding autonomic networks with the ability to manage themselves, deploy services on their own, and thus help with decreasing the complexity of network management, Barrère et al. [1] elaborated a survey regarding the analysis of current methods for vulnerability discovery, description, and detection. Although their work serves as a summary of good practices, it cannot be used for implementing the overall VM process, due to its lack of a description of a coherent system as well as user and asset management. Moreover, most research and education networks cannot fulfill the technical criteria to be considered as autonomic.

Cloud infrastructure and research networks do usually share the necessity to serve a huge amount of tenants, which might enable adopting similar VM approaches. In [3], the authors introduce "Cloudscope" – their approach of a VM system. Cloudscope can combine several existing vulnerability scanning tools and is able to perform asset discovery procedures, suitable for virtualized as well as physical hosts. Both aspects seem to be the major focus in this approach, whereas important management activities are missing.

3. THE VUDEC FRAMEWORK

VUDEC comprehends organizational aspects, covering users and roles, different organizational aspects, the management process itself with all its procedures as well as technical components and their structure within the network. The concept was developed based on experiences in a highly decentralized research and education network with more than 900 organizational units and can be adopted in other networks facing similar problems.

3.1 Roles and Organization

The role model in VUDEC is hierarchically structured, having defined each role's distinctive scope (cf. Figure 1). The roles are grouped into three different units, according to their responsibilities and tasks: The Network Level covers roles with institution- and component-independent responsibilities. The overall process' owner is the Chief Vulnerability Manager (CVM), conducting network-wide parameter settings (e. g., defining a network-wide vulnerability assessment system), monitoring and comparing the process' KPIs (e. g. vulnerability mitigation period) of each institute and the network as a whole. The Vulnerability Sources Manager keeps track of currently automatically queried vulnerability information databases (e. g., CVE or Bugtraq) and implements new query-mechanisms regarding further vulnerability sources.

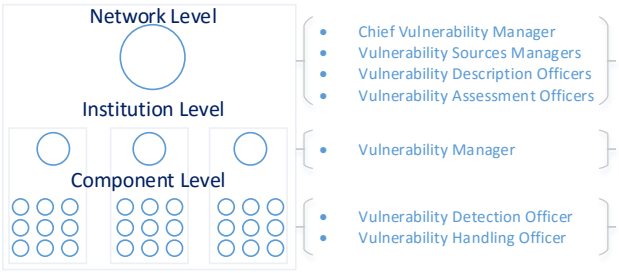

Figure 1: Hierarchy of user roles

Although Vulnerability {Description, Assessment} Officers are usually resident in different organizational units all over the network, their tasks are, however, institution-independent and cover the manual description and assessment of vulnerabilities, which cannot be processed automatically (e. g., due to a missing assessment score in external sources). Each institution's Vulnerability Manager basically has the same responsibilities as the singularly assigned CVM, yet limited to their own network part, i. e., he ensures that the vulnerability process in his own organizational unit is effective. The scope of roles within the component-level is limited to distinct computer systems, and, in this case, the detection as well as mitigation of vulnerabilities on them. One Vulnerability {Detection, Handling} Officer can have multiple assets assigned.

3.2 Technical Components and Structure

Enabling a network-wide approach especially depends on the technical platform of the VM process. A system reaching above all institutions and organizational units within the network is important due to the provision of a consistent course of actions and procedures like vulnerability documentation and assessment. Thus, the standardization of vulnerability descriptions allows system operators to make

use of information and knowledge (e. g., specific measures for vulnerability mitigation), which has been acquired by other network users earlier on. The architecture of our VM system is shown in Figure 2.

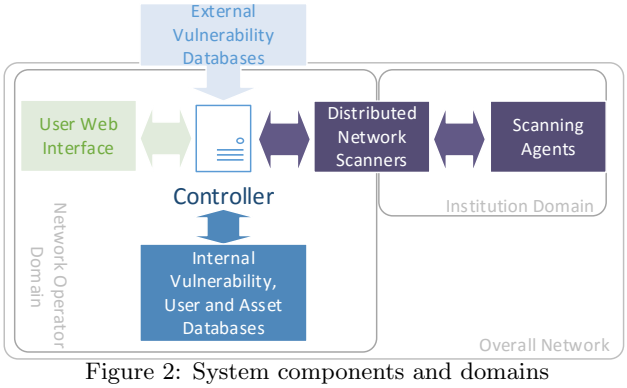

Figure 2: System components and domains

There are basically three different domains in our management system, indicating the organizational unit that is responsible for the management's technical platform component's operation: An implicit external domain, the network operator's domain, and the institution domain.

The network operator provides the central processing unit (Controller) with a web-based user interface (allowing users to scan their own assets and get further information regarding specific vulnerabilities and assets) for network-wide access as well as central Vulnerability, User and Asset Databases, ensuring a homogeneous documentation and description of vulnerabilities, user-assigned roles as well as assets. The Controller queries external, predefined vulnerability databases and updates, on the one hand, existing vulnerability entries in the system-wide databases and records new entries, triggered manually or by the Controller's scheduling unit. User information (e. g., name and institution) can be imported from existing, widely used systems like an Active Directory- or LDAP-service, roles must be assigned by a (Chief) Vulnerability Manager.

To guarantee vulnerabilities to be assessed in a homogeneous way, the Controller provides predefined aspects and questions to Vulnerability Assessment Officers through the web-interface and calculates a distinct criticality score within a given range (e. g., 0–10) based on a vulnerability's likeliness to be exploited and the exploitation's impact.

VUDEC benefits from different types of vulnerability scanners, which can be integrated and expanded dynamically due to our efforts on developing a well-defined and network-wide standardized framework for vulnerability detection. In our approach, performing a vulnerability scan is always triggered (manually or scheduled) by the controller, which *a)* prefilters relevant vulnerabilities (e. g., scanning for Microsoft Windows OS vulnerabilities on an asset running Linux is not reasonable), *b)* starting distributed network scanners, which call a scanning agent on the target, and *c)* merges the agents' results with the network scanners' results. Scanning mechanisms are always invoked the same way: By providing the target-asset's network ID (usually its IP address) and an arbitrary amount of parameters. Analogously the result of scanning mechanisms does is determined as well, on the one hand by returning *true* if a scanning mechanism positively detected a vulnerability, *false* otherwise. Additionally, scanning mechanisms can return arbitrary and non-boolean values, e. g., a set of open TCP ports on an asset. This vulnerability detection framework allows an implementation of further vulnerability testing mechanisms, e. g., through a simple script like shown in Listing 1 below, which makes use of nmap's capability to detect the Heartbleed Bug.

Listing 1: Sample detection script using nmap

```
1 #! /usr/bin/perl
2 my $netid= $ARGV[0]; #Get Asset's IP
3 my $out = qx/nmap --script /srv/scripts/
              ssl-heartbleed $netid/;
4 # Check nmap's output
5 if ($out =~ /VULNERABLE/) {print "true";}
6 else {print "false";}
```

Software agents can also be used to keep regarding records in the system's Asset Database up-to-date, for instance in terms of managing each asset's software inventory, which can be used in the vulnerability detection's preliminary filter-step.

Our system furthermore provides users with a vulnerability ticket system which shows users vulnerability-affected assets, the current handling status (open, closed) and existing and documented measures for vulnerability mitigation.

4. SUMMARY AND OUTLOOK

In this poster, we presented VUDEC, a comprehensive framework for VM in decentralized networks, considering an organizational structure and a technical platform, with the ability to scale-up to serve several thousands of users in a massively DN. Network users can benefit from a consistent approach, especially in terms of reusing acquired information like vulnerability mitigating measures and collectively documented vulnerabilities, and being guided through a reasonable sequence of management activities by a suitable user interface and management functionality like our ticket system.

Our future work will focus on interfaces to other information security management processes, especially risk management, and IT service management processes.Practical experiences gathered during the initial rollout and operation of the presented VM framework will be used for the continuous improvement of the system.

5. ACKNOWLEDGMENT

This work has been performed in the framework of the CELTIC EUREKA project SENDATE-PLANETS (Project ID C2015/3-1), and it is partly funded by the German BMBF (Project ID16KIS0549). The authors alone are responsible for the content of the paper.

6. REFERENCES

[1] M. Barrere, R. Badonnel, and O. Festor. Vulnerability assessment in autonomic networks and services: a survey. *Communications Surveys & Tutorials, IEEE*, 16(2):988–1004, 2014.

[2] M. Elliott, H. Yu, X. Yuan, and J. Zhan. Savmds: A software application vulnerability management dashboard system. In *Proceedings of the World Congress on Engineering*, volume 1, 2014.

[3] M. Kozlovszky. Cloud security monitoring and vulnerability management. In *Critical Infrastructure Protection Research*, pages 123–139. Springer, 2016.