MUNI
FI

Vulnerability Management

PA211 Advanced Topics of Cyber Security

September 27, 2022

Lukáš Sadlek, Jan Vykopal, and Pavel Čeleda

Exit tickets from last week – I

- **Q:** (Manual insertion of data to Netbox) ..., is there any alternative that would collect the information automatically?
- **A:** *Yes, utilities for Netbox [1] or complete asset management solutions.*
- **Q:** You mentioned netbox also provides a REST API. Is it used to automatically add/remove assets, e.g., devices of new employees?
- **A:** *Yes, e.g., Python client interacts with the REST API [2].*

Exit tickets from last week – II

- **Q:** Are we supposed to know Kibana on this level? I had hard time orientating in all the features.
- **A:** *At the end of the course, you should be able to solve tasks of similar difficulty. I will add troubleshooting commands, a video tutorial, and more details in solutions to the slides.*
- **Q:** What should guide us during question 7a?
- **A:** *Destination ports from IP flow dataset and hostnames from syslog dataset.*

Exit tickets from last week – III

- **Q:** Are netbox's clusters, device types just "groups/categories" we can create arbitrarily to ease our oversight of a big number of resources?
- **A:** *No, not arbitrarily. Difference will be visible with a lot of data. Clusters are logical groupings of physical hosts on which virtual machines reside, e.g., in virtual network emulating physical network. Device types represent a real type of hardware.*

Exit tickets from last week – IV

- **Q:** Are there any other programs as elk stack to query logs?
- **A:** *Yes, SIEM (Security Information and Event Management) tools, e.g., Splunk Enterprise Security.*
- **Q:** What is DMZ for?
- **A:** *This subnetwork contains services exposed to the Internet. The rest of the network is firewalled. The purpose is security.*

Goals of this lecture

- Become **acquainted** with:
  - Vulnerability management **lifecycle**
  - Vulnerability management **maturity model**
  - Standards, **enumerations**, and **data sources**
  - **Current possibilities** of vulnerability management

Essential terminology

Weakness

- **Flaws**, faults, bugs, and errors **in software and hardware**
- **Caused by** design, architecture, or implementation
- **Result in vulnerabilities** of systems, applications, and networks
- **Examples:**
  - Stack-based **Buffer Overflow** (CWE-121)
  - Use of **Single-factor Authentication** (CWE-308)
  - **SQL Injection** (CWE-89)

Vulnerability

- **Concrete instance** of weakness
- Appears in different **types of assets**
- We focus on vulnerabilities **in technologies**
- **Exploited** or triggered **by a threat** source
- **Examples**:
  - **Log4Shell** (CVE-2021-44228)
  - **Heartbleed** (CVE-2014-0160)

Vulnerability patch

- A “**fix**” for a piece of programming
- **Identified problem's** solution provided to **users**
- Sometimes published on the **manufacturer's website**
- Not **necessarily the best** solution compared to the product's **next release**

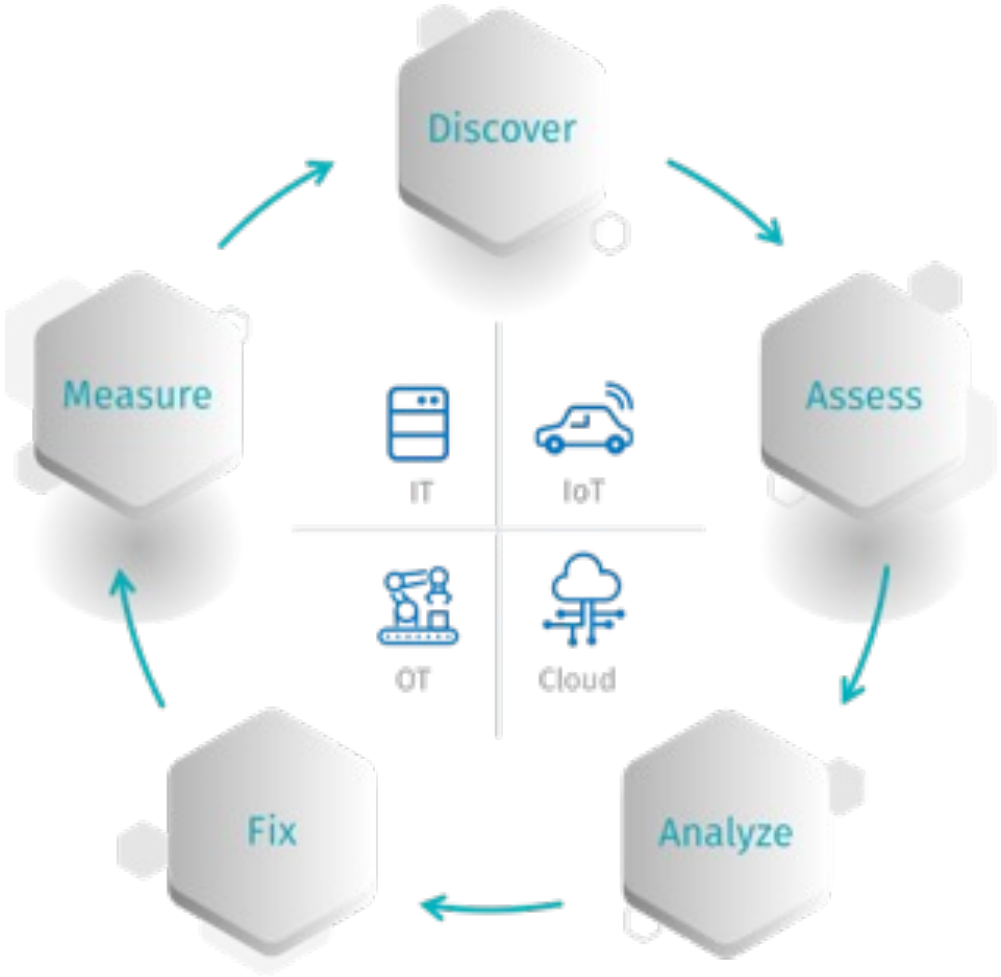
Vulnerability management

Vulnerability management

- A **process** of discovering, analyzing, and mitigating **vulnerabilities**
- Often includes **tracking of status** for each vulnerability
- **Dependent** on cybersecurity asset management
- Vulnerability scanning is only **one of** its **activities**

Vulnerability management lifecycle

- Different institutions provide **different stages**
- **Stages**
  1. Discover
  2. Prioritize/Asses
  3. Report
  4. Fix
  5. Verify
- **Example:** Tenable



MUNI FI

Vulnerability management maturity model

– Application of the **capability maturity model** on the **vulnerability management** (e.g., by SANS Institute)

Level	CMM	VMMM	Characteristics
0	—	Incomplete	Patching
1	Initial	Performed	Scanning policy
2	Repeatable	Managed	Scan & patch lifecycle
3	Defined	Defined	Prioritization, Full lifecycle
4	Managed	Quantitatively managed	Attack & threat centric
5	Optimizing	Optimizing	Business context

Vulnerability discovery

- Based on **asset discovery**
  - **Large-scale** vulnerability assessment
  - **Agent-based** – patching (Pakiti)
  - **Active monitoring** – vulnerability scanners (Nessus, OpenVAS)
  - **Passive monitoring** – less used
  - All approaches use **CPE** to determine vulnerabilities (**CVE**)
- **Direct exploits**
  - **Specific** vulnerabilities
  - **Metasploit** imports exploits from Exploit DB [1]

Tools for vulnerability management

- **Vulnerability scanners** – Nessus, OpenVAS, Nexpose
- **Solutions** for the whole **vulnerability management**
  - **Qualys** vulnerability management
  - **Rapid7** InsightVM
  - **Tenable.sc**
  - **F-Secure** Radar

OpenVAS

MUNI FI

Standards, enumerations, and data sources

Standards, enumerations and data sources

- Security Content Automation Protocol (**SCAP**)
- Common Vulnerabilities and Exposures (**CVE**)
- Common Weakness Enumeration (**CWE**)
- Common Vulnerability Scoring System (**CVSS**)
- National Vulnerability Database (**NVD**)

Security Content Automation Protocol (SCAP)

- A synthesis of **various standards** enabling **automated** management of vulnerabilities
- **Enumerations**
  - **CVE**, **CPE**, CCE
- **Scoring systems**
  - **CVSS**, CCSS
- **Languages**
  - OVAL

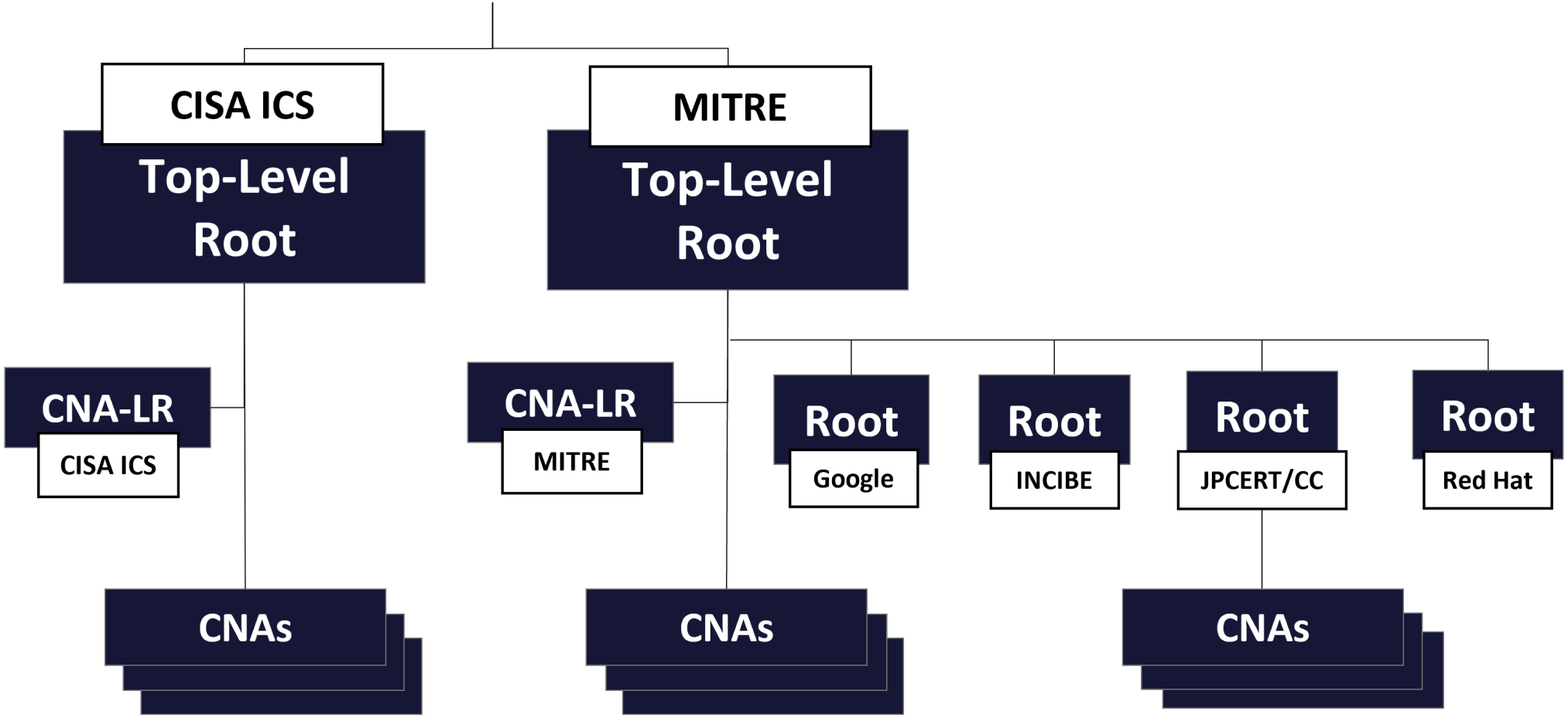
Common Vulnerabilities and Exposures (CVE)

- List of **publicly available** vulnerabilities
- Currently more than **185,000 CVEs**
- Contains for **each** vulnerability
  - **Identifier** – in form of **CVE-YYYY-NNNN**
  - **Description** – **without** standardized format
  - **References** – a website where vulnerability was **published**
- **Example:** Log4Shell CVE-2021-44228 [1]

CVE Numbering Authorities (CNAs) – I

- **Assign CVE IDs** withing their **scope** of responsibility
- CVEs are published on **vendor websites**
- **CVE Board** oversees hierarchy of **CNAs**
  - **Top-level** roots - Cybersecurity and Infrastructure Security Agency (**CISA**) Industrial Control Systems (**ICS**) and **MITRE**
  - CNA of **last resort –** CISA ICS, MITRE
  - **Root –** e.g., Google , RedHat
  - Other **CNAs**
- **Read more:** CNAs count [1], CNA list [2], CNA structure [3]

CVE Numbering Authorities (CNAs) – II



MUNI FI

Common Weakness Enumeration (CWE)

- An enumeration of **common weaknesses** (bugs, flaws)
- In **software** products or **hardware** devices
- **Hierarchical organization**
  - **CWE-74** - Improper neutralization of special elements (Injection)
  - **CWE-77** (Command Injection) is **child** of CWE-74
  - **CWE-81** (XML Injection) is **child** of CWE-74
- **Example:** CWE-94 (Code injection) [1]

MUNI FI

Common Vulnerability Scoring System (CVSS)

- A scoring system **evaluating** various **properties** of vulnerabilities
- **Two versions** are used in the NVD – v3.1 and v2
- **Impact metrics** – confidentiality, integrity, and availability
- **Exploitability metrics**
  - **v3.1** - attack vector, attack complexity, privileges required, user interaction
  - **v2** – access vector, access complexity, authentication
- **New in v3.1:** scope
- **Equation** for calculating score (v2 and v3.1)

National Vulnerability Database (NVD)

- **U.S. government repository** build upon **CVE**
- Assigns **additional information** to each CVE:
  - impact, severity and other **scoring information** in the form of **CVSS**
  - **product information** using **CPE configurations**
  - reference to the **weakness category** using **CWE**
- **CPE configurations**
  - **CPE match strings** connected by AND/OR relationships
  - Can contain **vulnerable and nonvulnerable** CPEs
- **Example:** Log4Shell (CVE-2021-44228) [1]

MUNI FI

Real-world observations (NVD)

- **CPE examples**:
  - cpe:2.3:o:linux:linux_kernel:*:*:*:*:*:*:*:*
  - cpe:2.3:o:debian:debian_linux:8.0:*:*:*:*:*:*:*
  - cpe:2.3:o:microsoft:windows_nt:4.0:sp1:*:*:embedded:*:x86:*
- CPE match string often contains **a lot of asterisks**
  - Parts **up to version** are usually assigned
  - Parts from update to the end of string have usually **ANY value**

MUNI FI

Current state

- **Observations** from related work
  - Even **20 years old** vulnerabilities can be **still** discovered
  - **No more than five years old** vulnerabilities are usually discovered
  - **Most** professionals do not know about **all organization's assets**
  - Unknown assets are left **completely unpatched**
- **CVSS criticism**
  - CVSS was **not intended** for prioritization
  - Many vulnerabilities with **highest CVSS severity** does not have **public exploit**
  - Prioritize also **less severe** but **exploitable** vulnerabilities

Supplementary materials

The Technical Specification for the Security Content Automation Protocol (SCAP): SCAP Version 1.3. Available from: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-126r3.pdf.

Mastering CVSS v3.1. Available from: https://learning.first.org/courses/course-v1:FIRST+CVSSv3.1+2020/about

MUNI

FI