MUNI
C4E

System for Continuous Collection of Contextual Information for Network Security Management and Incident Handling

Martin Husák, Martin Laštovička, Daniel Tovarňák

husakm@ics.muni.cz, lastovicka@ics.muni.cz, tovarnak@ics.muni.cz

Institute of Computer Science, Masaryk University, Brno, Czech Republic

August 17, 2021

Motivation & Goals of the Work

Motivation

- Incident handling (IH) and incident response (IR) are prone to human errors
- Incident handlers often lack important contextual data,
  their *cyber situation awareness* (CSA) is low
- Automation of IH/IR is difficult, automation of data collection is not

Goals of the Work

- Goal is to provide incident handlers with all the data they need for IH
- Data shall provide overview of the network and details on hosts and services in it
- Data collection shall be continuous so that the data are fresh and instantly ready

What Data Shall be Collected?

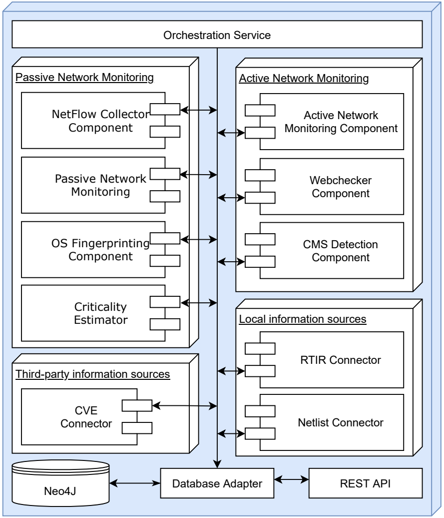
Network-wide information

- Network topology and segmentation (enumeration of subnets)
- Enumeration of hosts and responsible contacts (primary users or admins)
- Enumeration of critical hosts and their dependencies (critical infrastructure)
- History of incidents

Details on hosts in the network

- OS fingerprint – name and version
- Network services – software and version (for servers)
- Client software – web browser, antivirus (for workstations)
- Vulnerabilities on the hosts

System Design



Common components

- Orchestration service – Celery
- Database – Neo4j
- Database adapter & REST API

Data collection component

- Grouped by primary data
- Passive network monitoring – adapters to NetFlow monitoring infrastructure
- Active network monitoring – adapters to Nmap and other scanners
- Local and third-party sources – custom adapters to specific data and systems

Passive Network Monitoring

NetFlow collector component

- Connects to NetFlow monitoring infrastructure (collector)
- Queries NetFlow data, downloads records needed by other components

Passive network monitoring component

- OS fingerprinting – uses three methods to identify OS of communicating devices: TCP header, HTTP User-Agent, communication with specific domains (intensive ongoing research – developed separately)
- Service detection using NBAR2 signatures to identify services and software
- Web browser detection via HTTP User-Agent analysis
- Antivirus software detection via communication with specific domains

Active Network Monitoring

Active network monitoring component

- *Nmap*-based, scans 100 top ports for open services and network topology
- Complementary OS and software fingerprinting (CPE-formatted output)
- Time-consuming (16 hours in /16 network), clean-up and resume procedures

Webchecker

- Checks webservers if they provide content on port 80 or 443
- If port 443 is served, the certificate's validity is checked

CMS detection component

- Identification of CMS (*WordPress*, *Drupal*, ...) on previously discovered webservers
- Based on *WhatWeb* tool

Third-party and Local Information Source

CVE connector

- Downloads CVE records from NVD (primary) and vendors' databases (details)
- CVEs are matched with discovered software via CPE: $[CVE] - (CPE) - [Software]$

RTIR connector

- Downloads history of incidents from *Request Tracker for Incident Response*
- Incident details – timestamps, actors, status, ...

NetList connector

- Local list of network segments, IP ranges, and admin contacts:

  routers,10.0.0.0/24,networkadmin@organization

  servers,10.0.10.0/24,serveradmin@organization

Derived Information

Criticality estimator

- Varying definitions of critical infrastructures, manual enumeration is too laborious
- *Critical host* = *Critical node* in the network topology graph
- *Betweenness score* – how many shortest paths go through a node?
- Nodes with the highest betweenness score are considered critical
- The topic will be expanded in future work

CPE matching

- Matching CVE to software/services is enabled via CPE
- Matches are only partial, vulnerability assessment is not exact
- Vulnerabilities are assumed, not confirmed – still sufficient for CSA

Deployment scenario

Environment and measurements

- Masaryk University campus network (/16 IPv4 address range)
- 40,000 users, 29,000 devices (up to 15,000 active simultaneously)
- NetFlow probes located at several observation points
- Active probing from two locations (to increase coverage)

Hosting machines

- Master node – 8 core CPU, 32 GB RAM, database + orchestration service
- Worker node – 4 core CPU, 16 GB RAM, active probing + preprocessing
- NetFlow data were obtained from existing dedicated collector

Database Content and Vulnerability Assessment

Database Content

- Neo4j is a mature graph database, no performance issues
- Three months of historical data are enough for IH
- 2.5 GB, 697,783 nodes and 22,119,299 edges
- 29,335 unique IP addresses, 76,763 network services, 483,449 incidents

Vulnerability Assessment

- 18,749 hosts and 18,018 fingerprints via NetFlow, but only 85 % accuracy
- 5,771 hosts but only 1,620 fingerprints via Nmap
- Negligible (<100) number of observations by external tools (e.g., Shodan)
- 2,404 vulnerabilities mapped to 563 unique CPE identifiers

Conclusion and Future Work

Conclusion

- Design of a system for collection contextual information for incident handling
- The system uses existing data collection infrastructures, e.g., NetFlow
- The data facilitate CSA and provide valuable insights into the network
- Contextual information on handled hosts in the network are instantly accessible
- Using the system shall prevent human errors in incident handling

Future Work

- Procedural aspects of incident handling shall receive more attention
- Integrating the queries with IH tools (dashboards, request tracker plugins, ...)
- Establishing metrics to qualify and quantify impact of using the system

MUNI
C4E

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

C4E.CZ