Recommending Similar Devices in Close Proximity for Network Security Management

The 19th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2023)

Vladimír Bouček[1] **(492927@mail.muni.cz)**
Martin Husák[2] **(husakm@ics.muni.cz)**

[1] Faculty of Informatics, Masaryk University, Czech Republic

[2] Institute of Computer Science, Masaryk University, Czech Republic

June 23, 2023

Presenter's Biography

RDNr. Martin Husák, Ph.D.

- Researcher at Institute of Computer Science, Masaryk University, Czech Republic
- Member of Masaryk University's incident response team CSIRT-MU (`https://csirt.muni.cz/`)
- Currently a visiting researcher at The Cyber Center for Security and Analytics, The University of Texas at San Antonio, USA.
- This research was supported by OP JAK "MSCAfellow5_MUNI" (No. CZ.02.01.01/00/22_010/0003229).

Motivation

Ransomware and similar threat

- The rising complexity and variety of cyberattacks complicate **incident handling**.
- IDS and secure perimeter are bypassed by **social engineering attacks**, e.g., phishing.
- The malware further **spreads in the network**, exploiting surrounding computers.
- There is little chance of mitigating the spread of infection.

Incident handling

- Rapid incident response prevents spread of infection and reduces attack impact.
- Effective **triage and prioritization** of threats and incidents are of utmost importance.
- The behavior of malware can be **anticipated** to some extent.
- Social engineering is difficult to detect – we depend on **user reports**.

Approach: Recommender System

Recommender system for incident handling

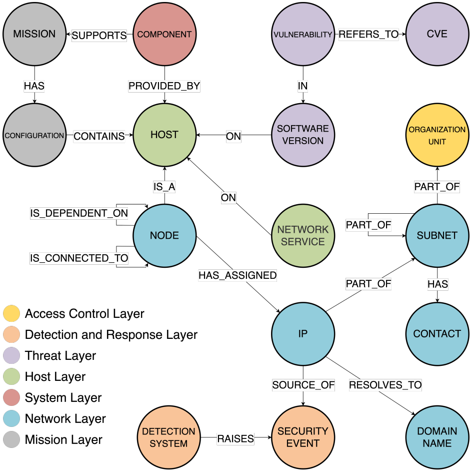
- Anticipating the attacker's behavior – how will the infection spread?
- The incident handlers would appreciate any piece of information that would guide them through the network and pinpoint nodes that are immediately threatened.
- The key question of an incident handler is:
  if this device is infected, which other devices can be infected or threatened?
- Requires timely data on the local environment.
- Theoretical background laid down in previous work[1].

[1] M. Husák. Towards a Data-Driven Recommender System for Handling Ransomware and Similar Incidents. In 2021 IEEE International Conference on Intelligence and Security Informatics (ISI).

Data for Making Recommendation

Data were collected by the CRUSOE toolset[a] in the network of Masaryk University and are stored in Neo4j graph database.

The dataset contains information on 31,135 IP addresses providing 97,829 network services in 844 subnets.



[a] Husák, M., Sadlek, L., Špaček, S., Laštovička, M., Javorník, M., & Komárková, J. (2022). CRUSOE: A toolset for cyber situational awareness and decision support in incident handling. Computers & Security, 115, 102609.

Data Collection

For each **host** in the network, the **CRUSOE** toolset collects the following:

- Fingerprint of the operating system (via NetFlow or Nmap),
- List of open ports and services, including the name and version of the underlying software (via NetFlow and NBAR2 signatures or Nmap),
- For web server: name and version of Content Management System (via WhatWeb),
- Name and version of a web browser used on the system (via NetFlow),
- Name of the antivirus software on the system and its latest update (via NetFlow),
- List of vulnerabilities (via vulnerability scanner or estimated from fingerprints),
- Location, purpose, contact on administrator or primary user.

Recommendation Procedure

The recommender system works in the following steps:

1. receives an identifier of a **host in the network** (e.g., IP address) on the input,
2. looks up the host in the database,
3. looks up devices in the **proximity** of the host,
4. calculates their **similarity** to the host on the input,
5. **prioritizes** the found hosts by their risk score,
6. returns a sorted list of **similar devices in close proximity** as the output.

Risk Score

- Formally, the hosts are sorted by their **risk score** (R) calculated as a quotient of the similarity (S) and distance (D) of the two hosts:

$$R = \frac{S}{D} = \frac{s_1 * s_2 * \dots s_n}{min\{d_1, d_2, \dots, d_n\}}$$

Proximity

Two hosts can be close to each other in physical and logical network topology, e.g., in the same room or in the same subnet. Alternatively, the two machines can be close to each other if they are controlled by the same users or administrators.

Similarity

Similar software equipment and vulnerabilities, role, profile, or shared history (past security incidents) of the two hosts.

Distance Calculation

When the ransomware is reported, we do not know yet how it spreads:

- Malware spreading over the network will typically spread in the same subnet.
- Malware infecting files and drives will spread to machines used by the same user.
- Malware in email attachments will spread in the same department.

Distance

The distance between the two hosts is the minimal value of various distance metrics.

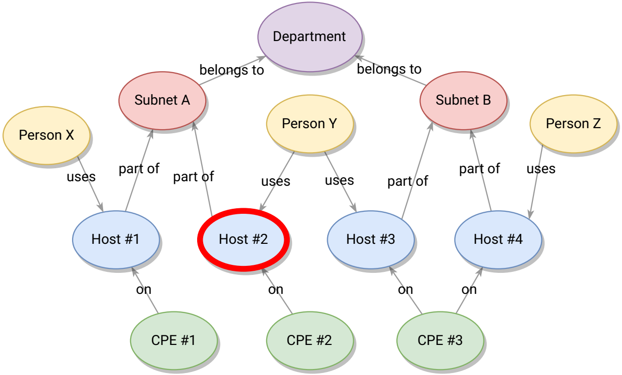
- Breadth-first graph traversal is used to find hosts with minimal distance in any of the distance metrics (in the implementation using graph database).
- The distance in logical network topology is the length of the path in the graph.
- Arbitrary distance metrics can be added as needed:
  physical distance, location in the same room, similarity of IP addresses, …

Distance calculation

Host #2 is reported to be infected, it’s distance to other hosts is:

- 2 to Host #1 (same subnet)
- 2 to Host #3 (same user)
- 4 to Host #4 (subnets belonging to the same department)

Host #4 is too far – the calculation of similarity between Host #2 and Hosts #1 and #3 follows.



Similarity Calculation

- The malware often uses exploits of specific software or services.
  - If malware uses SSH brute-forcing, then Linux machines with SSH servers are at risk.
- We do not know the exact software equipment and may only assume similarities.
  - If the malware exploits Outlook email client, we shall look up all Windows machines.

Similarity

The similarity is calculated as a product of partial similarities $s_1 * s_2 * ... s_n$.
Each partial similarity is a value in the range $< 0, 1 >$.

- The similarity of software equipment and network services are the main features.
- CPE strings represent pieces of software running on a host.

Similarity Calculation

Examples of similarity metrics

- CPE string similarity
  - CPE is an array of strings (vendor, product, version, ...) weighted 0.5, 0.25, 0.125, ...
  - The metric is the sum of weights of equal strings from the left to the first difference.
- CPE categories
  - If there is always 1 main CPE for each category, then simple CPE similarity is used.
  - Categories can be OS, browser, antivirus, ...
- Service similarity
  - If a service is provided by one host but not the other, a default value of 0.8 is used.
  - CPE strings are compared if both hosts provide the service.
- Similarities in vulnerabilities or past incidents
  - The number of common CVEs divided by the number of unique CVEs in the network.
  - The number of common past incidents divided by the total number of past incidents.

Similarity calculation

The OS fingerprint of Host #2 is compared to fingerprints of Hosts #1 and #3

- Host #1 and Host #2 share the same vendor, product, and version
  the similarity is 0.5 + 0.25 + 0.125 = 0.875
- Host #2 and Host #3 share only the vendor – their similarity is 0.5

CPE format	cpe:part:vendor:product:version:update:edition:language
Weights	0.5, 0.25, 0.125, 0.0625, 0.03125, 0.03125
CPE #1	cpe:2.3:o:microsoft:windows_7:-:sp2:*:*
CPE #2	cpe:2.3:o:microsoft:windows_7:-:sp1:*:*
CPE #3	cpe:2.3:o:microsoft:windows_10:-:*:*:*

The list of similar devices in close proximity for Host #2 goes as follows:

- Host #1, risk score is $0.875/2 = 0.4375$
- Host #3, risk score $0.5/2 = 0.25$

Configuration

```
{
    "max_distance": 2,
    "path": {
        "apply": true,
        "subnet": 1,
        "organization_unit": 1.25,
        "contact": 1.15
    },
    "comparators": {
        "os": {
            "apply": true,
            "critical_bound": 0.34927222,
            "diff_value": 0.2,
            "vendor": 0.9,
            "product": 0.075,
            "version": 0.025
        },
        "antivirus": {
            "apply": true,
            "critical_bound": 0.5,
            "diff_value": 0.4,
            "vendor": 0.6,
            "product": 0.25,
            "version": 0.15
        },
        "cms": {
            "apply": true,
            "require_open_ports": false,
            "critical_bound": 0.44568431,
            "diff_value": 0.4,
            "vendor": 0.6,
            "product": 0.25,
            "version": 0.15
        },
        "net_service": {
            "apply": true,
            "critical_bound": 0.25,
            "diff_value": 0.1
        },
        "cve_cumulative": {
            "apply": true,
            "critical_bound": 0.29492334
        },
        "event_cumulative": {
            "apply": true,
            "critical_bound": 0.00036752
        }
    }
}
```

Self-configuration

- **Weights** need to be assigned to all the features for best performance.
- Generic setting is unfeasible – depends on multiple factors, including local environment, prevalence of attack vectors in current malware, etc.
- Provisional automatic **self-configuration** is provided:
  - Mean bounds are calculated for all features in the current data.
  - For example, similarity of all pairs of OS fingerprints is calculated first.
  - Mean value is used as threshold/critical bound in similarity calculation.
  - Implemented as a part of the tool.
- Advanced settings is left for future work:
  - Data mining in the history of incidents.
  - Analysis of attack vector popularity.

Output

```
[
    ...
    {
        "ip": "147.*.*.*",
        "domains": [
            "*.cz."
        ],
        "contacts": [
            "*@*.cz"
        ],
        "os": {
            "vendor": "linux",
            "product": "linux_kernel",
            "version": "*"
        },
        "antivirus": null,
        "cms": null,
        "cve_count": 932,
        "security_event_count": 183,
        "network_services": [
            {
                "service": "NTP",
                "port": 123,
                "protocol": "UDP"
            }
        ],
        "risk": [
            5.67566254945783e-06
        ],
        "distance": 4,
        "path_types": [
            "Organization"
        ],
        "warnings": [
            {
                "message": "Similar OS between
                    hosts.",
                "partial_similarity": 1.0
            },
            {
                "message": "High number of common
                    net services between hosts",
                "partial_similarity": 1.0
            }
        ]
    },
    ...
]
```

Conclusion

Summary

- We implemented a **recommended system for incident handling**.
- Input is a reportedly compromised host in the network
- The system instantly provides a **prioritized list** of other **similar hosts** at risk.
- Text-based outputs or REST API for integration with other tools.
- Provisional self-configuration based on dataset analysis.

Future work

- Evaluation in cybersecurity operations (in collaboration with CSIRT-MU).
- The weights of the metrics will be inferred from past incidents using data mining.
- Integration with other incident handling tools (e.g., RTIR, TheHive).

MASARYK
UNIVERSITY