Week 7 Ass 2 Cyber Threat Intelligence

Task Overview

You will identify vulnerabilities of an information system and analyse Cyber Threat Intelligence (CTI) for a known Threat Actor (TA). This task is expected to take approximately 7 hours: 4 hours to search for vulnerabilities and 3 hours to investigate a TA and prioritise the vulnerabilities.

Groupwork

You may work alone or in groups of 2 or 3 people for this task.

All group members must be identified in the groupwork artefacts. Evidence must be provided that all group members contributed adequately to the final submissions. All group members must submit via the unit website. The moderation process might allocate group members different marks. Sharing of artefacts, for example, code or virtual machines, between groups is not permitted.

Repository

Create a private Git repository for yourself or for your group, for example, on GitHub. One code repository is to be used by all group members. Invite your tutor, the unit coordinator and your group members to the private repository.

Due

The CTI task is due in Week 7

Return

The CTI task will be marked with the Attack task. Feedback for the Attack task will be provided within 2 weeks of the due date.

Submission Overview

Submit artefacts to both your private code (Git) repository and to the unit website. Submit a link to your private repository to the unit website. All group members must submit.

Criteria Overview

You will be marked on aspects such as analysis of CTI and use of code repository and vulnerability search tools.

Scenario

You work at the Monto[1] Caravan and Cabin Park. Unfortunately, the company which you outsource your bookings to has been attacked by BlackCat ransomware. Your boss has asked you to improve the cybersecurity of our systems to prevent or mitigate further attacks.

For this task, the information system is comprised of the following nodes:

- DC VM

[1] 'It is believed that "monto" is probably a Gureng Gureng Aboriginal word meaning "plains with ridges on them" ' (https://www.aussietowns.com.au/)

- Odoo VM

You will:

- Install CQU_Office 2019 into DC
- Install additional app(s) into DC
- Setup a shared folder on Wazuh
- Search the information system for vulnerabilities without Wazuh
- Search the information system for vulnerabilities with Wazuh
- Prioritise the vulnerabilities

Install CQU_Office 2019 into DC
In DC, download and install the following file. You might already have installed this app during your tutorial.
https://github.com/jamieshield/coit11241/blob/main/office_2019.exe

Install additional app(s) into DC
Download an additional app from the "*Ass 2 Task CTI Install this app in DC*" activity on the unit website. These apps are fingerprinted to check for academic misconduct.

If you are working in a group, you will need to share your additional app with your group members. For a group of three, you will each need to collect three additional apps. You must install all of your group members' additional apps inside DC even if the apps appear to be the same.

Setup a shared folder on Wazuh
Setup a shared folder on Wazuh. If the mount point is left empty, a shared folder named *host* will appear in Wazuh as /media/sf_host. You can use the shared folder to copy files such as cpe_helper.json back to your host.

Search the Information System for Vulnerabilities without Wazuh
Search for vulnerabilities in the information system using tools such as Nessus, Nikto, Legion, Nmap and Wafw00f. Scan all nodes in your information system. Collect artefacts such as scan reports and screenshots as evidence of your searches.

Academic misconduct: For Nessus scans, you must include your name in the Nessus scan name, for example, DC_JamieShield. Generate a PDF report of the Detailed Vulnerabilities by Host. **Nessus reports that do not include your name might be tested for academic misconduct.** When working in a group, only one group member's name needs to be included in the scan name.

Search the Information System for Vulnerabilities with Wazuh
Setup Wazuh to perform vulnerability searches for packages installed in Windows nodes using Cpehelper.

1. Start and log into Wazuh.
2. Copy the cpehelper file into your shared folder so that you can edit the file on your host using a text editor such as Notepad. For example if your shared folder is named host:

```
wazuh$ sudo cp /var/ossec/queue/vulnerabilities/dictionaries/cpe_helper.json /media/sf_host
```

   Alternatively, you could edit the file in situ using nano or vi.

3. Ensure Office 2019 has been added to cpe_helper. You might already have completed this task in your tutorial.

a. Edit your cpehelper file: you can edit files in your shared folder using a GUI text editor such as Notepad on your host. Add an extra entry into the dictionary array [] using the following template.

```
{ "target": "windows",
  "source": {
    "vendor": [ "INVENTORY_VENDOR" ],
    "product": [ "INVENTORY_NAME" ]
  },
  "translation": {
    "vendor": [ "cpe_vendor" ],
    "product": [ " cpe_product" ]
  },
  "action": [ "replace_vendor", "replace_product" ]
},
```

b. Determine the installation product name and vendor of Office 2019. For example, go to Wazuh > Agents > **Agent** > Inventory data. Find the Office 2019 package in the inventory list.
c. Enter the installation product name and vendor of Office 2019 into cpe_helper.json. That is, in cpe_helper.json, replace INVENTORY_NAME and INVENTORY_VENDOR with the package Name and Vendor. These entries are regular expressions, for example, if the package name was followed by the version number, you could add .* after the package name.
d. Enter the CPE translation for Office 2019 in cpe_helper.json. For example, go to https://nvd.nist.gov/products/cpe/search and search for Office 2019. Find the most appropriate CPE URI that matches. In cpe_helper.json, replace cpe_vendor and cpe_product with the CPE Vendor and Product.

4. Add all other packages to CPE Helper. To save time, exclude Keyfinder, Notepad++, Visual C++, Python, Process Hacker, 7-Zip and update packages.
5. Save and copy the shared file back to Wazuh, for example:

```
wazuh$ sudo cp /media/sf_host/cpe_helper.json /var/ossec/queue/vulnerabilities/dictionaries/
```

   Alternatively, if you edited the file in situ, you should now copy cpe_helper.json to your shared folder as a backup using the cp command in step 2.

6. Check the cpe_helper.json file has no JSON syntax errors

```
wazuh$ sudo ~wazuh-user/checkCpeHelper.sh
```

7. Allow Wazuh to perform a vulnerability scan. A Wazuh vulnerability scan can be triggered by restarting Wazuh. There should be some additional vulnerabilities found by Wazuh. Export the complete list of vulnerabilities as a .CSV and submit the file to your Git repository.
8. Submit cpe_helper.json to your Git repository.

Prioritise the Vulnerabilities

Prioritise the vulnerabilities you have found based on their likelihood of exploitation by BlackCat. Start by creating a PowerPoint slide that provides an overview of BlackCat's attack lifecycle. Create

2-5 PowerPoint slides that identify the most important ATT&CK TTPs, CAPECs and CWEs for the Monto Caravan and Cabin Park information system. To help you understand the task, Appendix A provides an example summary of the lifecycle of a different malware (BumbleBee) including its important ATT&CK TTPs. Identify the top five vulnerabilities that you would fix first assuming that BlackCat is the largest threat for the Monto Caravan and Cabin Park. Include your rationale for your choices in a PowerPoint slide.

Task Submission

You should submit evidence of your Threat Actor investigation and vulnerability search and prioritisation including scan reports, Wazuh vulns (.csv), cpe_helper.json and PowerPoint slides including screenshots to your private Git repository and to the unit website. All group members must submit.

Task Criteria

Each of the following marking criteria have equal weighting.

Criteria	Indicative of 100%	75%	50%	25%	0%
Vulnerability search	Excellent use of tools ← Excellent search scope ←			→ Insufficient search scope, e.g. 1 tool used or some VMs not searched → Additional apps not installed	→ cpe_helper.json not provided in Git.
CTI	Excellent lifecycle summary ← Excellent prioritisation of threats & vulns including ATT&CK TTPs, CAPECs, CWEs & CVEs ← Excellent references, e.g. trustworthy ←			→ Poor references → Additional apps not installed	

Appendix A: BumbleBee's Lifecycle

This appendix provides a sample summary of BumbleBee's lifecycle.

BumbleBee's main aim is to load other malware such as ransomware. BumbleBee achieves this aim by (1) sending spearphishing emails with malicious files and motivating the user to (2) open the malicious files. Once initial access has been obtained, BumbleBee (3) injects malicious DLL code into processes to establish command and control and (4) download and (5) execute other malware.

1. BumbleBee's initial access techniques: T1566 Phishing - via links and attachments. ISO and ZIP files have been used (https://www.bleepingcomputer.com)
2. The user needs to act on the malicious attachment or link: T1204 User execution – of files and links
3. BumbleBee injects a malicious DLL into processes: T1055 Process injection – of DLLs (https://blog.cyble.com)
4. BumbleBee downloads other malicious payload(s) via a COM object and saves the payload as web.exe: T1559.001 COM (https://blog.cyble.com)
5. BumbleBee uses cmd.exe to execute payloads: T1059.003 Windows command shell.

MITRE ATT&CK lists over thirty techniques for BumbleBee. The five techniques provided above are the most crucial to enable BumbleBee to achieve its aim of loading other malware such as ransomware.