How do Hackers Hack? – Motivations, Techniques and Tools

1. INTRODUCTION

1.1. Background Information

In recent years, there has been a rapid increase in the number of cyber attacks globally, as seen from the number of data records being breached exceeding 22 billion in 2021 [1]. With such a tremendous number of cyber attacks, there needs to be a way to effectively categorise them in order to tackle and produce solutions against these attacks. Apart from this, the sophistication of the attackers, also known as Advanced Persistent Threat (APT) groups [2], is another area for concern. APT groups are more complex as compared to the typical "hacker". Their main goals may include espionage, financial gain, intellectual property theft and sabotage, according to [3]. To address the above, experts in the field have come up with the concept of Cyber Threat Intelligence (CTI) [4], which consists of 3 types: Tactical, Operational and Strategic [5]. By collecting CTI about APT groups, defenders are able to understand them better, thus allowing blue-team members to predict and prepare for possible intrusions. A popular way of categorising the behaviours of APT groups is through their Tactics, Techniques and Procedures (TTPs), which briefly, represent their tactical goals, methods to achieving those goals and specific steps to those methods respectively [6].

1.2. Evaluation of Current Solutions - The Gap in CTI

In the cybersecurity community, there are open-source frameworks such as MITRE ATT&CK [7] and MISP Galaxy [8], which allow organisations to share information about CTI. However, the structured data from these sources might not all conform to the same format, which can result in discrepancies between sources and increased difficulty in collating CTI. On top of that, CTI does not always come in structured formats – they might come in the form of text (unstructured data) in reports and articles. Unstructured data is difficult to store in large amounts and does not provide concise and specific information that is necessary for a swift response to an attack. On the contrary, structured data is portable, easy to access and actionable [9]. However, it requires time to convert all this unstructured information into structured data, which at the same time, is a major deciding factor to a prompt defence. It is predicted that the number of cyber attacks will increase [10], which can then lead to an increase in the amount of CTI data available. Consolidating CTI about adversaries will become more time consuming, especially when it comes to unstructured data sources.

1.3. Relevance - Closing the CTI Gap

Consider a scenario where an unlearned APT group is disrupting operations in the healthcare industry and many organisations in the region have already been attacked. The other healthcare organisations would ramp up their defences against this APT group in particular, and need to learn quickly from the previous attacks. However, waiting for the cybersecurity community to consolidate information about the attacks into structured data may not be fast enough to allow these organisations to take appropriate action. By addressing this issue, organisations can quickly gather useful data from a mass of information, such as reports of similar cyber attacks, to aid them in understanding their attackers and act sooner.

1.4. Aims and Objectives

The project aims to aid cybersecurity professionals in collating structured and unstructured CTI data into a database. Then, CTI data about adversaries can be queried and graphs can be

generated for a quick overview of the situation, thereby increasing the effectiveness of deterring and protecting against an APT intrusion.

2. MATERIALS AND METHODS

2.1. Materials and Overview of Methods

Python 3.8 and MongoDB Atlas were used. The Python libraries installed include: attackcti, stix2, taxii2-client, pymongo, beautifulsoup4, PyPDF2, nltk and spacy.

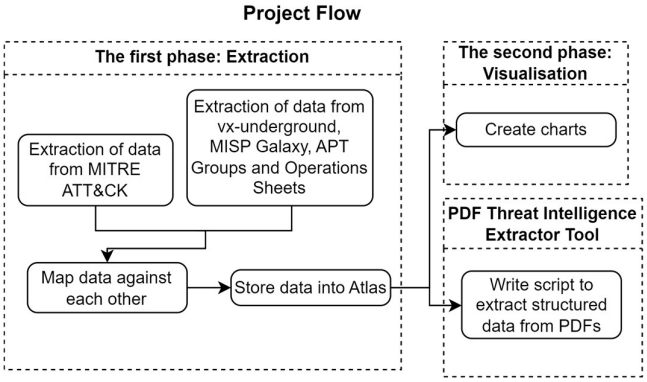

Fig. 1: Flowchart of project flow

To store all the CTI data, a MongoDB Atlas database (Community Edition) was first set up. It was chosen to allow easy replication of the project.

The Extraction phase consists of the following summarised steps: *(ref. Appendix A1)*

1. Extraction of data from MITRE ATT&CK using the attackcti Python module.
2. Extraction of data from vx-underground, MISP Galaxy and APT groups and Operations online Google Sheets.
3. Storing the data into MongoDB Atlas, an online version of MongoDB.

The Visualisation phase consists of the following summarised step: *(ref. Appendix A2)*

1. Creating charts using the "Charts" tool on MongoDB Atlas with the data.

Lastly, a PDF Cyber Threat Intelligence Extractor Tool was created to automate the process of extracting and collating reports and articles into structured formats.

2.2. Setting up the MongoDB database

MongoDB is a NoSQL database that uses nested dictionaries, which makes inserting and retrieving data much easier. It is flexible, portable and easy to expand. Its Charts Tool was also used to generate graphs from the data.

The steps taken to set up the database are as follows:

1. Create a cluster and a database
2. Create an admin user for the database
3. Establish a connection with the database through signing in with the admin user

2.3. Extraction Phase

2.3.1. *Extraction of data from MITRE ATT&CK*

The attackcti library is an API for retrieving information from the MITRE ATT&CK framework. With this library, data about APT groups, their TTPs, software, etc. was extracted.

Python libraries used: attackcti, pymongo, stix2, taxii2-client
The steps taken to extract data from the MITRE ATT&CK framework are as follows:

1. Connect to the ATT&CK client
2. Extract the necessary data and convert them into the dictionary data type
3. Upload the data from step 2 onto the MongoDB database using the connection set up

2.3.2. *Extraction of data from vx-underground*

This website was used to show the possibility of extracting data from unstructured sources, which can be repeated for other sources in the future.

Python libraries used: beautifulsoup4, PyPDF2, nltk, spacy
The steps taken to extract the PDFs and its contents from vx-underground are as follows:

1. Write a script to download all the PDF files from the website onto the computer
2. Write a script to extract the text from the PDFs using Natural Language Processing (NLP) and measure the proximity between words
3. Retrieve data about APT groups and the software they use from the database
4. Match the retrieved data against those from the PDFs
5. Upload the newly found data onto the MongoDB database

2.3.3. *Extraction of data from MISP Galaxy and APT groups and Operations*

Data from MISP Galaxy and the spreadsheets was extracted *(ref. Appendix A3)* and combined with the data from MITRE ATT&CK into the database. The collation of data from the two sources include APT groups and the software they used.

Python libraries used: json (built-in), csv (built-in)
The steps taken to extract the data from these 2 sources are as follows:

1. Write a script to extract data from the json files in MISP Galaxy
2. Download the spreadsheets from APT groups and Operations Google Sheets as csv files and write a script to extract the information from these files
3. Write a script to map the data from the sources, using MITRE ATT&CK as the basis. Mapping of data on malwares uses data from MISP Galaxy, fields used are shown in Fig. 2. The mapping of data on APT groups will use data from the spreadsheets and a file from MISP Galaxy, named 'threat-actor.json', fields used are shown in Fig. 3.
4. Upload the collated data onto the database

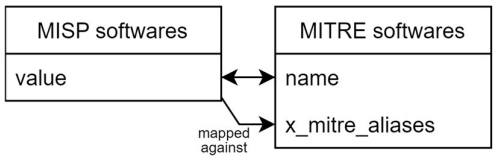

Fig. 2 Diagram of mapping of malwares

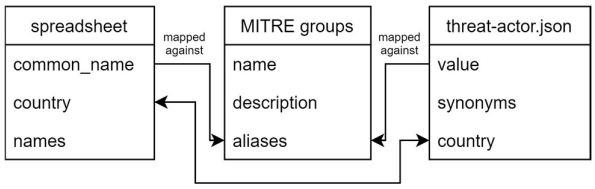

Fig. 3 Diagram of mapping of APT groups

2.4. Visualisation Phase

Through the MongoDB Charts Tool, users can create custom charts using the available data. After creating a chart, the data used for that chart can be exported elsewhere for more in-depth research. If needed, Charts also allows users to embed the graphs in websites.

2.5. PDF Cyber Threat Intelligence Extractor Tool

A script was written to analyse a directory of PDFs and extract structured data from them to increase the efficiency of obtaining information from reports. This will be particularly useful with the addition of more CTI sources besides vx-underground.

3. RESULTS AND ANALYSIS

3.1. Extraction Phase

3.1.1. *Results from MITRE ATT&CK*

Table 1: Summary of information extracted from MITRE ATT&CK

Type of information*	Data count
data components	224
data sources	40
mitigations	104
enterprise malware	507

Type of information	Data count
software (incl. enterprise malware)	718
techniques	761
groups (i.e. APT groups)	133

**ref. Appendix A3 for definitions*

3.1.2. *Results from vx-underground*

Table 2: Summary of information extracted from vx-underground

Type of information*	Data count	Estimated Accuracy
number of sources (PDFs)	1465	-
new software-group pairs	7	70%
hashes-malware pairs	1482	75%
hashes-group pairs	2924	70%
hashes-malware_family pairs	6779	80%
hashes (total)	35128	100%

**ref. Appendix A3 for definitions*

Implications: There is a significant amount of data that can be extracted from unstructured CTI sources, which reinforces the need for cybersecurity professionals to be actively gathering structured data from them. This in turn, reinforces the need and relevance of the tools that can help with these actions, e.g. the PDF Cyber Threat Intelligence Extractor Tool.

Limitations & Mitigations - software-group pairs: During the extraction of the software-groups pairs, NLP was used to extract noun phrases from the text to be matched against the data (APT groups and software) from the database. This method of data collection assumes that if a software and an APT Group are in the same PDF, they are associated with each other. But in some cases, there could be more than one APT Group or software, and it does not necessarily mean that both APT groups use both software. To address this possible source of inaccuracy in the data, checks were implemented whenever a PDF contained more than one APT Group. The check would require the user to manually read through the PDF to validate which APT Group used which software, if at all.

Limitations & Mitigations - pairs related to hashes: Hashes are easy to locate in a PDF file with the use of regular expressions, because they have a fixed format. However, in some of the PDFs, the hashes might just be listed in the appendix without any meta data or description. Therefore, relationships between hashes and other information could only be extracted from the body of the report. In this case, NLP would not be useful. Instead, the method of measuring the proximity between hashes and the surrounding text was used. It was assumed that the relationships between hashes and other information would be close to each other. From observations, it can be concluded that this is a largely reasonable assumption.

3.1.3. *Results from mapping of malwares*

Table 3: Summary of information from mapping of malwares

Type of information	Data count
Total number of malwares from MISP	899
Number of malwares not in MITRE	819

Implications: Useful information from MISP Galaxy that is not found in MITRE include the type of malware (e.g. backdoor malware) and its variants. This could help find possible relationships between the malware.

Limitations & Mitigations: There were only 53 overlapping malwares (out of 800+) before standardising the capitalisation of words. This suggests that different sources have different naming conventions for the same malware. After the malware names were uncapitalised and mapped against MITRE's malware aliases, there were 27 more overlaps, though the number is still relatively insignificant.

3.1.4. *Results from mapping of APT groups*

When mapping the countries where the APT groups are from, the 2 sources may provide different data. To increase readability, a point system is introduced, as shown in Table 4.

Table 4: Table of point system for country information of APT groups

Point	Conditions
1	groups that have matching country information from both sources
0.5	groups that have country information from one source
0	groups that have no country information or differing country information from both sources

Table 5: Summary of information from mapping of APT groups

Type of information	Data count
total number of groups from both sources*	587
number of groups not in MITRE	452

Type of information	Data count
groups with score of 1	76
groups with score of 0.5	236
groups with score of 0	269 (no info) + 4 (diff)

*contain duplicate groups from the different sources

Implications: These two sources provide useful information such as the country of origin, potential targets, target sectors, etc. By gaining a deeper understanding of APT groups, the techniques they may use can be better anticipated and in turn defended against.

Limitations & Mitigations: APT groups are named differently by different security agencies (e.g. 'Wicked Panda' from Crowdstrike can be referred to as 'APT41' from Mandiant), and some groups may be related or part of the same organisation. It becomes difficult to map groups based on their names. To address this, the group names were standardised using Mandiant's naming convention, as its format is the most straightforward. To map the data, the group name value from the other sources was first checked against the group aliases values from MITRE. If no matches were found, the group name value of the two other sources was then directly compared.

3.2. Visualisation Phase

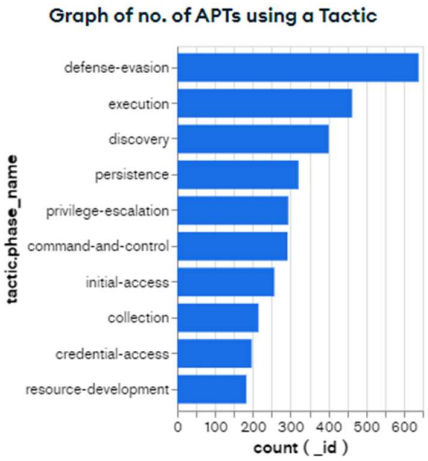

Fig. 4 Graph of no. of APTs using a Tactic

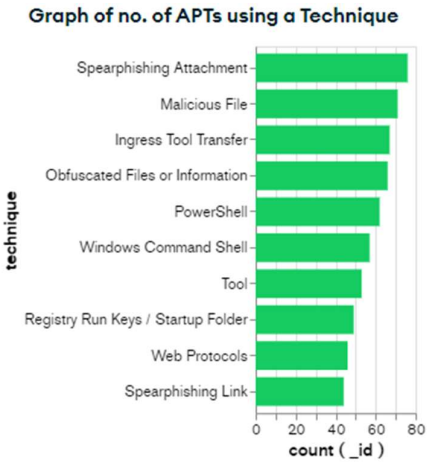

Fig. 5 Graph of no. of APTs using a Technique

From Fig. 4 and Fig. 5, relationships between TTPs and APT groups can be established. Questions cybersecurity experts would likely have regarding TTPs and APT groups can be answered by these graphs, assisting them in combating potential adversaries in the future.

In Fig. 4, the most common tactic of APT groups is defence evasion. It can be deduced that many APT groups attempt to intrude the system quietly to minimise the chances of being detected. This also implies that cybersecurity professionals would need to prioritise the enhancement of their detection systems to be more sensitive to suspicious activity, which would increase the possibilities of deterring a full fledged attack.

In Fig. 5, the most common technique used by APT groups is using a Spear Phishing Attachment, which is a social engineering technique that exploits human weaknesses, such as curiosity. Victims would be enticed to perform actions such as opening an attachment which actually downloads malware.

Implications: The insights from generating visual representations of CTI data go beyond the two examples provided. As MITRE introduces more methods for categorising the behaviours of APT groups, and the more data is amassed from other sources, a wider variety of graphs can be generated to produce deeper insights.

3.3. PDF Cyber Threat Intelligence Extractor Tool

Limitations: The NLP Python library used was not trained on identifying cybersecurity related information, but rather for general use. CTI tends to contain terms that are not found in the English vocabulary and using NLP would affect the accuracy of the data being extracted.

4. CONCLUSION

Now and in the future, the amount of CTI data will increase immensely. Providing a solution to properly categorise, store, share and extract more of such data is vital in maintaining the sanitation of knowledge in this area. In this project, CTI data was extracted and collated across multiple platforms into a centralised database, graphs were generated for visualisation of data, and a tool was created to automate the extraction of CTI from unstructured data sources. This project helps make sense of the heaps of information in a shorter amount of time through visuals and by summarising the data, which is crucial in deterring an impending attack. Future work could include improving the accuracy of data extraction through the use of a machine learning model trained in the context of cybersecurity and introducing a confidence indicator for the extracted results.

REFERENCES

[1] Security Magazine. (2022, February 10). *Over 22 billion records exposed in 2021.* https://www.securitymagazine.com/articles/97046-over-22-billion-records-exposed-in-2021?

[2] Crowdstrike. (2022, June 15). *What is an advanced persistent threat?.* https://www.crowdstrike.com/cybersecurity-101/advanced-persistent-threat-apt/

[3] L. Burita and D. T. Le, "Cyber Security and APT Groups," 2021 Communication and Information Technologies (KIT), 2021, pp. 1-7, doi: 10.1109/KIT52904.2021.9583744.

[4] Qiang, L., Zeming, Y., Baoxu, L., Zhengwei, J., Jian, Y. (2017). Framework of Cyber Attack Attribution Based on Threat Intelligence. In: Mitton, N., Chaouchi, H., Noel, T., Watteyne, T., Gabillon, A., Capolsini, P. (eds) Interoperability, Safety and Security in IoT. SaSeIoT InterIoT 2016 2016. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 190. Springer, Cham. https://doi.org/10.1007/978-3-319-52727-7_11

[5] Correa, A. (2021, June 16). *Three Types of Cyber Threat Intelligence*. Malware Patrol. https://www.malwarepatrol.net/three-types-of-cyber-threat-intelligence/#

[6] Dunham, K., & Lucas, C. (2017, January 19). *TTPs within cyber threat intelligence.* Optiv. https://www.optiv.com/explore-optiv-insights/blog/tactics-techniques-and-procedures-ttps-within-cyber-threat-intelligence

[7] The MITRE Corporation. (2022, October). ATT&CK, MITRE. https://attack.mitre.org

[8] Dulaunoy, A., Iklody, A., Dereszowski, A., Studer, C., Vandeplas, C., Andre, D., Servili, D., Wagener, G., Righetti, L., Vinot, R., Mokaddem, S., Rommelfangen, S., & Clement, S. (2019). MISP - Malware Information Sharing Platform and Threat Sharing - The Open Source Threat Intelligence Platform. Misp-Project.org. https://www.misp-project.org/

[9] zvelo. (2020, August 6). *Cyber Threat Intelligence (CTI): Planning and Direction.* https://zvelo.com/cti-cyber-threat-intelligence/

[10] Check Point Software Technologies, INC.. (2022, November 10). Check Point Software's Cybersecurity Predictions for 2023: Expect More Global Attacks, Government Regulation, and Consolidation. GlobeNewswire News Room. https://www.globenewswire.com/en/news-release/2022/11/10/2553575/0/en/Check-Point-Software-s-Cybersecurity-Predictions-for-2023-Expect-More-Global-Attacks-Government-Regulation-and-Consolidation.html

[11] Van Rossum, G. (1991). Python Programming Language. https://www.Python.org

[12] MongoDB. (2014). pymongo, MongoDB. https://www.mongodb.com

[13] Rodriguez, R., & Rodriguez, J. (2022). Attack Python Client. https://attackcti.com/intro.html

[14] OASIS Cyber Threat Intelligence Technical Committee. (2021). cti-Python-stix2. https://stix2.readthedocs.io/en/latest/

[15] OASIS Cyber Threat Intelligence Technical Committee. (2021). cti-Python-taxii2. https://taxii2client.readthedocs.io/en/latest/

[16] Hackett, B. (n.d.). pymongo: Python driver for MongoDB. PyPI. https://pypi.org/project/pymongo/

[17] Richardson, L. (n.d.). beautifulsoup4: Screen-scraping library. PyPI. https://pypi.org/project/beautifulsoup4/

[18] Fenniak, M. (2016, May 18). PyPDF2. PyPI. https://pypi.org/project/PyPDF2/

[19] Bird, S., Loper, E., & Klein, E. (2009). NLTK: Natural Language Toolkit. https://www.nltk.org/

[20] spaCy. (2015). spaCy · Industrial-strength Natural Language Processing in Python. https://spacy.io/

[21] vx-underground. (2021, November 26). vx-underground - malware samples. https://www.vx-underground.org/malware.html

[22] GitHub. (n.d.). misp-galaxy/clusters at main · MISP/misp-galaxy. Retrieved December 19, 2022, from https://github.com/MISP/misp-galaxy/tree/main/clusters

[23] Roth, F. (2022). APT Groups and Operations - Google Sheets. https://www.google.com/url?q=https://docs.google.com/spreadsheets/d/1H9_xaxQHpWaa4O_Son4Gx0YOIzlcBWMsdvePFX68EKU/edit%23gid%3D1864660085&sa=D&source=docs&ust=1671434005521825&usg=AOvVaw1GehMOFYF2dDAy9TIF1jwD

APPENDICES

Appendix A1: Extraction Phase (Materials and Methods)

Extraction of data from MITRE ATT&CK
Installation of libraries on Python

```
!pip install stix2
!pip install taxii2-client
!pip install attackcti
!pip install pymongo[srv]
```

Importing libraries on Python

```
from attackcti import attack_client
from pandas import *
import json

import pymongo
from pymongo import MongoClient

import logging
logging.getLogger('taxiiclient2').setLevel(logging.CRITICAL)
```

Getting Raw Data from MITRE

```
lift = attack_client()
techniques = lift.get_techniques()
groups = lift.get_groups()
groups_techniques = lift.get_techniques_used_by_all_groups()
```

Creating Database Connection and Constituent Collections

```
client =
pymongo.MongoClient("mongodb+srv://Admin:__password__@cluster0.kq
ydzgq.mongodb.net/?retryWrites=true&w=majority")
db = client.get_database("mitre")
db_techniques = db.techniques
db_groups = db.groups
db_groups_techniques = db.groups_techniques
```

Restructuring Data and Inserting them into Database

```
all_groups = []
for t in groups:
    all_groups.append(json.loads(t.serialize()))
db_groups.insert_many(all_groups)
```

Extraction of data from vx-underground
Downloading and Grouping PDFs

```
def download_and_group_pdfs():
    APT_GROUPS_LINK = "https://www.vx-underground.org/malware.html" #
Url to the website to download the PDFs from
    SLT_CARDS = "div.leftcolumn div.card" # CSS Selector for the links
on the main page
    SLT_APT_REFERENCES = "td.link a" # CSS Selector for the links of
the PDF files (download links)

    folder_name = ""
    def download_pdf(href):
        try:
            r = requests.get(href, allow_redirects=True)
            if not r.ok:
                return "Response not ok"

            save_path = os.path.join(folder_name,
os.path.basename(href))

            if os.path.exists(save_path): # Check if previously
downloaded already
                print("{} already exists, skipping
download.".format(save_path))
                return save_path

            Downloader.download(save_path=save_path,
response_content=r.content)

        except Exception as e:
            return e

    soup = bs4.BeautifulSoup(requests.get(APT_GROUPS_LINK).text,
"html.parser")
    cards = soup.select(SLT_CARDS)[1:] # The first card is ignored
because it contains malware samples, cards is a soup object
```

```
    # print(cards)
    for card in cards:
        folder_name = card.h2.text.replace(':', '') # Replace the semi
colon to allow similar naming conventions on Windows File Explorer
        print("Downloading pdfs for folder: ", folder_name)
        if not os.path.exists(folder_name):
            os.makedirs(folder_name)

        apt_group_links = [a["href"] for a in card.find_all("a")] #
Find all the links in the card soup object
        print(apt_group_links)
        for apt_group_link in apt_group_links:
            refs_soup = bs4.BeautifulSoup(requests.get(apt_group_link +
"Paper/").text, "html.parser") # Get the soup object of the page which
contains link of the PDF
            pdf_links = [a['href'] for a in
refs_soup.select(SLT_APT_REFERENCES, href=True) if a.get('href')][1:] #
Get all the links of the papers found in the page
            for pdf_link in pdf_links:
                download_pdf(pdf_link)
```

This set of code is specifically meant for downloading and grouping PDFs from vx-underground. If you wish to scrap PDFs from other sources, you can adapt this code according to the structure of the website, e.g. according to whether it has sub-directories, different HTML structure, naming convention of different folders/urls etc. Once you run this script, the PDFs will be downloaded into the same directory as the script.

Getting text from a PDF

```
def get_text_from_pdf(self, pdfFileObj):
        pdfReader = PyPDF2.PdfReader(pdfFileObj) # Get PDF file object
        total_pages = pdfReader.numPages

        text = ''
        for page_num in range(total_pages):
            pageObj  = pdfReader.getPage(page_num)
            text += pageObj.extractText() #Extracting the text from
each page
        if text == '':
            text = textract.process(pdfFileObj, method='tesseract',
encoding='utf-8', language='eng')
        return text
```

Getting noun phrases from text using NLP

```
def get_noun_phrases(self, text):
        nlp = spacy.load("en_core_web_sm") # Loading the English
Language trained Machine Learning Model
        doc = nlp(text) # Getting a document object from the text
        fdist = FreqDist([np.text for np in doc.noun_chunks if
np.text.lower() not in stopwords.words('english') and np.text.lower()
!= np.text])
        return fdist
```

Getting the hashes from the PDF using Regular Expression (RegEx)

```
def get_hashes(self, text, folder_name, file_name): # Getting all the
hashes from a text which is in string
        RE_HASHES = r"[a-fA-F0-9]{64}|[a-fA-F0-9]{32}" # Regex to find
the hashes
        hashes = re.findall(RE_HASHES, text) # Using re library to
return a list of all the occurrences of hashes in string
        self._hash_dictionary[folder_name][file_name] = hashes # Adding
the hashes into the dictionary
```

Getting hashes pairs using proximity

```
def get_hashes_malware(self, text): # Getting pairs of hashes and
malware pairs
        word_list = [word.strip().replace("\xa0", "") for word in
text.split(" ") if word.strip() != ""] # Splitting plain the text into
individual words
        hashes = re.findall(r"[a-fA-F0-9]{64}|[a-fA-F0-9]{32}", text) #
Getting a list of all the hashes present in the plain text
        for hash in hashes: # For every hash that is found using REgex,
try to find a software pair for it in the word_list
            for i in range(len(word_list)): # Finding the position of
the hashes in the word_list
                if hash in word_list[i]:
                    hash_position = i
            flag = False
            counter = 1 # The distance in words between the position of
the hash and the pointers
            while flag == False and counter < 6: # Keep going outwards
from the position of the hash until a malware with an extension is
found
                left_pointer = hash_position - counter
                right_pointer = hash_position + counter
```

```
                for extension in self._malware_extensions: # Checking
through all the file extensions for every position the pointers point
at
                    if word_list[left_pointer].endswith(extension):
                        self._hashes_malware[word_list[left_pointer]] =
hash # If there is an occurrence of a hash-malware pair, add to the
dictionary
                        print("====================\nFound hash
malware pair\n====================")
                        flag = True # Once there is pair identified,
skip to the next hash
                    elif word_list[right_pointer].endswith(extension):
                        self._hashes_malware[word_list[right_pointer]]
= hash # If there is an occurrence of a hash-malware pair, add to the
dictionary
                        print("====================\nFound hash
malware pair\n====================")
                        flag = True # Once there is pair identified,
skip to the next hash
                counter += 1
```

Using this code snippet, we can find the hashes malware pairs using proximity search. This was similarly implemented for hashes malware family pairs, hashes software pairs and hashes groups pairs.

Extraction of data from MISP Galaxy

```
json_files = ['android.json', 'backdoor.json', 'banker.json',
'botnet.json', 'cryptominers.json', 'exploit-kit.json', 'rat.json',
'stealer.json']
all_data_list = []
misp_names = []

for jsonfile in json_files:
  with open(jsonfile, 'r') as json_file:
    json_load = json.load(json_file)

  for malware in json_load['values']:
    misp_names.append(malware['value'])

  all_data_list.append(json_load)

# all_data_list[2]
print(len(misp_names))
```

This code snippet extracts data from the json files using Python's built-in library.

Extraction of data from Sheets

```
import csv

files = ['APT groups and Operations.xlsx - China.csv', 'APT groups and
Operations.xlsx - Russia.csv', 'APT groups and Operations.xlsx - North
Korea.csv', 'APT groups and Operations.xlsx - Iran.csv', 'APT groups
and Operations.xlsx - Israel.csv', 'APT groups and Operations.xlsx -
NATO.csv', 'APT groups and Operations.xlsx - Middle East.csv', 'APT
groups and Operations.xlsx - Others.csv', 'APT groups and
Operations.xlsx - Unknown.csv']
excel = []

# getting data from the spreadsheet (downloaded as csv file)
for filename in files:
  with open(filename, 'r') as csvfile:
    csvreader = csv.reader(csvfile)
    rows = []
    for row in csvreader:
      rows.append(row)

  country = rows[0][0]

    # other names
  for value in rows[1]:
    if 'Operation' in value:
      name_idx = rows[1].index(value)
      break
  other_names_source = rows[1][:name_idx] #include common name

    # operations
  for value in rows[1]:
    if 'Toolset' in value or 'Origin' in value:
      op_idx = rows[1].index(value)
      break

    # other fields
  headers = rows[1][op_idx:]
  headers = [header for header in headers if header != '']

  # adding the values
  for row in rows[2:]:
```

```
    common_name = row[0]
    dict_ = {'country':country, 'common_name':common_name}
    dict_placeholder = {'operations':{}}

    # other names
    name_values = row[:name_idx]
    name_values = [value for value in name_values if value != '']
    dict_.update({'names': name_values})

    # operations
    op_values = row[name_idx:op_idx]
    op_values = [value for value in op_values if value != '']
    dict_.update({'operations': op_values})

    # other fields
    links = []
    for header in headers:
      idx = rows[1].index(header)
      value = row[idx]
      if 'Link' in header:
        if value != '':
          links.append(value)
        else:
          pass
      else:
        dict_.update({header.lower():value})
    dict_.update({'links':links})
    excel.append(dict_)
```

This code snippet extracts data from the csv files obtained from APT groups and Operations Google Sheets and converts it into a list of dictionaries for convenience of querying.

Appendix A2: Visualisation Phase (Materials and Methods)

Navigating to MongoDB Charts

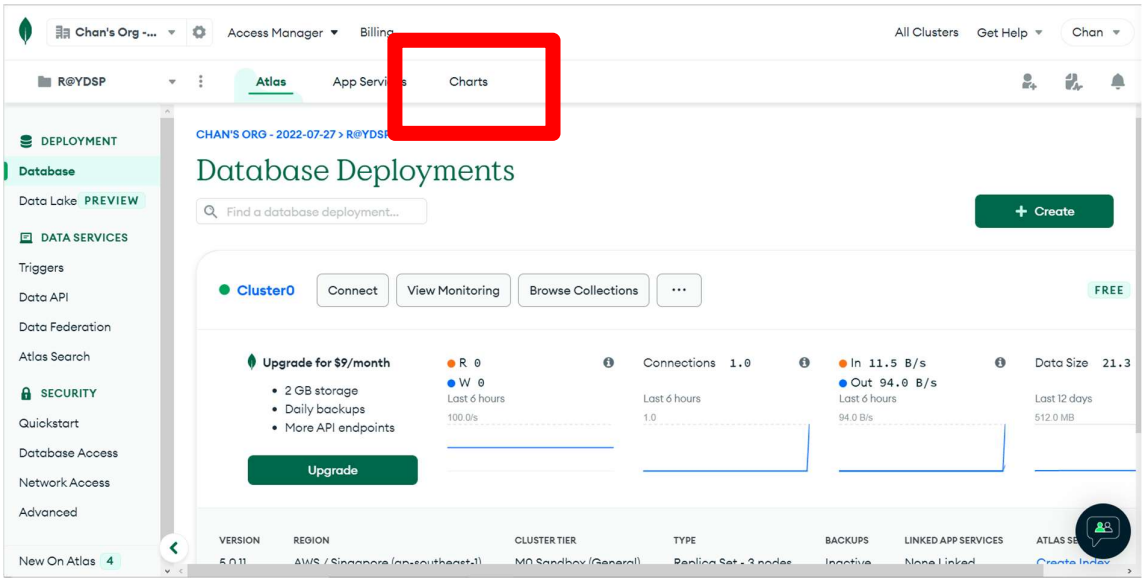

Fig. A2.1 MongoDB Atlas Home Page

To navigate to the charts tool on MongoDB Atlas, press the "Charts" tab near the top of the screen (boxed in red). Afterwards, if a dashboard has not been created, press on the green button at the top right corner "Add Dashboard" to create a new one. Next, click on a preferred dashboard and click the green button at the top right corner "Add Chart" to create a new chart.

Creating a Chart using "Charts" tool

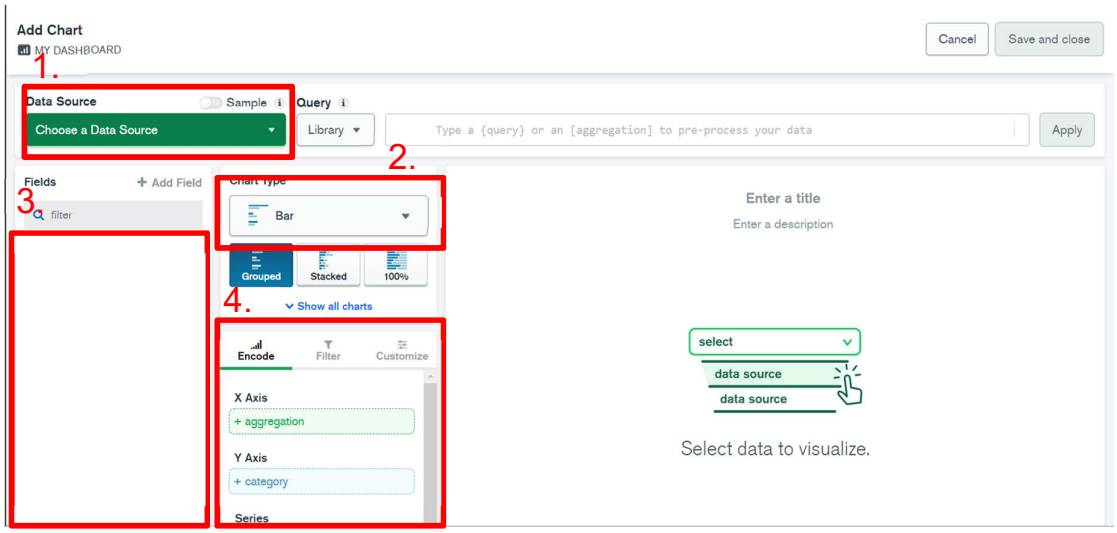

Fig. A2.2 Creating a Chart using MongoDB Charts

1. Click on the green button "Choose a Data Source". Then, choose a data source to generate a graph from.

2. Choose the type of chart you want to generate.
3. In this area, there will be a list of fields that your data contain.
4. This is where you drag and drop the fields from 3. to generate your graph. Depending on what your type of graph is, the X and Y axis may change according to either category or aggregation. The category axis is where you should put the field you want to sort the data according to. The aggregation axis would be the field you want to measure. After a short while, a graph will be generated on the right side of the screen, according to your specifications in steps 1-4.

Appendix A3: MISP Galaxy and APT groups and Operations (Materials and Methods)

Definitions

The MISP project is a well-known project in the cybersecurity community which mainly collates information of malware families.

The APT groups and Operations Google Sheets collates data of APT groups such as the groups' aliases and country of origin.

Appendix A4: Extraction Phase (Results and Analysis)

Definitions

Data sources refer to the source of CTI information gathered from detection systems about APT groups' behaviours during an intrusion.

Data components are subsets of data sources and are the groups of relationships between the data sources and data elements, which give an additional layer of context.

Mitigations are recommended strategies to counter a specific sub-technique.

The software-group pairs are in the form of APT Group uses software.

The hashes malware pairs are in the form of a malware file that produces hash.

The hashes group pairs are in the form of a hash associated with APT Group.

The hashes malware family pairs are in the form of hash belongs to a certain malware family.