Цифровые технологии и проблемы информационной безопасности / под ред. Е.В. Стельмашонок, И.Н. Васильевой. – СПб. : Изд-во СПбГЭУ, 2021. – 163 с.

Федоров Д. Ю., старший преподаватель СПбГЭУ

ИСПОЛЬЗОВАНИЕ ИНТЕРАКТИВНОЙ СРЕДЫ JUPYTER LAB ДЛЯ ФОРМИРОВАНИЯ АНАЛИТИЧЕСКИХ НАВЫКОВ У БАКАЛАВРОВ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

Несмотря на утверждение в 2016 году профстандартов [1] на сегодняшний день недостаточно внимания уделяется аналитическим навыкам при подготовке бакалавров информационной безопасности (ИБ). В связи с этим в статье приводится авторский опыт преподавания дисциплины «Программно-аппаратные средства защиты информации», устраняющий этот недостаток.

Набор навыков современного специалиста по ИБ крайне широк, но условно его можно разделить на две области (в терминах военных учений) – Blue и Red Teams. Каждая из этих команд решает определенные задачи.

Red Team («Красная команда») занимается исследованием компьютерных систем на наличие уязвимостей, иногда в их сферу деятельности входит проникновение на территорию компании с помощью методов социальной инженерии, например, под видом коммунальных служб. Red Team разрабатывает набор инструментов и эксплойтов для тестирования системы на проникновение. «Красной команде» достаточно найти одно слабое звено в системе, чтобы получить доступ, например, к учетным записям пользователей и другим конфиденциальным сведениям. Навыки Red Team можно приобрести, участвуя в соревнованиях по кибербезопасности (CTF).

Blue Team («Синяя команда») ищет следы инцидентов ИБ и реагирует на них. В их компетенцию входит анализ логов (сетевого трафика), собранных от всех критических элементов компьютерной системы, установка и настройка систем защиты информации (SIEM), активная реакция на инциденты и пр. Современные APT-атаки распределены во времени: злоумышленник обычно устанавливает бэкдор, способный обновлять свои компоненты и собирать информацию о целевой системе (см. Kill Chain [2]). Отсюда участникам Blue Team важно обладать аналитическим складом мышления и навыками расследования компьютерных преступлений. Полный перечень навыков «Синей команды» представлен в [3].

На взгляд автора, в вузе целесообразно сосредоточиться на формировании у обучающихся аналитических навыков Blue Team, и помочь в этом может интерактивная среда разработки Jupyter Lab [4], которая входит в состав дистрибутива Anaconda [5].

Впервые широко о применении Jupyter Lab в деятельности специалистов по ИБ заговорил Джон Ламберт [6, 7], инженер компании Microsoft.

Jupyter Lab [8] – это расширяемая клиент-серверная среда разработки, которая поддерживает более 40 языков программирования (или kernels): Python, R, C++ и пр. [9]. Клиентская часть открывается через браузер и позволяет в отдельных ячейках запускать код на выбранном языке программирования.

Исходный код в Jupyter Lab группируется в Блокноты (Notebooks), идею которых переняли из Wolfram Mathematica [10]. Блокноты состоят из набора ячеек. Каждая ячейка может быть исполняемой или содержать комментарии в формате Markdown, включая LATEX. Таким образом, каждый Блокнот превращается в интерактивную статью с возможностью верификации полученных результатов (см. Рисунок 1). Благодаря мировому сообществу ученых Jupyter Lab превратился в одну из самых популярных сред для проведения анализа данных [11].

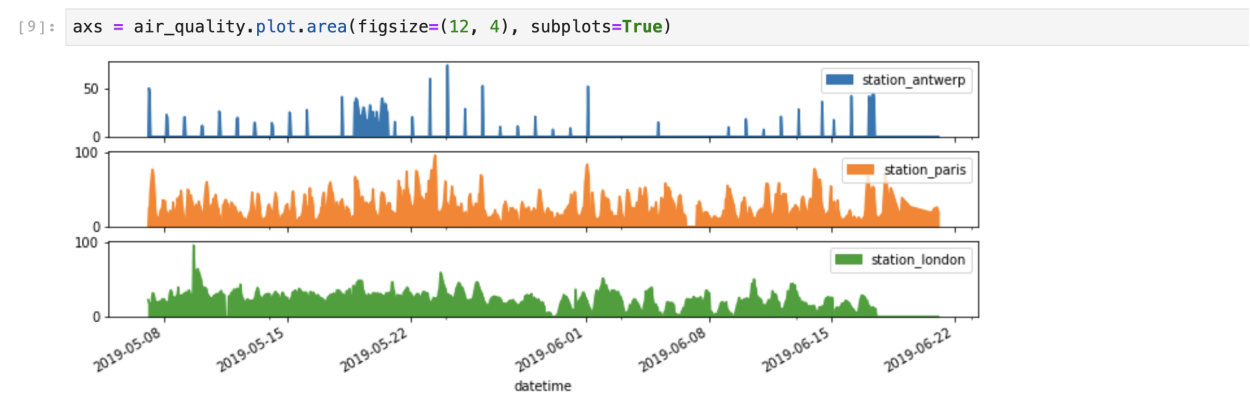

Рисунок 1 – Пример Блокнота Jupyter Lab с графиком и комментариями

Jupyter Lab поддерживает десятки языков программирования, но на сегодняшний день наиболее перспективным для решения задач автоматизации является Python [12]. Он создавался в начале 90-ых с одной стороны, как замена командной оболочке bash, а с другой – как язык для обучения программированию [13]. В связи с этим будем рассматривать схему работы Jupyter Lab (Рисунок 2) на примере языка Python.

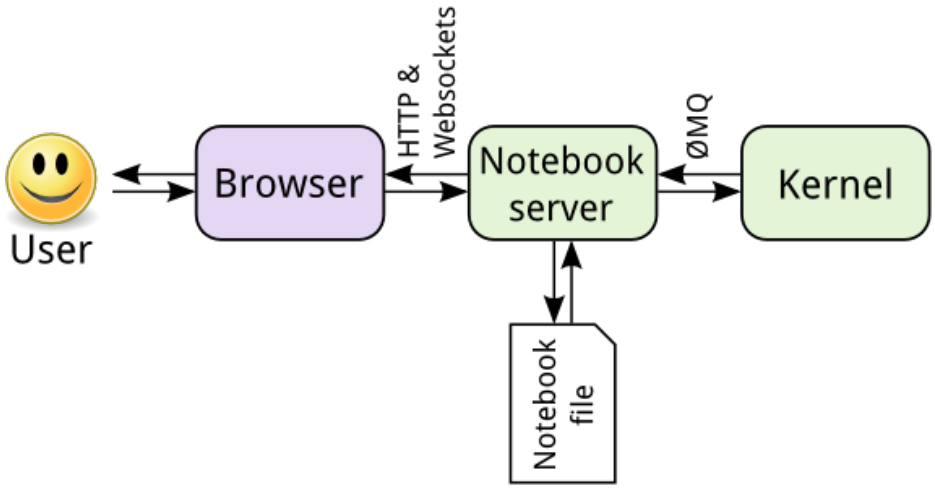

Рисунок 2 – Архитектура среды Jupyter Lab [14]

Пользователь в браузере выполняет код на языке Python. Notebook-сервер обращается к Kernel, в качестве которого выступает приложение IPython – расширенная оболочка языка Python (историю ее создания см. в [15]). Затем, получив результат от IPython, Notebook-сервер генерирует файл в формате JSON с расширением *.ipynb* (Блокнот) и отправляет его обратно клиенту. Таким образом внутри Jupyter Lab происходит интерактивное взаимодействие между браузером и оболочкой IPython.

Помимо встроенных возможностей языка Python приложение IPython (Kernel на Рисунке 2) позволяет в интерактивном режиме обращаться к командному интерпретатору операционной системы (ОС).

Рассмотрим несколько примеров работы в Jupyter Lab.

Первый – получение идентификаторов процессов, принадлежащих пользователю. Данная задача разбивается на несколько шагов.

1) Вывести информацию о всех выполняющихся в ОС процессах:

```
In [1]: !ps aux

USER             PID  %CPU %MEM      VSZ    RSS   TT  STAT STARTED      TIME COMMAND
dm_fedorov      1133  18,6  2,6  5121356 213948   ??  S    27сен20 103:49.27 /Applications/Go
dm_fedorov     40741  13,7  1,8  8903832 150876   ??  S     1:51     0:05.27 /Applications/Go
…
```

2) Выполнить фильтрацию (в методе *grep* указывается аргумент-столбец, по которому производится поиск строки 'dm_fedorov'):

```
In [2]: ps = !ps aux
        ps.grep('dm_fedorov', field=0).fields(1)

['1133',
 '40741',
 '1135',
 '1139',
 '1179',
 '415',
 '39767'
…
```

Детальное описание этого кейса с примером Блокнота приведено в [16].

Второй пример заключается в определении списка подкаталогов в текущем каталоге. Эту задачу с помощью Python можно решить несколькими способами [17], но остановимся только на возможностях, предоставляемых средой Jupyter Lab.

1) Вывести информацию о содержимом текущего рабочего каталога:

```
In [1]: !ls -a ./

.            bash_infosec    log_analysis    scapy
..           elf             networks        traffic-analysis
```

2) Получить содержимое каталога и отфильтровать только подкаталоги:

```
In [2]: import os
        file_list = !ls -a ./
        file_list.grep(os.path.isdir)

['.', '..', 'bash_infosec', 'log_analysis', 'scapy', 'elf, networks', 'traffic-analysis']
```

Представленный способ выполнения команд можно перенести на деятельность специалистов по ИБ. Рассмотрим несколько кейсов, которые демонстрируются в дисциплине «Программно-аппаратные средства защиты информации».

В первом примере Блокноты Jupyter Lab используются для написания YARA-правил [18], служащих одним из форматов индикаторов компрометации [19].

YARA-правило, которое ищет подозрительные строки в любом файле, выглядит следующим образом:

```
rule suspicious_strings
{
```

```
strings:
  $a = "Synflooding"
  $b = "Portscanner"
  $c = "Keylogger"
condition:
  ($a or $b or $c)
}
```

В Блокноте [20] приведенное правило (в файле *suspicious_01.yara*) с помощью утилиты *yara* применяется к различным файлам, например:

```
In [34]: !yara -r yara-rules/suspicious_01.yara samples
```

Второй учебный кейс относится к анализу исполняемых PE-файлов и содержится в Блокноте по ссылке [21]. В процессе анализа файла вычисляется его хеш-сумма:

```
In [13]: import hashlib
         content = open("samples/task-1.exe","rb").read()
         print(hashlib.md5(content).hexdigest())
```

```
a82a243ff5dbf90677c64eae4f0b6a8e
```

Далее с помощью открытого API сервиса VirusTotal [22] на языке Python выполняется поиск файла в базе данных malware по значению хеш-суммы:

```
In [17]: import requests
         api_url = 'https://www.virustotal.com/vtapi/v2/file/report'
         params = dict(apikey='<key>', resource='a82a243ff5dbf90677c64eae4f0b6a8e')
         response = requests.get(api_url, params=params)
         if response.status_code == 200:
            result=response.json()
            print(result)
```

```
{'response_code': 0, 'resource': 'a82a243ff5dbf90677c64eae4f0b6a8e', 'verbose_msg': 'The requested resource is not among the finished, queued or pending scans'}
```

Приведенные Блокноты можно клонировать в собственные github-репозитории для дальнейшего изучения и модификации.

Таким образом, в статье были представлены кейсы по использованию наглядных Блокнотов Jupyter Lab для формирования аналитических навыков у бакалавров ИБ. В качестве перспективы можно отметить возможность построения в Блокнотах моделей машинного обучения для данных, подготовленных учащимися.

Список использованных источников

1. Федоров Д. Ю., Стельмашонок Е. В. Компетенции Ворлдскиллс, трудовые функции профстандартов и повышение качества образования студентов в области защиты информации // Конвергенция цифровых и материальных миров: экономика, технологии, образование. Сборник научных статей международной научной конференции. 21–22 июня 2018 г. Санкт-Петербург. Conference of St.-Petersburg State University of Economics. / Под ред. проф. В.В. Трофимова, В.Ф. Минакова. – СПб.: Изд-во СПбГЭУ, 2018. – С. 269-273
2. Убийственная цепочка или что такое Kill Chain. Блог Алексея Лукацкого. URL: https://lukatsky.blogspot.com/2016/10/kill-chain.html (дата обращения: 06.10.2020)

3. Как стать SOC-аналитиком. Блог Алексея Лукацкого. URL: https://lukatsky.blogspot.com/2020/01/soc_10.html (дата обращения: 06.10.2020)
4. Официальный сайт Project Jupyter. URL: https://jupyter.org/ (дата обращения: 06.10.2020)
5. Официальный сайт дистрибутива Anaconda. URL: https://www.anaconda.com/ (дата обращения: 06.10.2020)
6. The Githubification of InfoSec. John Lambert. URL: https://medium.com/@johnlatwc/the-githubification-of-infosec-afbdbfaad1d1 (дата обращения: 06.10.2020)
7. Гитхабификация Информационной Безопасности. Джон Ламберт. URL: https://habr.com/ru/company/microsoft/blog/487584/ (дата обращения: 06.10.2020)
8. Ian Rose, Grant Nestor. JupyterLab: The Evolution of the Jupyter Notebook: URL: https://www.youtube.com/watch?v=NSiPeoDpwuI (дата обращения: 06.10.2020)
9. Официальный сайт Project Jupyter. URL: https://jupyter.org/try (дата обращения: 06.10.2020)
10. Официальный сайт системы Mathematica. URL: https://www.wolfram.com/mathematica/ (дата обращения: 06.10.2020)
11. Jeffrey M. Perkel. Why Jupyter is data scientists' computational notebook of choice. An improved architecture and enthusiastic user base are driving uptake of the open-source web tool. Nature 563, 145-146 (2018), URL: https://www.nature.com/articles/d41586-018-07196-1 (дата обращения: 06.10.2020)
12. Рейтинг языков программирования TIOBE. URL: https://www.tiobe.com/tiobe-index/ (дата обращения: 06.10.2020)
13. Федоров, Д. Ю. Программирование на языке высокого уровня Python : учеб. пособие для прикладного бакалавриата / Д. Ю. Федоров. – 2-е изд., перераб. и доп. – М. : Издательство Юрайт, 2019. – 161 с.
14. Официальная документация Project Jupyter. URL: https://jupyter.readthedocs.io/en/latest/projects/architecture/content-architecture.html (дата обращения: 06.10.2020)
15. Эволюция командной оболочки Python. Персональный блог Дмитрия Федорова. URL: http://blog.dfedorov.spb.ru/all/evolyuciya-komandnoy-obolochki-python/ (дата обращения: 06.10.2020)
16. Получить идентификаторы процессов, принадлежащих пользователю. Персональный блог Дмитрия Федорова. URL: http://blog.dfedorov.spb.ru/all/poluchit-identifikatory-processov-prinadlezhaschih-polzovatelyu/ (дата обращения: 06.10.2020)
17. Определяем подкаталоги в текущем каталоге. Персональный блог Дмитрия Федорова. URL: http://blog.dfedorov.spb.ru/all/opredelyaem-podkatalogi-v-tekuschem-kataloge/ (дата обращения: 06.10.2020)
18. Пишем YARA правила. Персональный блог Дмитрия Федорова. URL: http://blog.dfedorov.spb.ru/all/pishem-yara-pravila/ (дата обращения: 06.10.2020)
19. Индикатор компрометации (IoC). Персональный блог Дмитрия Федорова. URL: http://blog.dfedorov.spb.ru/all/indikator-komprometacii-ioc/ (дата обращения: 06.10.2020)
20. Notebook. URL: https://nbviewer.jupyter.org/github/dm-fedorov/infosec/blob/master/re-tools/yara-%D0%BF%D1%80%D0%B0%D0%B2%D0%B8%D0%BB%D0%B0.ipynb (дата обращения: 06.10.2020)
21. Notebook. URL: https://nbviewer.jupyter.org/github/dm-fedorov/infosec/blob/master/re-tools/hashes%20%D0%B8%20PE-%D1%84%D0%B0%D0%B9%D0%BB%D1%8B.ipynb (дата обращения: 06.10.2020)
22. API Scripts and client libraries. URL: https://support.virustotal.com/hc/en-us/articles/360006819798-API-Scripts-and-client-libraries (дата обращения: 06.10.2020)