Динамический анализ производится различными сканерами и ручным способом в ходе работы приложения. Статический анализ так же известен как метод белого ящика. В ходе данной проверки весь код ПО анализируется на уязвимости. Стоит отметить, что упомянутые способы поиска OS Command Injection могут подойти для поиска не только инъекций, но и большинства других уязвимостей.

Для защиты от OS Command Injection можно использовать следующие способы:

1. Белые/чёрные списки.
2. Средства защиты веб-приложений.
3. Экранирование символов.
4. Фильтр входных значений.

Белые списки применяются для фильтрации символов, которые разрешено вводить. Чёрные списки, в свою очередь, содержат запрещённые символы. Для приложений с высоким риском несанкционированного доступа рекомендуется использовать белые списки, т. к. они гарантируют большую безопасность относительно второго типа. Примером белого списка может быть список, состоящий только из букв латинского алфавита и цифр. Примером черного списка может служить набор следующих символов: «*{} () <> & * '| = ; [] $ - # ~! . ”% / \: +, `*».

В качестве средств защиты можно использовать WAF (Файрвол веб-приложений), который представляет собой совокупность мониторов и фильтров, необходимых для обнаружения и блокирования сетевых атак на веб-приложение.

Функции экранирования символов заменяют или скрывают в тексте символы, которые могут использоваться для инъекций. Например, если полезная нагрузка будет в виде «`pwd`», то функция преобразует строку в «\`pwd\`», тем самым консоль не распознает команду и вернёт ошибку.

Фильтры входных значений, помимо белых и чёрных списков, проверяют значения на действительность. Например, если в поле ввода необходимо ввести название города, то будет проверяться, существует ли такой город.

Следует подчеркнуть, что при общем подходе данные способы защиты могут подойти не только для OS Command Injection, но и для других типов инъекций.

Литература

1. OWASP Top Ten [Электронный ресурс]. – Режим доступа: https://owasp.org/www-project-top-ten/. Дата доступа: 21.03.2024.

А. А. Воевода
(ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **В. В. Грищенко**, ст. преподаватель

РАЗРАБОТКА ПРОГРАММЫ ДЛЯ СБОРА ИНФОРМАЦИИ ОБ ОПЕРАЦИОННОЙ СИСТЕМЕ WINDOWS И ПОДКЛЮЧЕННЫХ УСТРОЙСТВАХ

При администрировании локальных сетей и устройств, находящихся в них, возникает необходимость иметь информацию о данных устройствах. Для того, чтобы получать такую информацию быстро и удобно, разработаем программу, которая решит нашу задачу.

Перед разработкой, обозначим для себя следующие требования:

– информация должна быть актуальной;
– информация должна быть получена быстро;
– информация должна быть представлена в понятном формате.

Для разработки используем следующие инструменты: язык программирования Python и Windows Management Instrumentation (WMI). WMI - это одна из базовых технологий для централизованного управления и слежения за работой различных частей компьютерной инфраструктуры под управлением платформы Windows [1]. Программа, написанная на языке программирования Python, будет запускаться на компьютере с операционной системой Windows. Информация о компьютере будет собираться с помощью WMI. Получить данные от нашей программы в формате JSON мы сможем с помощью HTTP-запроса.

Для того, чтобы обрабатывать HTTP-запросы, написано простое веб-приложение с использованием фреймворка Pyramid. Данный фреймворк позволяет использовать только необходимые функции, это сказывается на скорости и производительности нашего веб-приложения. Pyramid имеет открытый исходный код, хорошую документацию и надежность, этим обусловлен выбор данного фреймворка [2]. Определим список конечных точек веб-приложения, обращаясь к которым будем получать необходимые ресурсы:

– /api/disks (для получения данных о физических накопителях);
– /api/network (для получения данных о сетевом оборудовании);
– /api/system (для получения данных о операционной системе);
– /api/motherboard (для получения данных о материнской плате);
– /api/cpu (для получения данных о процессоре);
– /api/gpu (для получения данных о видеоадаптерах);
– /api/memory (для получения данных об оперативной памяти);
– /api/all (для получения данных обо всех вышеперечисленных устройствах).

Напишем функции получения данных об устройствах, используя модуль Python WMI. Данный модуль представляет собой легкую оболочку поверх расширений pywin32 и скрывает некоторые запутанные процессы, необходимые для взаимодействия Python с API WMI [3]. Пример функции для получения данных об оперативной памяти представлен на рисунке 1.

```
def memory() -> (dict, bool):
    memory_dict = {'Memory': []}
    if memory_data := c.Win32_PhysicalMemory():
        for mem in memory_data:
            info = {"Manufacture": mem.Manufacturer,
                    "SerialNumber": mem.SerialNumber,
                    "PartNumber": mem.PartNumber.strip(),
                    "FormFactor": MemoryFormFactor(mem.FormFactor).name,
                    "Type": MemoryType(mem.SMBIOSMemoryType).name,
                    "Speed": mem.Speed,
                    "Capacity": size_gb(mem.Capacity)}
            memory_dict['Memory'].append(info)
    return memory_dict
```

Рисунок 1 – Функция сбора данных об оперативной памяти компьютера

Данная функция собирает следующую информацию об оперативной памяти: производитель, серийный номер, код производителя, форм-фактор, тип, скорость работы, объем. Аналогичные функции были написаны для сбора данных об остальных устройствах компьютера.

Используя модули sys, os, subprocess языка программирования Python, был написан скрипт, позволяющий запускать веб-приложение в фоновом режиме и управлять им через командную строку Windows.

Для тестирования нашего веб-приложения используем программное обеспечение Postman. Результат выполнения HTTP-запроса представлен на рисунке 2.

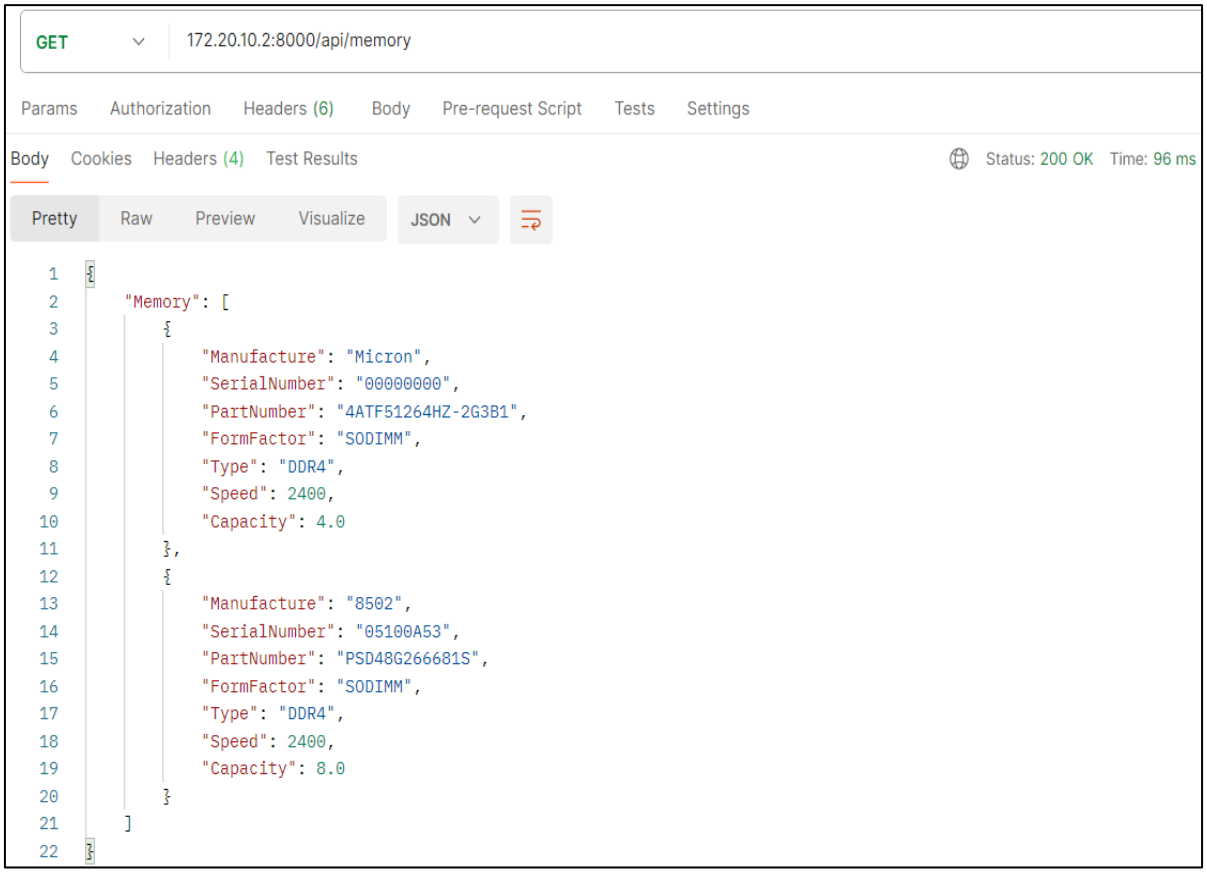

Рисунок 2 – Результат выполнения HTTP-запроса в программе Postman

Как видно из рисунка 2, наш запрос был выполнен за 96 миллисекунд, что является отличным результатом.

Заключение: была разработана программа, которая работает в фоновом режиме, собирает информацию о системе и устройствах компьютера и предоставляет ее в качестве ответов на HTTP-запросы. Данная программа удовлетворяет нашим требованиям и позволяет быстро и удобно получить актуальную информацию о компьютере в локальной сети. В дальнейшем данная программа может быть доработана и использована в качестве агента мониторинга.

Литература

1. WMI [Электронный ресурс] / Свободная энциклопедия Википедия. – Режим доступа: https://ru.wikipedia.org/wiki/WMI. – Дата доступа: 21.03.2024.

2. Pyramid Introduction [Электронный ресурс] / The Pyramid Web Framework. – Режим доступа: https://docs.pylonsproject.org/projects/pyramid/en/2.0-branch/narr/introduction.html. – Дата доступа: 21.032024.

3. WMI [Электронный ресурс] / The Python Package Index. – Режим доступа: https://pypi.org/project/WMI/. – Дата доступа: 21.03.2024.