RED

Методика проверки кода на уязвимости RepoJacking

Описание инструментов и методов исследования кода

Данная инструкция позволит найти в коде все ссылки на репозитории, контроль над которыми может быть получен злоумышленниками. Методика поиска уязвимых зависимостей создана на основе исследования команды MTC RED ART, в ходе которого были проверены и успешно применены все нижеперечисленные компоненты.

1. Извлечение ссылок, ведущих на GitHub

Для получения списка ссылок, ведущих на https://github.com из вашего репозитория, потребуется:

1.1 Скачать скрипт pars_url.py из нашего набора https://github.com/RED-Advanced-Research-Team/Supply_chain_research.

1.2 Создать текстовый файл MY_rep.txt в том же каталоге, где расположен pars_url.py.

1.3 Записать в MY_rep.txt полный путь до вашего репозитория (например, https://github.com/RED-Advanced-Research-Team/Supply_chain_research).

1.4 Результатом выполнения скрипта будет создание файла return_github.txt, который будет содержать все ссылки из вашего кода ведущие на https://github.com/.

2. Получение битых ссылок (corrupted links)

Для поиска битых ссылок вам потребуется применить утилиту, входящую в стандартный набор Kali Linux — nuclei (инструмент для отправки запросов на основе шаблонов с ожиданием заданного ответа).

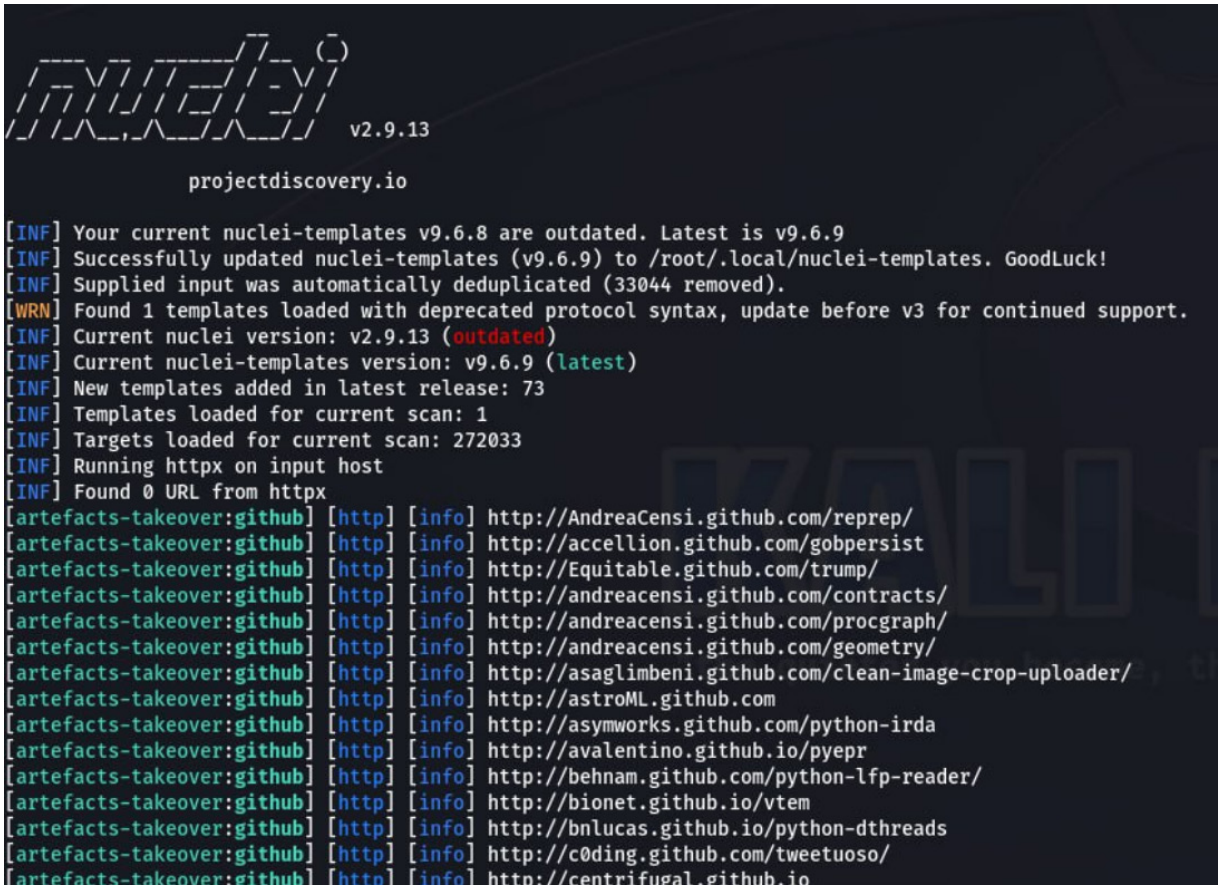
2.1 Для получения необходимого результата вам потребуется шаблон для nuclei, который даст команду реагировать на следующие фразы: "Sign in to GitHub" и "a GitHub Pages site here For root URLs". Готовый шаблон с этими функциями лежит в нашем репозитории под названием artefacts_github.yaml.

2.2 Следующим шагом потребуется в произвольный каталог (например: /opt) перенести два файла: return_github.txt и artefacts_github.yaml.

2.3 Заключительный шаг данного этапа — выполнение команды: nuclei -t artefacts_github.yaml -l return_github.txt -o result_github.txt -stats -elog error_github.txt, где:

-t	путь до шаблона
-l	путь до списка со всеми ссылками GitHub, полученными из вашего кода
-o	файл с результатом выполнения
-stats	онлайн-вывод процесса обработки
-elog	файл с потенциально возможными ошибками

В результате выполнения команды в файле result_github.txt вы получите список битых ссылок из вашего кода.



В частности, при поиске по всем публичным репозиториям Google Cloud Console и менеджерам пакетов PyPi и NPM мы обнаружили 7820 битых ссылок (corrupted links).

3. Проверка аккаунтов пользователей, доступных для перерегистрации с целью подмены репозиториев по неработающим ссылкам (уязвимость для атаки типа RepoJacking)

Все битые ссылки стоит удалить из кода. Однако дополнительно мы добавим явный метод выяснить, какие из них могут быть получены злоумышленниками в любой момент через атаку типа RepoJacking.

3.1 Для подтверждения перерегистрации владельца битой ссылки вам потребуется скрипт clicker_github.py из нашего набора https://github.com/RED-Advanced-Research-Team/Supply_chain_research.

3.2 Также потребуется отфильтровать полученный ранее файл result_github.txt таким образом, чтобы в нем остались только имена пользователей (пример: из https://github.com/RED-Advanced-Research-Team/Supply_chain_research необходимо получить только RED-Advanced-Research-Team). Назовём новый файл result_github_user.txt.

3.3 Далее потребуется произвольный почтовый ящик, к которому вы имеете доступ и пароль.

3.4 Получив все необходимые компоненты, открываем файл clicker_github.py и заменяем два параметра – test@mail.com на ваш почтовый ящик и test_password на сгенерированный вами пароль и сохраняем.

При запуске скрипта clicker_github.py вы сможете в режиме реального времени наблюдать возможность перерегистрации каждого из аккаунтов в списке result_github_user.txt.