



작성자	심주완
분석 일자	2024.05.16
작성 일자	2024.05.16
분석 대상	zoo.pcap
문서 버전	3
작성자 E-mail	rd002@naver.com

0. 목차

1. 문제 3

2. 분석 도구 3

3. 환경 3


4. Write-Up..... 4

5. Flag 8

6. 별도 첨부 9

7. Reference10

1. 문제

URL	https://dreamhack.io/wargame/challenges/1205
문제 내용	<p>드림핵 공간에는 엄청난 동물원이 있어요.</p> <p>주어진 pcap 파일을 분석하여 동물원으로 가보세요!</p> <p>동물원에서 많은 동물들을 살펴보고 플래그를 찾으세요!</p>
문제 파일	 <p>zoo.pcap</p>
문제 유형	Network forensics
난이도	2 / 3

2. 분석 도구

도구명	다운로드 링크	Version
wireshark	https://www.wireshark.org/	4.2.5
OpenStego	https://github.com/syvaidya/openstego/releases	0.8.6

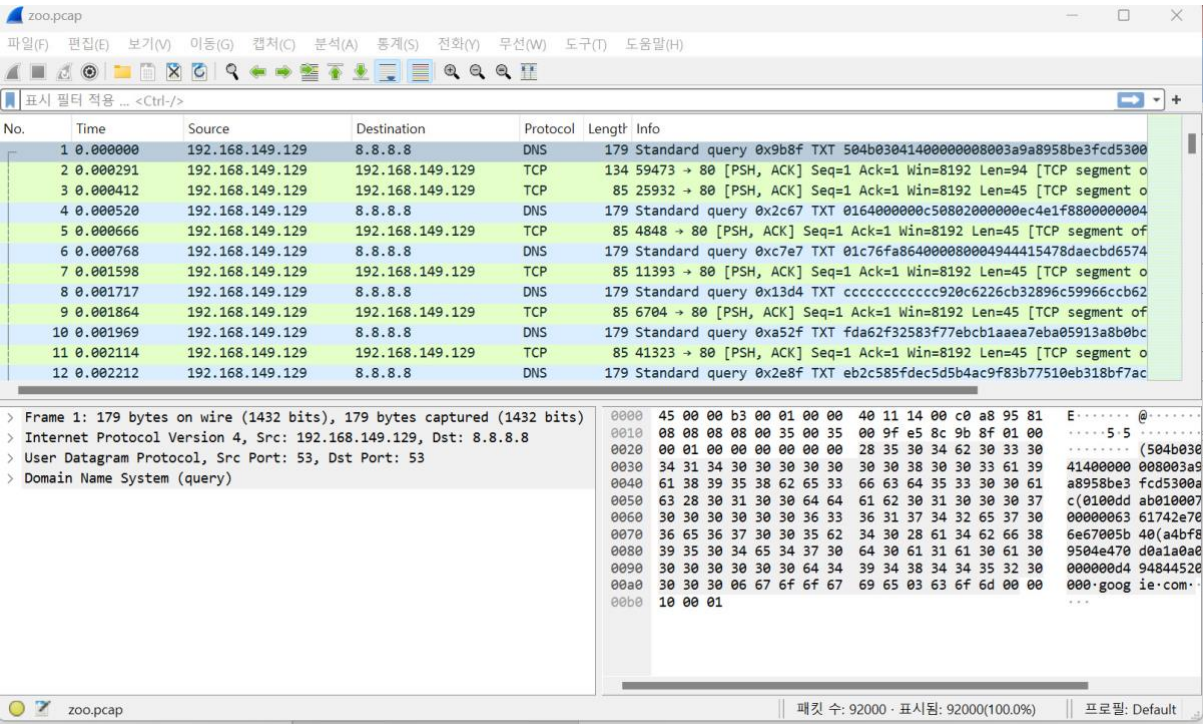
3. 환경

OS
Window 11 Home

4. Write-Up

파일명	zoo.pcap
용량	13.1 MB
SHA256	011d77a04cf6c313c7eb80ac8639f07ce2f41632c18c10cff5489bacc7792d93
Timestamp	2024-04-10 06:20:46

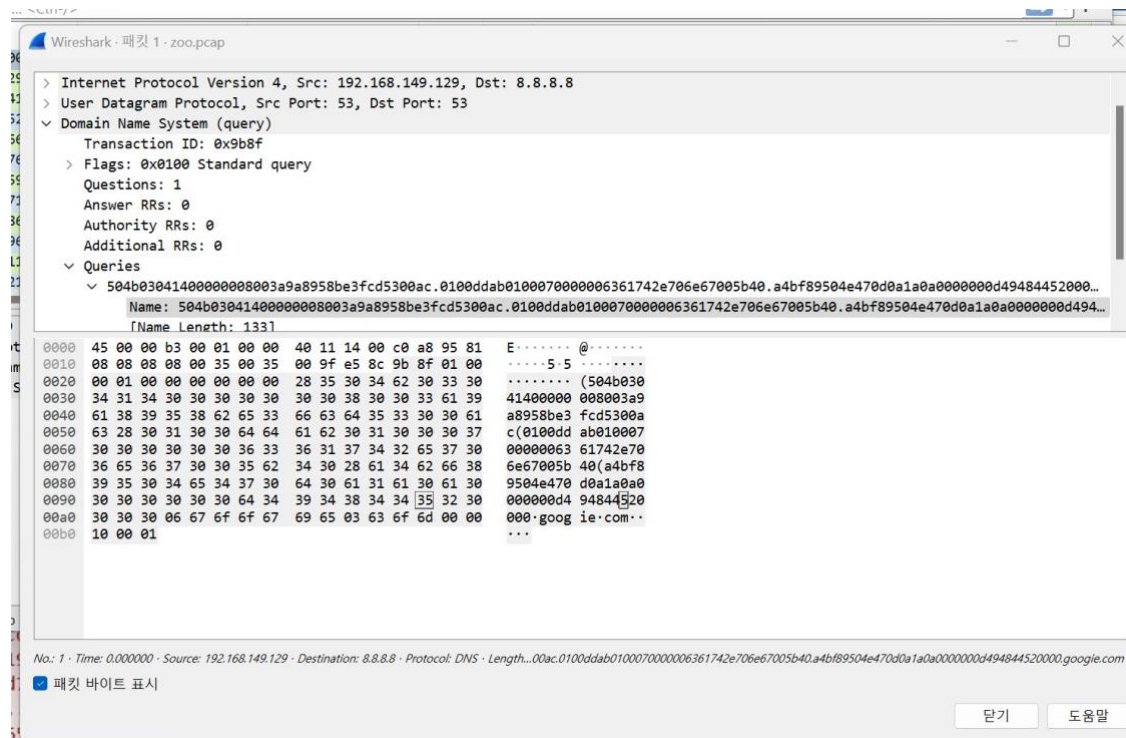
네트워크 포렌식 문제는 처음이라 쉽지가 않았다. 역시 실전에서 많이 배우는구나 하고 느꼈던 문제인 것 같다. 먼저 wireshark 를 통하여 문제 파일인 zoo.pcap 를 열어보자.



[사진 1] WireShark를 통하여 연 zoo.pcap

다음과 같이 확인할 수 있는데, DNS를 확인해보면 계속 TXT 쿼리문을 날리고 있다. 자세히 살펴보자.

[WHS-2] .iso



[사진 2] DNS 패킷

확실히 비정상적인 패킷을 보내고 있다. 의심되는 루트 도메인이 google.com이라는 정보도 확인할 수 있다. 그렇다면 보내고 있는 정보를 종합하여 어떤 정보를 보내려고(유출하려고) 하는지 확인해볼 수 있다. PowerShell을 통하여 다음과 같은 명령문을 실행시켰다.

```
$ tshark -r zoo.pcap -Y "dns.flags.response==0 && dns.qry.type==16"
-T fields -e dns.qry.name | grep google.com | sed "s/\.//g" | sed
"s/google//g" | sed "s/com//g" > dump
```

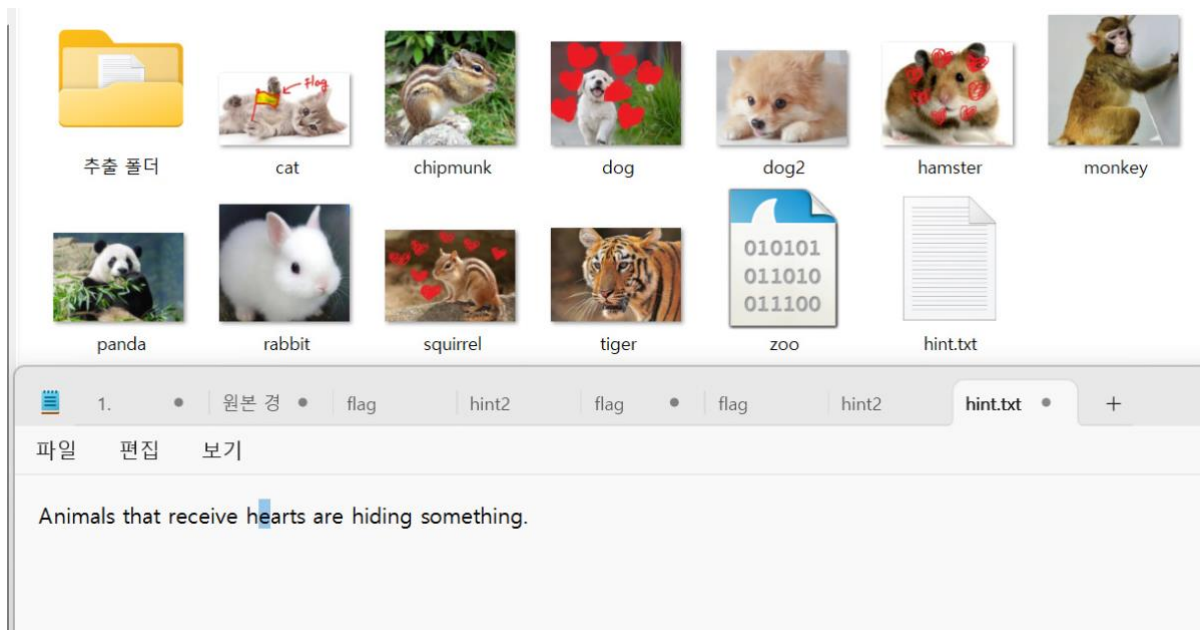
이 코드는 총 가지로 나뉜다.

1. tshark를 통하여 zoo.pcap를 연다.
2. 플래그가 0(사용자가 보낸 형식) 과 쿼리 타입이 16(txt)인 부분만 검색한다.
3. 필드 타입으로 나누고, 쿼리의 name 부분(txt형식의 내용이 있는 부분)을 출력한다.
4. 출력문에서 Google.com을 빼고 통합한다.

```
$ xxd -p -r dump > extract
$ file extract
```

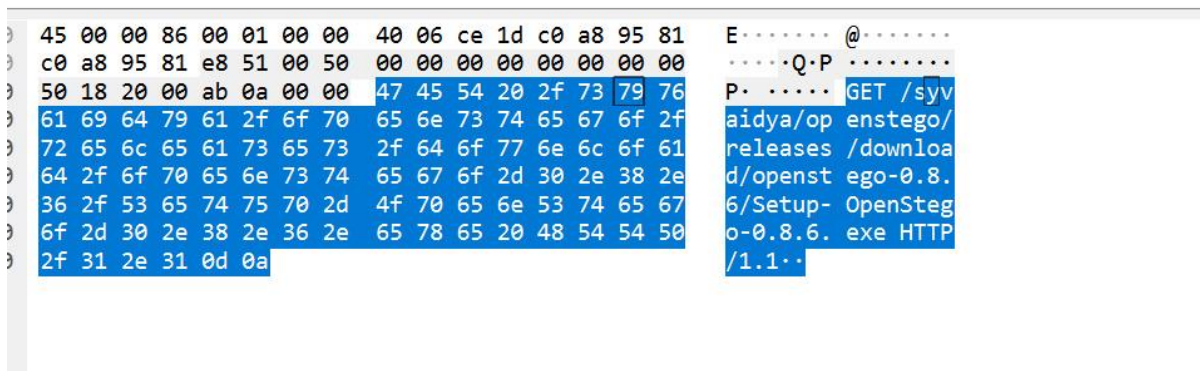
다음 코드를 통하여 최종적으로 extract 파일을 만들 수 있었다. 압축을 풀고 내부를 확인하면,

[WHS-2] .iso



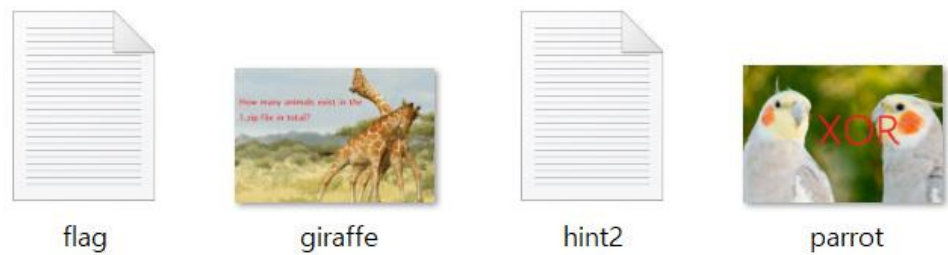
[사진 3] 추출 파일

다음과 같은 사진들을 확인할 수 있었다.(추출 폴더는 같이 추출된 파일이 아니다. 어떻게 생성하였는지는 뒤에 설명하겠다) 이 이유 때문에 Dream Zoo 라는 제목을 사용했구나 싶었다. hint.txt 의 파일 안에는 다음과 같은 내용이 있었다. 하트가 있는 사진을 조사하라는 내용은 있었지만 어떤 방법을 통하여 조사를 해야할지 막막했다. 이에 대한 답은 TCP 패킷에서 확인할 수 있었다.



[사진 4] TCP패킷 내용

OpenStego로 패킷을 날리는 과정을 확인할 수 있었고, 이는 사진 속 스테가노그래피를 확인할 수 있는 프로그램이었다. 이를 통해서 하트가 보이는 동물들의 사진을 OpenStego로 분석해보았다.



[사진 4] 추출 파일 내부

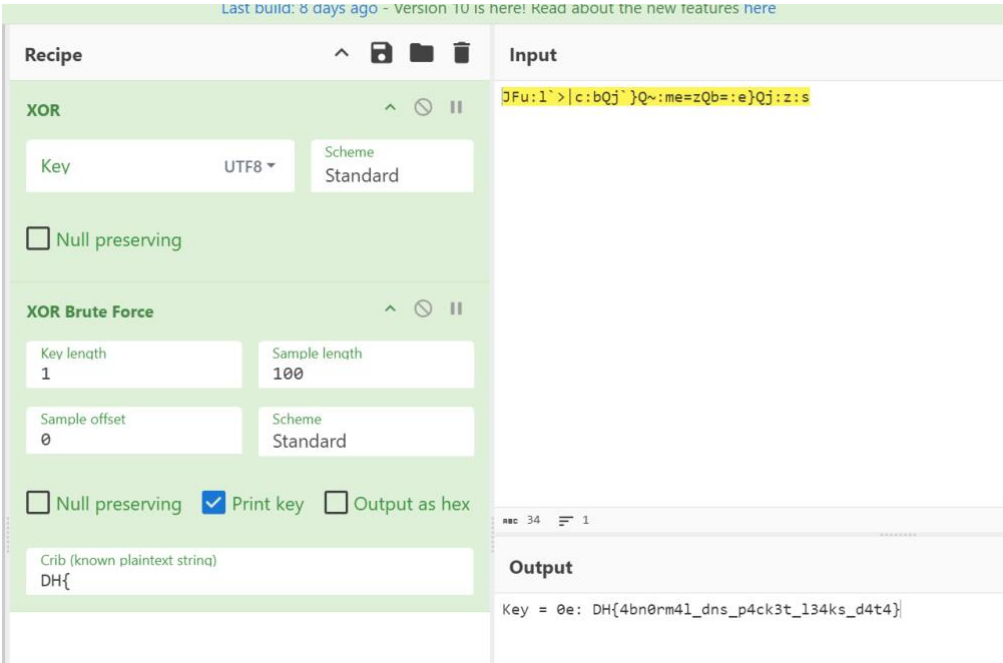
이를 전부 추출한 파일이 아까 확인하였던 추출 폴더이다. hint2.txt 의 파일의 내용은 다음과 같았다.

(cat's hidden) (squirrel's hidden) (dog's hidden) = FLAG

flag.txt 파일이 각 cat's hidden 이고, parrot 파일이 squirrel's hidden 이고 giraffe 파일이 dog's hidden 이다. 이에 해당하는 값을 집합시키면 플래그 값은

JFu:l`>|c:bQj`Q~:me=zQb=:e}Qj:z:s XOR 14

를 연산한 값이 될 것이다. 나의 경우 giraffe 에 해당하는 값을 14 가 아닌 12 로 놓고 풀어 계속 답이 나오지 않았다. 결국 XOR Brutefoce tool 을 사용해서 풀긴 했다. 사이트는 레퍼런스에 넣어두었다.



[사진 5] flag

5. Flag

DH{4bn0rm4l_dns_p4ck3t_l34ks_d4t4}

6. 별도 첨부

7. Reference

- [https://gchq.github.io/CyberChef/#recipe=XOR\(%7B'option':'UTF8','string':'"%7D','Standard',false\)XOR_Brute_Force\(1,100,0,'Standard',false,true,false,'DH%7B'\)&input=SkZ1OmxgPnxjOmJRamB9UX46bWU9elFiPTplfVFqOno6cw](https://gchq.github.io/CyberChef/#recipe=XOR(%7B'option':'UTF8','string':')