2021 한국차세대컴퓨팅학회
춘계학술대회

장　　소 : 김대중컨벤션센터 301호 ~ 303호
일　　시 : 2021. 5. 13(목) 13:00 ~ 5. 14(토) 12:30
주최 · 주관 : 한국차세대컴퓨팅학회
후 원 : 광주관광재단, ㈜트라콤, 아주대학교 MR-IoT 재난대응인공지능연구센터

웹 브라우저 렌더링을 활용한 컨테이너 기반 다크웹 스캐너 설계

Design a Container-base Dark Web Scanner using Web Browser Rendering

Chanmo Yang
Information Security
Sejong University
Seoul, Korea
yanglampp@gmail.com

Junghoon Lee
Information Security
Sejong University
Seoul, Korea
jhleestock@gmail.com

Seonghwan Kim
Information Security
Sejong University
Seoul, Korea
hellheit2@gmail.com

Jiwon Park
Information Security
Sejong University
Seoul, Korea
giwon9977@gmail.com

Ki-Woong Park†
Information Security
Sejong University
Seoul, Korea
woongbak@sejong.ac.kr

Abstract

다크 웹은 일반 브라우저(Chrome, FireFox 등)를 통해 접근할 수 없고, 특정 브라우저(Tor)를 이용하여 최소 3개의 익명 노드를 거쳐 접근하는 익명성이 보장된 웹 사이트이다. 익명성의 보장으로 다크 웹 내에서는 마약, 총기 거래 등 다양한 범죄가 발생하고 있다. 다크 웹 내에서 발생하는 범죄의 이해를 높이려면 해당 서비스에서 제공하는 콘텐츠(이미지, 비디오 등)에 대한 포괄적인 정보 수집 및 보관이 필요하다. 또, 이를 일반 브라우저에서 확인할 수 있게 한다면 다크 웹의 실상에 다수가 보다 쉽게 접근할 수 있다. 이를 위해 본 연구에서는 다크 웹 불법 서비스의 당시 페이지를 일반 브라우저에서 렌더링 할 수 있는 형태로 저장하는 컨테이너 기반 다크 웹 스캐너를 제안한다. 다크 웹 접속에 필요한 토르(Tor) Proxy 컨테이너를 다중 생성하고, 다수의 다크 웹 내 HTTP 요청을 병렬 처리하여 속도 성능을 개선했다. 제안한 방법과 토르(Tor) 브라우저를 통한 렌더링 속도를 비교 실험하였고, 제안한 방법이 토르(Tor) 브라우저 렌더링 속도에 비해 약 2배 빠름을 확인했다. 또한, 일반 브라우저에서 다크 웹 접속 없이 해당 서비스와 동일하게 렌더링 할 수 있음을 확인했다.

Keywords: 다크 웹, 컨테이너, 스캐너, 토르(Tor)

1. Introduction

최근 코로나로 인해 전 세계적으로 온라인 범죄가 증가하고 있다. 일반 웹 환경에서의 범죄뿐만 아니라 다크 웹 내의 불법 거래 또한 증가하고 있다.[1] 다크 웹은 최소 3개의 익명 노드를 거쳐 접속되는 웹 사이트이다. 이 익명 노드들은 최초 출발지와 목적지를 알지 못하며, 바로 이전 노드와 이후 노드만을 알고 있다. 이러한 익명성의 보장으로 많은 불법 서비스가 운영되고 있다. 대표적으로 마약, 총기, 위조 화폐 거래와 음란물 서비스들이 운영되고 있다. 이런 서비스들은 주로 이미지 파일이나 동영상 파일을 통해 이용자들에게 정보를 제공한다. 또한 최근 웹 서비스들은 반응형 디자인을 통해 정보를 제공하기 때문에, 평문 HTML 코드만을 저장하는 기존 다크 웹 스캐너들은 정보 수집이 제한적이다. 이런 문제를 해결하기 위해 본 연구에서는 Figure 1과 같은 스캐너를 제안한다. 가장 먼저 스캐너는 HTML 내에 콘텐츠를 호출하는 태그와 속성을 먼저 탐지한다. 이후 해당 콘텐츠와

†교신 저자

HTML 파일을, 토르(Tor) Proxy 컨테이너를 다수 생성하여 병렬적으로 다운로드한다. 저장한 HTML 파일과 콘텐츠들을 일반 브라우저에서 렌더링 될 수 있도록 경로와 이름을 매핑(Mapping)하여 재저장한다. 사용자는 일반 브라우저를 통해 탐지 당시의 다크 웹 불법 서비스와 동일한 콘텐츠를 제공하는 사이트를 확인할 수 있다.

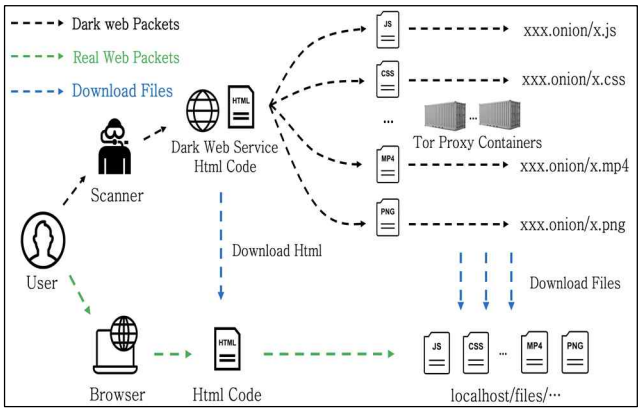

Figure 1. 제안한 스캐너의 Overview

본 논문의 구성은 다음과 같다. 2 장에서 기존 다크 웹 스캐너의 한계점과 제안하는 스캐너의 차이점을 설명한다. 3 장에서는 제안한 스캐너에서 HTML 내 렌더링에 필요한 태그와 주소를 탐지하는 수집기와 Tor Proxy 컨테이너를 구성하는 방법에 대해 설명한다. 4 장에서는 다크 웹 내 마약 거래 사이트 1 개를 대상으로, 토르(Tor) 브라우저와 제안한 스캐너를 비교 실험한다. 마지막 5 장에서는 결론과 향후 연구를 설명한다.

2. Related Works

웹 브라우저 렌더링 및 스크립팅 작업 제거를 통한 토르(Tor) 기반 다크 웹(Dark Web) 수집 성능 개선[2] 연구에서는 일반적으로 속도가 느린 다크 웹의 수집 성능을 개선하기 위해, 웹 브라우저 렌더링 및 스크립팅 작업이 제거되는 명령행 인터페이스 기반 방식과, 단일 토르(Tor) 애플리케이션 Proxy를 이용한다. 토르(Tor) 브라우저를 사용하지 않고 명령행 인터페이스 기반으로 정보를 수집할 경우, 다크 웹 내 불법 서비스에서 제공하는 이미지 파일, 비디오 파일 등과 같은 중요한 시각화 정보들을 수집하지 못한다. 또한, CSS, JS와 같은 파일들도 수집하지 못하기 때문에 수집한 정보를 평문 Text로 밖에 확인하지 못하는 한계점이 있다.

본 연구에서 제안하는 스캐너도 명령행 인터페이스 기반으로 구성된다. 그러므로 토르(Tor) 브라우저를 통해 직접 웹 렌더링을 수행하지 않는다. 하지만, 렌더링에 필요한 파일(HTML, CSS, JS, Image)들을 미리 다수의 토르(Tor) Proxy 컨테이너와 명령행 인터페이스 기반 스캐너를 통해 병렬적으로 다운로드한다. 해당 파일들과 HTML 코드 내 주소를 매핑하여, 다크 웹 접속이 필요 없는 일반 브라우저(Chrome, IE 등)를 통해 웹 렌더링을 수행한다.

3. Methods

본 장에서는 제안한 스캐너를 구성하는 HTML 렌더링 호출 태그 및 수집기와 HTTP 요청을 토르(Tor) Proxy 컨테이너에 대해 설명한다.

3.1. HTML 렌더링 호출 태그 및 주소 수집기

HTML 코드 내에서 브라우저 렌더링에 사용되는 파일을 로드하기 위해서는 특정 HTML 태그들과 속성이 필요하다. 해당 수집기는 HTML 내에서 콘텐츠를 다운로드 하기 위한 주소를 얻기 위해 탐지할 태그와 속성을 <Table 1>와 같이 지정하여 탐지한다. 탐지한 태그의 속성에는 콘텐츠를 가리키는 주소가 명시되어 있다. HTML 내에서 주소를 가리키는 방법은 크게 외부 호스트의 절대 주소, 호스트 본인의 절대 주소, 호스트 본인의 상대 주소가 있다.

<Table 1> 외부 콘텐츠를 로드하는 태그 및 속성

Tag	Attributes	Tag	Attributes

<a>	href	<link>	href
<applet>	codebase	<object>	classid
<area>	href	<object>	codebase
<base>	href	<object>	data
<blockquote>	cite	<object>	usemap
<body>	background	<q>	cite
<del>	cite	<script>	src
<form>	action	<audio>	src
<frame>	longedesc	<button>	formaction
<frame>	src	<command>	icon
<head>	profile	<embed>	src
<iframe>	longdesc	<html>	manifest
<iframe>	src	<input>	formaction
<img>	longdesc	<source>	src
<img>	src	<video>	poster
<img>	usemap	<video>	src

파일을 가리키는 주소를 Request를 통해 다운로드 할 수 있는 호스트를 포함한 절대 주소로 변경하여 일괄 저장한다. 저장된 주소는 스레드를 통해 토르(Tor) Proxy 컨테이너로 분할 전송된다.

3.2. 토르(Tor) Proxy 컨테이너

일반적으로 다크 웹 스캐너들은 다크 웹 내의 웹 사이트에 HTTP 요청을 보내기 위해서 Socks5 기반으로 통신하는 토르(Tor) Proxy를 사용한다.[3] 토르(Tor) Proxy를 구동하기 위해서는 토르(Tor) 애플리케이션을 실행해야 한다. GUI 환경의 OS에서는 토르(Tor) 브라우저, CLI 환경의 OS에서는 토르(Tor) 실행 파일을 통하여 Proxy를 구성한다. 본 연구에서는 많은 파일을 빠른 시간에 저장하기 위해, 토르(Tor) 애플리케이션(실행 파일)을 호스트 OS에서 단일로 실행하지 않고, Docker 컨테이너로 구성하여 다중으로 실행한다. 이렇게 컨테이너화 하여 Proxy를 구성할 경우, 다중 HTTP 요청을 다수의 Proxy에 분할 처리하여 속도적 측면에서 성능을 개선할 수 있다. 토르(Tor) Proxy 컨테이너를 생성하는 Dockerfile은 Figure 2와 같다.

```
1 FROM alpine:latest
2 RUN apk update && apk add tor
3 COPY CustomTorrc /etc/tor/CustomTorrc
4 RUN chown -R tor /etc/tor
5 USER tor
6 ENTRYPOINT ["tor"]
7 CMD ["-f", "/etc/tor/CustomTorrc"]
```

Figure 2. 토르(Tor) Proxy 컨테이너를 생성하는 Dockerfile

컨테이너 내에서 실행하는 토르(Tor) 애플리케이션의 설정 파일은 Dockerfile과 같은 경로에 CustomTorrc 파일에 “SocksPort:9050”을 기입하여, 9050 포트에 Socks Port를 할당한다. 이후 Docker build 명령어를 통해 도커 이미지 파일을 생성한 후, “docker run -d [이미지 명]” 명령을 반복 수행하여, 컨테이너를 다중으로 실행한다. 이렇게 생성된 컨테이너들에 요청할 패킷을 병렬 및 분할하여 요청할 수 있다.

4. Experiments

본 장에서는 다크 웹 내 불법 서비스 1개를 대상으로, 토르(Tor) 브라우저에서의 특정 페이지 렌더링 완료 시간과 Python 언어 기반의 병렬 스레드 기능과 3개의 Tor Proxy 컨테이너로 일반 브라우저에서 렌더링 되도록 파일을 다운로드 하는 시간을 비교한다. 또한, 다운로드 완료된 파일이 다크 웹 접속 없이, 일반 브라우저(Chrome)를 통해서 정상적으로 렌더링 될 수 있는지 검증한다.

4.1. Experimental setup

우선 실험에 사용된 환경은 macOS와 Intel i9 (8 Core) CPU, 16 GB(DDR4-2400) Memory 자원을 사용하여 수행했다. 해당 환경에 토르(Tor) Proxy 컨테이너를 3개 구동시키고, HTTP 요청을 병렬로 전송할 스레드는 Python 언어와 current 패키지의 futures 모듈 내 ThreadPoolExecutor와 as_completed를 사용하였다.

4.2. Experimental result

본 연구에서 제안한 스캐너를 구동시켰을 때 단일 사이트 페이지를 대상으로 실험 결과는 Figure 3과 같다. 제안한 스캐너를 통한 방식은 평균 15.17초가 소요되었으며, 토르(Tor) 브라우저는 평균 34.32초가 소요되었다. 이를 통해 속도의 성능이 약 2배 정도 빠름을 확인하였다.

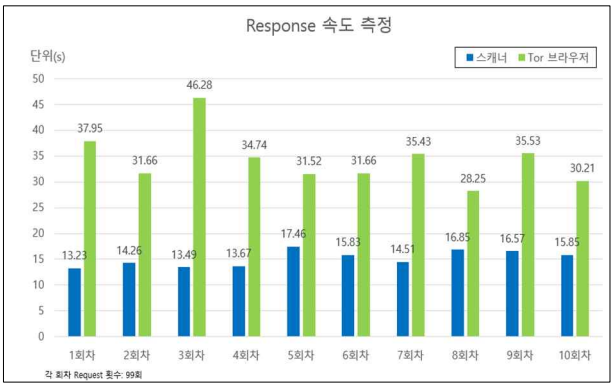

Figure 3. 스캐너와 토르(Tor) 브라우저 렌더링 속도 비교

또한, 스캐너를 통해 다운로드한 파일과 HTML을 일반 브라우저(Chrome)에서 렌더링 했을 경우 Figure 4와 같이 토르(Tor) 브라우저를 통해 렌더링된 결과와 동일하게 동작함을 확인했다.

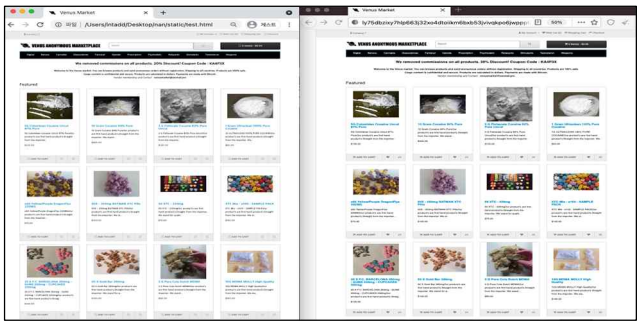

Figure 4. 일반 브라우저 와 토르(Tor) 브라우저 렌더링 비교(좌측 Chrome, 우측 토르(Tor))

대상 페이지는 PNG 파일 89개, JS 파일 6개, CSS 파일 4개를 로드하는데, Figure 4와 같이 사용된 파일들이 정상적으로 렌더링 됨을 확인했다.

5. Conclusions

본 연구는 토르(Tor) Proxy 컨테이너를 다수 생성하고, 해당 Proxy 컨테이너에 HTTP 요청을 분할 및 병렬 전송하여, 빠른 속도로 렌더링에 필요한 파일을 다운로드하는 컨테이너 기반 스캐너를 제안하였다. 이를 통해 당시 다크 웹 불법 서비스에서 제공하는 시각 정보를 다크 웹 접속 없이 일반 웹 브라우저를 통해 확인 및 수집할 수 있다. 본 논문에서는 다크 웹 불법 서비스가 수행하는 기능적인 측면까지 확인할 수 없다는 한계점이 있다. 향후 연구에서는 해당 스캐너를 통해 다크 웹 정보를 하나의 공용 데이터베이스에 중앙화하여, 불법 서비스의 기능을 유추할 수 있는 연구를 수행할 계획이다.

Acknowledgement

본 연구는 2019년도 과학기술정보통신부의 재원으로 정보통신기획평가원의 지원(No.2019-0-00426) 및 한국연구재단 연구과제(NRF-2020R1A2C4002737)의 지원을 받아 수행된 연구임

References

[1] ElBahrawy, Abeer, et al. "Collective dynamics of dark web marketplaces." Scientific reports 10.1 (2020): 1-8.

[2] 문현수, 김수현, and 이영석. "웹 브라우저 렌더링 및 스크립팅 작업 제거를 통한 토르 (Tor) 기반 다크 웹 (Dark Web) 수집 성능 개선." 정보과학회논문지 47.10 (2020): 1008-1013.

[3] Sarah Jamie Lewis, (2016). “onionscan”, GitHub repository, https://github.com/s-rah/onionscan