졸업 작품 보고서

액세스 로그를 활용한 비정상적으로 높은 접근에 대한 이상 탐지

2023년 11월

서경대학교 컴퓨터공학과

황유현

목 차

Ⅰ. 서론

1.1 제작 동기 및 목적

단순 검색부터 IoT, 클라우딩 컴퓨팅 등 다양한 분야에 널리 사용되고 있는 웹 서버는 높은 접근성을 가진 만큼 다른 시스템보다 더 높은 보안을 필요로 한다. 하지만 해결하기 힘든 다양한 취약점으로 인해 개인정보 유출, 서버 마비 등 피해를 입는 경우가 많다. 웹 서버의 로그 데이터 안에는 일반 사용자뿐만 아닌 여러 피해를 발생시키는 악의적 사용자의 활동 흔적이 기록된다. 로그에 기록되어있는 사용자의 정보를 이용하여 사용자의 행동 패턴을 파악하고, 이를 통해 악의적인 접근을 방지하고 통제하고자 한다.

웹 취약점을 이용한 공격 기법 중 하나인 DDoS 공격은 계속해서 다양한 형태로 변화하며 공격이 증가하는 추세를 보이고 있으며, 이로 인한 경제적 피해는 물론이며 응급상황 발생 시 제어불능으로 인해 인명 피해까지 발생시킬 수 있다. 최근 나타나고 있는 DDoS 공격의 특징을 파악하고, DDoS 공격을 예방하는데 효과적인 방법들에 대해 알아보았다.

Ⅱ. 관련 연구 및 동향

2.1 DDoS 공격

2.1.1 DoS/DDoS 공격

DoS 공격이란 공격 대상 호스트로 대량의 네트워크 트래픽을 발생시켜 대상 호스트이 네트워크 서비스 기능을 일시적으로 또는 완전히 정지시키는 공격 유형을 말한다. 그 중에서 DDoS 공격은 이러한 DoS 공격을 하나가 아닌 여러 개의 호스트가 담당하는 데 그 차이가 있으며, 일반적인 DoS 공격보다 훨씬 더 강력한 파괴력을 가지고 있다.

DDoS 공격에 대해 좀 더 구체적으로 살펴보면 아래와 같다.

2.1.2 DDoS

DDoS 공격은 1990년대 말경에 나타난 공격 유형이다. 기존의 DoS 공격은 공격 대상에 제약을 가지고 있고, 공격자의 추적 및 공격 행위의 차단이 가능하다는 취

약점을 가지고 있다. DDoS 공격은 이러한 DoS 공격 방법에 분산 처리 개념을 도입하고, 기존의 DoS 공격 툴들의 공격 형태를 자동화하도록 만들어진 DDoS 공격 툴들을 이용하여 모든 공격이 자동화된 공격법을 말한다. 서버(Server) - 에이전트(Agent)로 구성된 2tier 방식을 기본으로 하여, 많은 수의 호스트들에 대량의 패킷을 전송할 수 있는 공격용 프로그램들이 인터넷으로 연결된 여러 시스템에 분산 설치된다. 이들이 서로 통합된 형태로 특정 공격 대상 시스템이나 네트워크에 대하여 일시에 많은 양의 데이터 패킷을 보냄으로써 공격 대상 시스템의 성능 저하 또는 시스템 마비를 유발시키는 공격 형태를 가진다. 한 대의 컴퓨터가 실시하는 DoS 공격과 달리, 수많은 컴퓨터가 동시에 공격을 수행하기 때문에 강력한 파괴력을 가지게 된다.

2.2 DDoS 공격 유형

DDoS의 공격 유형은 DoS 공격을 분산화, 자동화한 형태이므로, DoS의 공격 유형을 먼저 살펴보도록 한다. DoS 공격을 수행하기 위해서 사용되는 공격 유형은 크게 Flooding 공격과 Malformed Packet 공격으로 나누어 볼 수 있다. 전자의 공격 유형은 대량의 패킷을 단시간에 특정 호스트에 집중시켜 공격 대상 호스트가 정상적인 서비스를 하지 못하게 하는 공격으로 TCP SYN Flooding 공격, TCP Flood 공격, UDP Flood 공격, Smurf Flood 공격 등이 있다. 후자의 공격 유형은 운영체제나 프로토콜상 취약점을 이용하여 비정상적인 형태의 패킷을 보내어 시스템을 마비시키는 공격으로 Ping of Death 공격, Chargen 공격, Tear Drop 공격, Land 공격, Win Nuke 공격 등이 있다.

이러한 DoS 공격 중 많은 공격들이 부분적으로 사용하고 있는 IP 스푸핑 기법 또한 간과할 수 없다. IP 스푸핑은 공격 대상 시스템에 접근하거나 패킷을 보낼 때, IP를 위조하여 보냄으로써 공격자를 속일 수 있고, DoS/DDoS 공격의 효과를 높이게 된다. IP 스푸핑을 통하여 공격 대상 호스트 시스템이나 라우터로 하여금 공격자의 패킷이 인증된 호스트 혹은 네트워크로부터 온 것처럼 하여 방화벽을 무사히 통과하거나 필터링되지 않고 공격을 할 수 있게 되는 것이다.

두 가지 대표적인 공격 유형 중 배포된 DoS/DDoS 공격 툴에서 많이 사용되는 Flooding 공격 유형에 대해 자세히 살펴보면 다음과 같다.

2.2.1. GET flooding 공격 기법

GET flooding은 좀비 PC가 IP를 변조하지 않고 3 way-handshake를 수행한 후 정상적인 트래픽과 유사하게 동일한 URL에 대한 반복요청을 통하여 웹 서버의 CPU 및 connection과 같은 리소스 고갈을 유도하거나, 웹 서버 내 DB를 호출하는 dynamic 콘텐츠에 대한 대량의 GET request를 보냄으로써 웹서버와 DB서버 간 부하를 유발하여 웹서비스를 중단시키는 형태의 공격으로, 소규모의 좀비로도 큰 효과를 낼 수 있어 공격자 입장에서 보면 매우 효 과적인 공격 기법이라고 할 수 있다. GET flooding 공격에는 패턴에 따라 크게 세 가지로 나눌 수 있다.

첫 번째는, 3 way-handshake로 연결 설정 후 GET request를 한번 하고 나서 RST나 4 way-handshake로 연결을 종료하는 과정을 반복하는 경우이다. 이러한 유형의 공격은 GET request 보다 3way-handshake로 세션을 설정 및 해제하는 과정에서 시스템의 부하를 발생시키고자 하는 목적이다. 그러므로 이러한 유형의 공격은 세션 설정과 종료 과정에서 3 way-handshake가 발생하므로 SYN flooding으로 탐지될 수도 있다.

두 번째는, 3 way-handshake로 연결 설정 후 GET request를 반복적으로 수행하는 경우이다. 이는 지속적인 GET request를 통하여 웹서버의 부하 유발이 주목적이라 할 수 있다. 따라서 GET flooding 공격에서 자주 발견되는 공격방식이기도 하다. 특히 최근에 나타나고 있는 application-level DDoS 공격을 보면 GET request 시 dynamic 콘텐츠에 대한 URI가 주를 이루는데, 이는 dynamic 콘텐츠가 웹서버와 database가 연동하는 과정에서 서버부하와 database 응답지연을 유발할 수 있으므로 static 콘텐츠에 대한 request보다 더 효과적이기 때문이다.

세 번째는, 3 way-handshake로 연결 설정 후 GET request도 하지 않고 4 way-handshake로 세션을 끊지도 않은 상태에서 반복적으로 새로운 3 way-handshake를 맺는 것이다. 이런 경우 웹서버는 client로부터 GET request를 기다리다가 timeout이 될때까지 대기하게 되므로 정상적인 요청을 처리하지 못하도록 하는 형태의 공격이라고 할 수 있다.

2.3 DDoS 탐지

일반적으로 DDoS는 패킷을 이용하여 탐지하고 하드웨어 차원에서 차단하지만, WAD 프로그램은 애플리케이션 계층에서 DDoS 공격을 탐지한다. 한 논문 '리눅스 아파치 웹 서버 실시간 로그 분석을 통한 공격 탐지 프로그램 개발'에서는 패킷 카운팅 대신 접근 로그량을 이용하여 DDoS를 탐지하였다.

해당 논문의 WAD의 DDoS 탐지 알고리즘은 그림9와 같다. 메인 함수에서 DDoS

를 탐지하는 스레드를 생성한다. 이 스레드는 단위 시간(본 연구에서는 5초로 정하였다)마다 DDoS를 검사한다. 스레드에서는 평균 로그량과 단위 시간당 접근 로그량의 중간값을 계산했다. '평균 로그량과 단위 시간당 접근 로그량의 차'의 절댓값이 허용 임계치보다 크면 DDoS로 판별한다. 이때 허용 임계치는 평균 로그량과 중간값을 이용하여 구하는데 일정 기간 학습된 결과로 하였다.

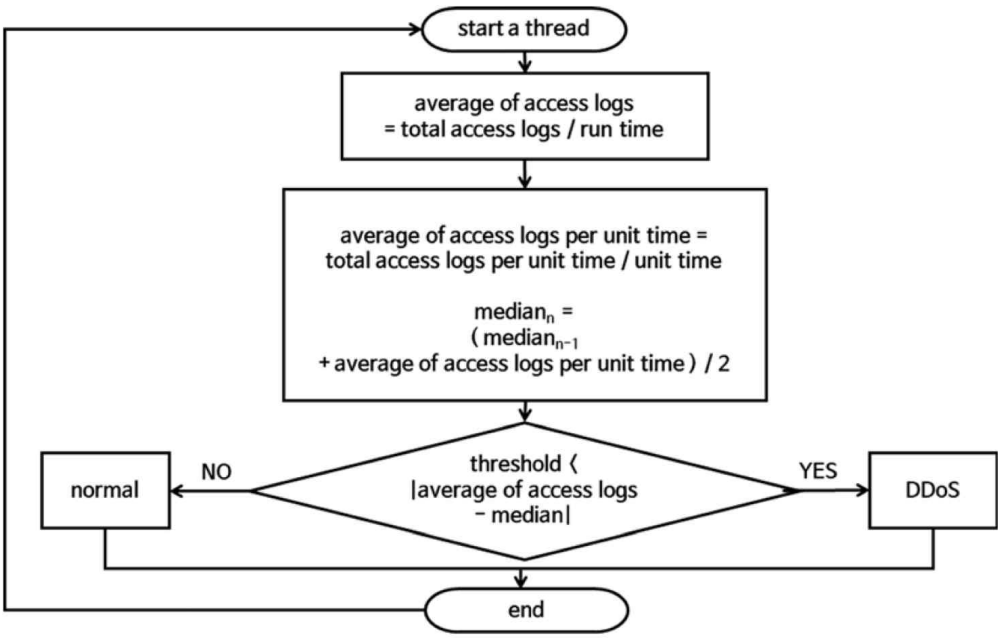


그림 9 DDoS 탐지 알고리즘

Fig. 9 The algorithm of DDoS detection

Ⅲ. 개발 내용

3.1 GET Flooding 공격 탐지 모델

제안하는 GET Flooding 공격 탐지 기법은 웹 서버로 전달되는 접근 로그량을 기반으로 비정상적으로 접근 로그량이 많은 악의적 사용자의 이상 행위를 탐지하는 방법이다.

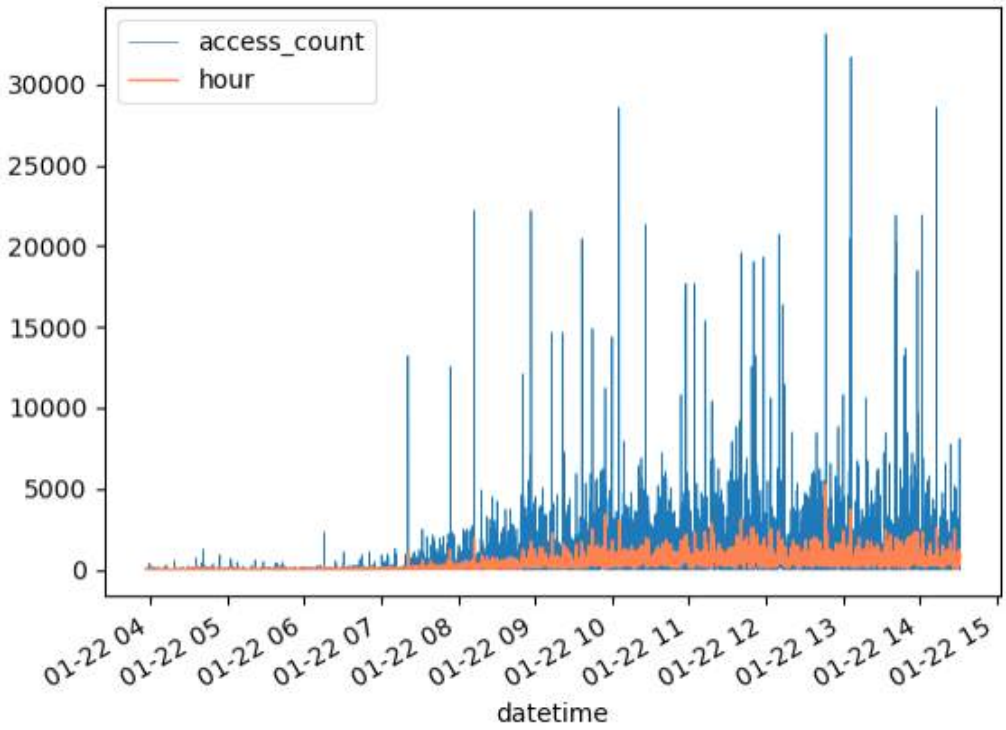
접근 시간에 대한 정보를 포함한 시계열 정보를 반영하여 LSTM 모델을 활용한 비지도 학습을 통해 악의적 사용자를 탐지했다. 여기서 정상적인 트래픽과 유사하게 동일한 URL에 대한 반복 요청을 통하여 단시간 내에 웹 서버의 리소스 고갈을 유도하는 GET Flooding 공격의 특징을 반영하여 접근 로그에 대한 가중치를 부여했다. 동일 IP 주소가 동일한 시간에 접근한 데이터의 개수에 따라 해당 로그에 가중치를 부여하였으며, 동일 시간에 접근한 로그량이 많을수록 큰 가중치를 부여했다.

3.2 데이터 분석

3.2.1 시간 시각화

준비한 데이터의 비이상적인 접근으로 판단되는 데이터량을 확인하기 위해 데이터를 선그래프로 시각화하여 나타냈다. 하지만 데이터가 초단위로 기록이 되어있기 때문에 선그래프로 표현하기에 데이터의 양이 너무 많아 전체적인 데이터의 추세를 알아보기 어려웠다. 그래서 데이터에 추세선을 같이 삽입하여 선 그래프로 나타냈다. 추세선은 이동평균(Moving average) 방법을 사용하여 데이터의 연속적인 그룹의 평균을 구하여 나타내지는 데이터의 전체적 흐름을 나타내는 선이다.

출력된 결과를 살펴보면 데이터는 1월 22일 4시부터 대략 14시까지의 짧은 시간동안 수집된 로그 데이터로 짧은 시간동안 기록된 데이터임을 알 수 있다. 8시 이전에는 접속자의 수가 적어 발생된 로그 데이터의 양이 적지만, 8시 이후부터 단위시간당 출력되는 로그 데이터량이 증가한 모습을 확인할 수 있다. 전체적으로 단위시간당 데이터가 1만개 이하로 출력되지만 중간중간 비이상적으로 많은 데이터가 출력되는 것도 확인이 가능하다.



또한 단위 시간을 1초로 설정하여 초당 접근 데이터량도 살펴보았다. 대부분의 IP의 초당 데이터량은 0에서 20회 사이로 적은 접근 로그를 발생시키고 있다. 이러한 적은 접근량 사이에서 초당 100회 이상의 접근을 시도하는 IP도 다수 발견되었다. 크기가 작은 서버에서 초당 100회 이상의 접근이 발생하는 경우는 비이상적인 접근으로 판단할 수 있다고 생각했다.

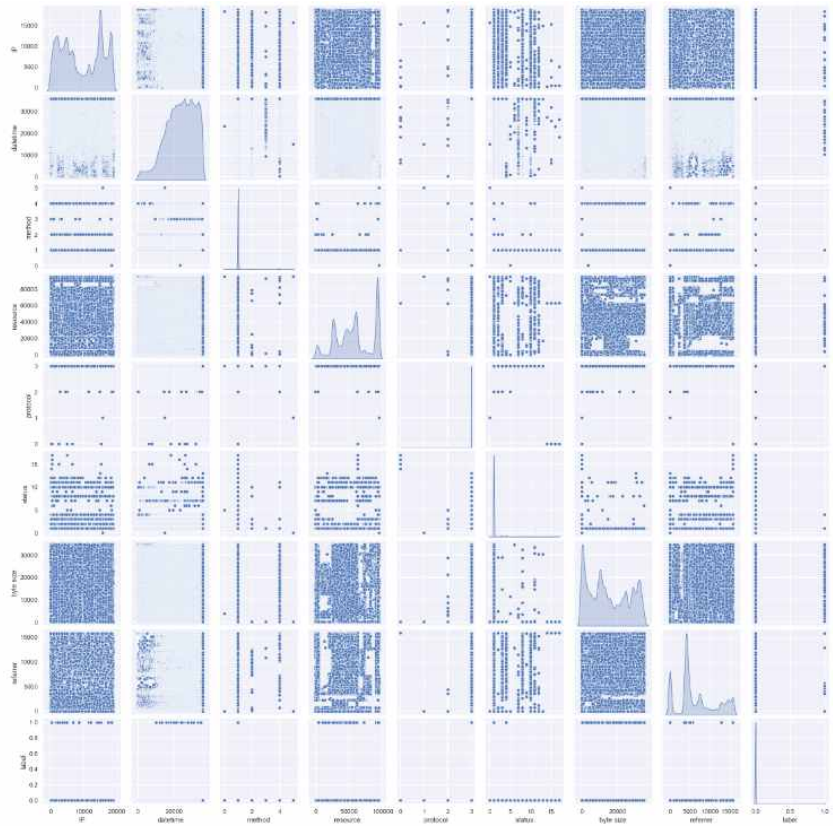
```
Time Range: 2019-01-22 07:20:53 , IP: 31.47.54.211, Log Count: 112
Time Range: 2019-01-22 07:53:52 , IP: 31.47.38.22, Log Count: 109
Time Range: 2019-01-22 08:12:26 , IP: 5.160.103.154, Log Count: 132
Time Range: 2019-01-22 08:50:22 , IP: 217.66.204.130, Log Count: 101
Time Range: 2019-01-22 08:56:50 , IP: 185.189.123.251, Log Count: 129
Time Range: 2019-01-22 09:13:10 , IP: 212.80.12.74, Log Count: 107
Time Range: 2019-01-22 09:21:50 , IP: 217.66.204.130, Log Count: 107
Time Range: 2019-01-22 09:36:33 , IP: 188.229.12.236, Log Count: 130
Time Range: 2019-01-22 09:44:47 , IP: 46.143.249.87, Log Count: 111
Time Range: 2019-01-22 10:00:09 , IP: 178.252.166.14, Log Count: 104
```

3.2.2 상관성 분석

IP	datetime	method	resource	protocol	status	byte size	referrer
31.56.96.51	2019-01-22 03:56:16	"GET	/image/60844/productModel/200x200	HTTP/1.1"	200	5667	"https://www.zanbil.ir/m/filter/b113"
40.77.167.129	2019-01-22 03:56:17	"GET	/image/14925/productModel/100x100	HTTP/1.1"	200	1696	"-"
91.99.72.15	2019-01-22 03:56:17	"GET	/product/31893/62100/%D8%B3%D8%B4%D9%88%D8%A7%...	HTTP/1.1"	200	41483	"-"
66.249.66.194	2019-01-22 03:56:18	"GET	/filter/b41,b665,c150%7C%D8%A8%D8%AE%D8%A7%D8%...	HTTP/1.1"	200	34277	"-"
40.77.167.129	2019-01-22 03:56:18	"GET	/image/57710/productModel/100x100	HTTP/1.1"	200	1695	"-"

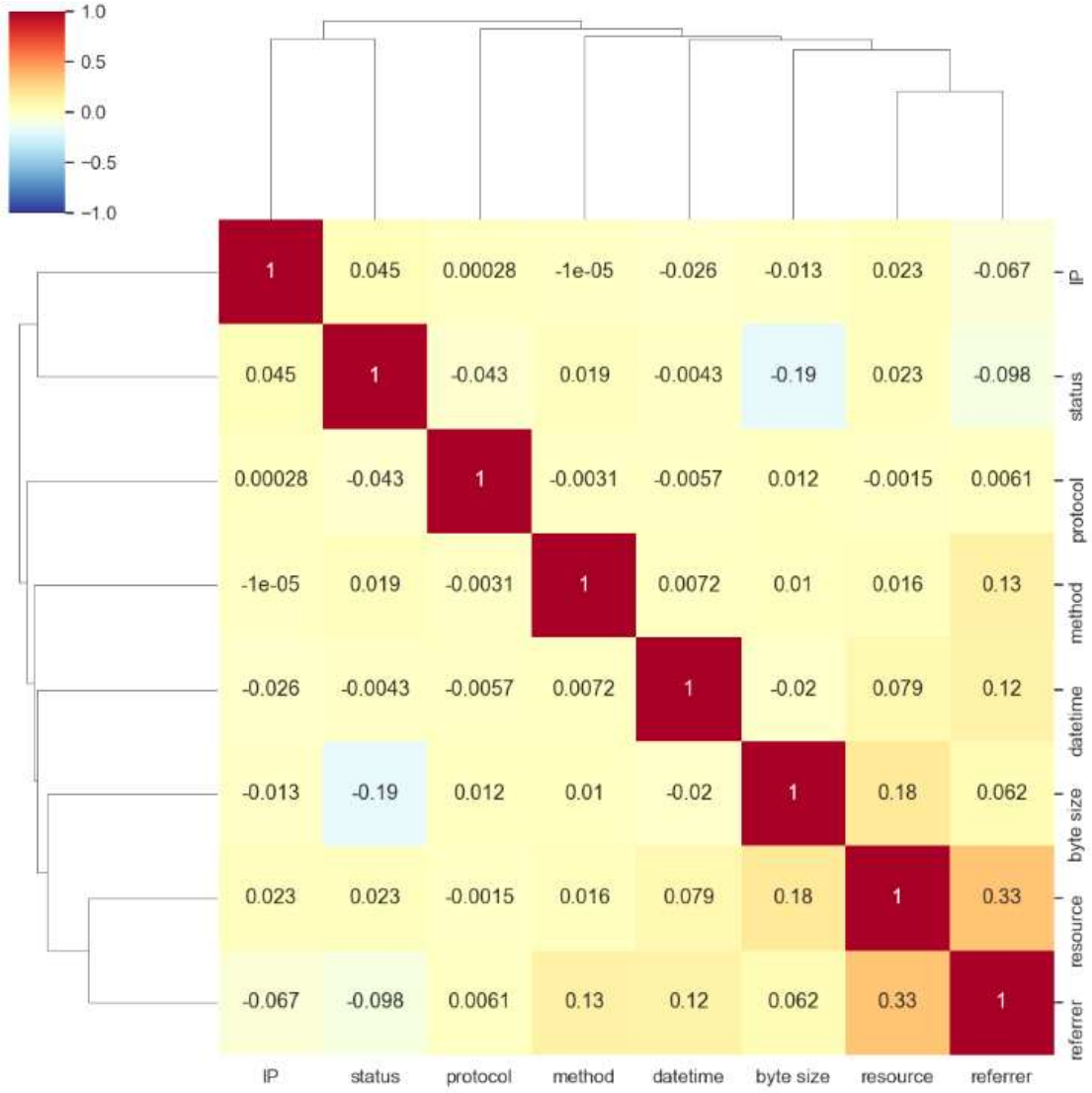
준비된 데이터셋은 총 8개의 칼럼으로 IP, datetime, method, resource, protocol, status, byte size, referer으로 기록돼 있다. byte size를 제외한 모든 칼럼은 명목 척도로 명목형 데이터로 변환이 가능하다. 칼럼의 type을 모두 숫자형 변수로 변환이 가능하기 때문에, 공분산과 상관계수를 분석하기에 적합하다.

간단한 데이터 전처리를 통해 산점도 행렬을 그려서 전체 변수 간의 관계가 어떤지 한눈에 확인해보았다. 결과를 살펴보면 전반적으로 데이터 사이의 연관성이 매우 떨어지는 것을 확인할 수 있다.



변수의 개수가 많아 산점도 행렬로 변수 사이의 관계를 분석하기에 가독성이 떨어진다고 판단하여 피어슨 상관계수를 통한 상관 분석을 통해 상관성을 분석해보았다. 히트맵을 통해 시각화한 상관계수는 아래 사진과 같다.

노란색에 가까울수록 양의 상관관계를 보이고 짙은 보라색에 가까울수록 음의 상관관계를 보인다. 동일한 변수 간에는 상관계수가 1로 나오고 있고, 나머지의 값들은 -1에서 1사이의 값을 통해 상관관계가 표현되고 있다. 전체적으로 대부분의 변수들 사이에 상관관계는 낮은 형태로 보여진다. resource 변수와 referrer 변수 사이에 0.33의 양의 상관관계를 나타내는 것을 확인할 수 있지만, 본 작품에서 찾고자하는 서비스에 비이상적으로 높은 접근을 탐지하는데에 도움이 되는 상관관계로 보여지지는 않는다.



3.2 데이터 전처리

데이터 탐색을 통해 확인한 결측값과 이상값을 제거하여 최종적으로 활용하게 된 데이터셋의 정보는 아래와 같다. 총 50만 개의 데이터 세트를 준비하였다.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 526115 entries, 3 to 1048537
Data columns (total 9 columns):
 #   Column     Non-Null Count   Dtype
---  ------     --------------   -----
 0   IP         526115 non-null  object
 1   datetime   526115 non-null  object
 2   method     526115 non-null  object
 3   resource   526115 non-null  object
 4   protocol   526115 non-null  object
 5   status     526115 non-null  object
 6   byte size  526115 non-null  object
 7   referrer   526115 non-null  object
 8   label      526115 non-null  bool
dtypes: bool(1), object(8)
memory usage: 36.6+ MB
```

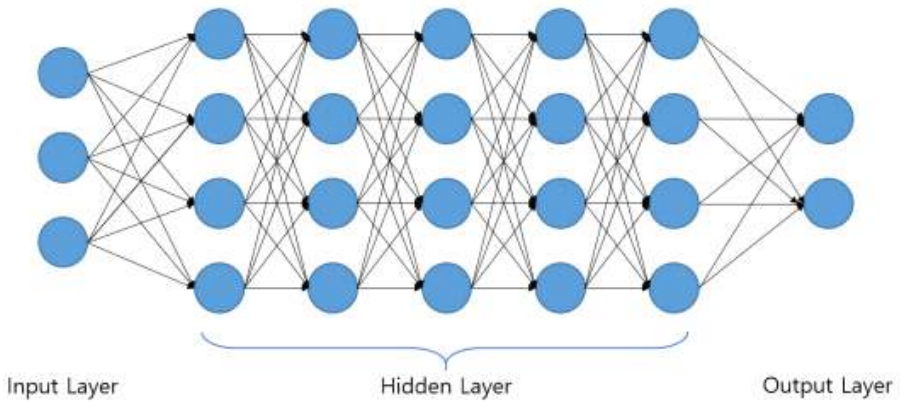
비지도 학습의 정확도를 높이기 위해 object 형태의 데이터 필드를 모두 LabelEncoder를 활용하여 수치화하였다. 이후 IP별 접속 빈도를 그룹별로 분류하여, 그룹의 크기별로 다른 가중치를 부여하였다. 여기서 접속 빈도가 높은 그룹에는 1에 가까운 가중치를, 접속 빈도가 낮은 그룹은 0에 가까운 가중치를 부여했다.

	IP	datetime	method	resource	protocol	status	byte size	referrer	label	weight
0	5306	0	1	53863	3	1	30320	7845	False	0.005814
6	6303	1	1	50007	3	1	2757	0	False	0.017442
8	6303	1	1	50137	3	1	4029	0	False	0.017442
9	4594	1	1	74092	3	1	23213	0	False	0.017442
12	6303	2	1	55891	3	1	4987	0	False	0.023256
...	...	...	...	...	...	...	...	...	...	...
1048530	4996	35820	1	91754	3	10	2	3829	False	0.127907
1048533	18335	35821	1	88169	3	1	2558	12772	False	0.168605
1048534	1368	35821	1	76918	3	1	24426	3853	False	0.168605
1048535	10969	35821	1	60256	3	1	15797	7473	False	0.168605
1048537	1020	35821	1	30436	3	1	11514	4304	False	0.168605

526115 rows × 10 columns

3.4 LSTM 모델

LSTM은 순환신경망(RNN)의 장기의존성에 대한 문제의 해결책으로 등장한 신경망 모델이다. LSTM도 RNN처럼 연속된 체인과 같은 구조로 되어 있으나 하나의 모듈에 4개의 층이 서로 정보를 주고받도록 설계되어있다.



시계열 데이터인 로그 데이터를 분석하기에 이러한 LSTM 모델이 적절하다고 판단했다. 비정상적인 접근 로그량을 예측하기 위해 LSTM 레이어 2개, Dropout 레이어 2개, Dense 레이어 1개로 구성된 네트워크 모델을 사용했다.

LSTM 레이어의 활성함수(Activation function)은 relu로 구성하였으며, Dense 레이어의 활성함수는 sigmoi ㅇfh 설정하였다. 또한 네트워크의 갱신을 위한 손실함수는 binary_crossentropy로 설정하였으며, 최소화하는 방향으로는 가중치를 결정하는 여러 방법 중에서 시계열 예측에서 주로 사용되어지는 Adam optimizer를 사용하여였다. 2개의 LSTM 레이어 사이에 과대적합(Overfitting)을 방지하기 위해 Dropout을 적용했다. Dorpout이 적용되는레이어는 학습을 하는 동안 무작위로 해당 층의 출력이 제한되며, 네트워크의 연속성이 단절되어서 일부 뉴런들의 부적합한 연결을 방지하여 과대적합을 감소시킨다.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 1, 16)             1664

 dropout (Dropout)           (None, 1, 16)             0

 lstm_1 (LSTM)               (None, 16)                2112

 dropout_1 (Dropout)         (None, 16)                0

 dense (Dense)               (None, 1)                 17

=================================================================
Total params: 3,793
Trainable params: 3,793
Non-trainable params: 0
_________________________________________________________________
```

Ⅳ. 개발 결과

5.1 모델 학습 결과

접근 빈도에 따라 가중치를 부여한 비지도 학습을 통한 GET Flooding 공격과 유사한 비정상적으로 높은 접근에 대한 이상 탐지를 진행한 결과를 4가지의 성능 지표로 확인한 결과는 아래와 같다.

성능 지표	train	test
정확도(Accuracy)	0.9991	0.9992
정밀도(Precision)	0.8957	0.9062
재현율(Recall)	0.9954	0.9886
F1 Score	0.9429	0.9456

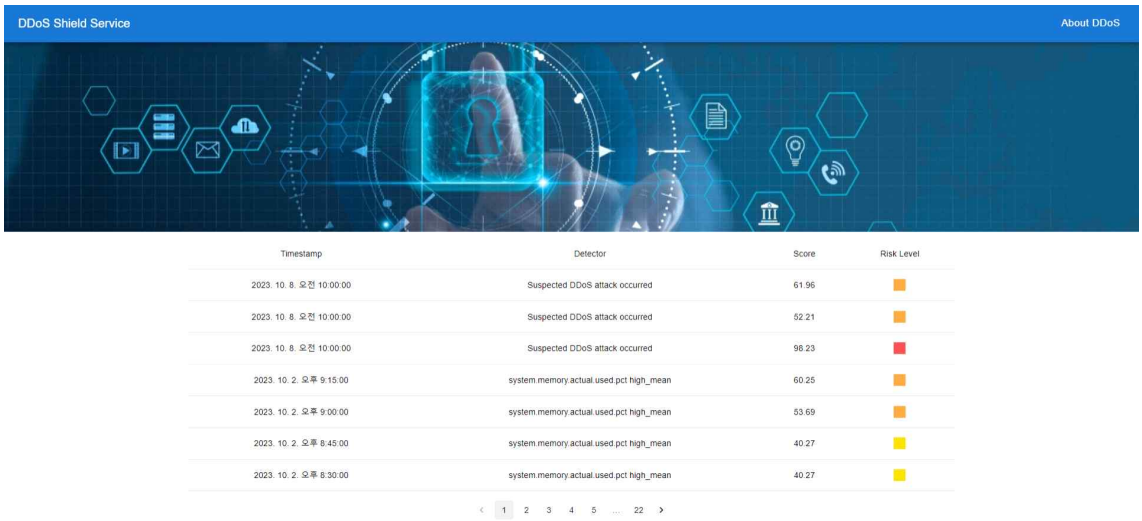
학습 데이터 세트와 테스트 데이터 세트로 확인한 결과는 대부분의 값이 90% 이상의 좋은 결과로 나온 만큼 성공적인 학습이 이루어졌음을 확인할 수 있었다. 이러한 결과를 통해 웹 액세스 로그의 시간당 접근 빈도에 따라 데이터에 가중치를 부여한 행위가 의미 있는 행위임을 확인할 수 있었다. 각각의 성능 지표에 대해 간단하게 설명하고 넘어가겠다.

(1) 정확도 : 전체 예측 중 올바르게 예측한 비율
(2) 정밀도 : 양성으로 예측한 값 중 실제로 양성인 값의 비율
(3) 재현율 : 실제 양성인 값 중에서 양성으로 예측한 값의 비율
(4) F1 Score : 정밀도와 재현율의 조화평균

5.2 이상 탐지 알림 서비스

개발한 모델이 제대로 작동하는지 확인을 위한 DDoS 이상 탐지 알림 서비스를 추가적으로 개발하였다. 해당 서비스는 웹 서버로 전달되는 접근 로그량을 기반으로 비정상적으로 접근 로그량이 많은 DDoS 공격을 수행하는 악의적 사용자의 이상 행위를 탐지하고 이를 실시간으로 알리는 목적을 갖는다.

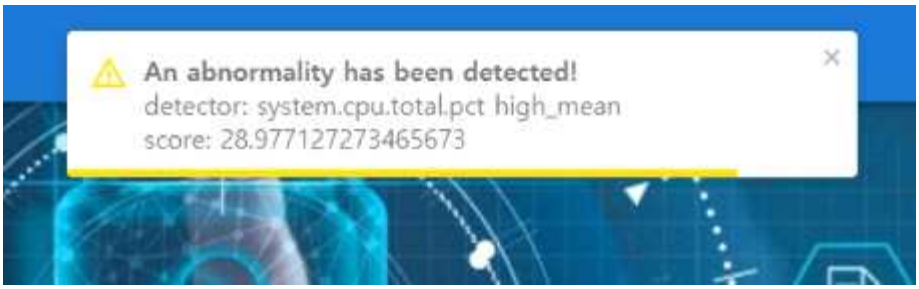
해당 서비스는 이상 탐지가 발생한 시간 정보, 이상 탐지 명, 탐지된 이상의 위험도 및 위험 레벨을 테이블의 형태로 제공한다. 위험도 및 위험 레벨은 로그 데이터의 접속 빈도에 따라 주어진 가중치를 기반으로 4계의 레벨로 위험도를 분류하였다. 위험 점수가 0점에서 25점 사이인 경우 minimul 단계(파란색), 25점에서 50점 사이인 경우 low 단계(노란색), 50점에서 70점 사이인 경우 high 단계(주황색), 마지막으로 70점에서 100점 사이인 경우 risk 단계(붉은색)로 분류하였다.



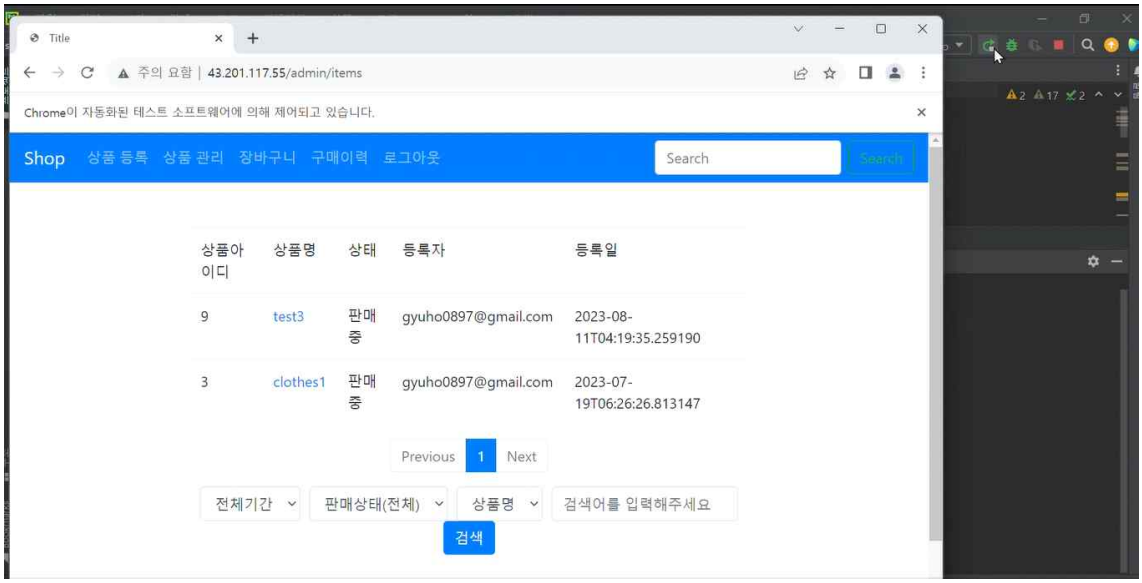
< 웹 서비스 실제 화면 >

5.3 서버 부하 유발

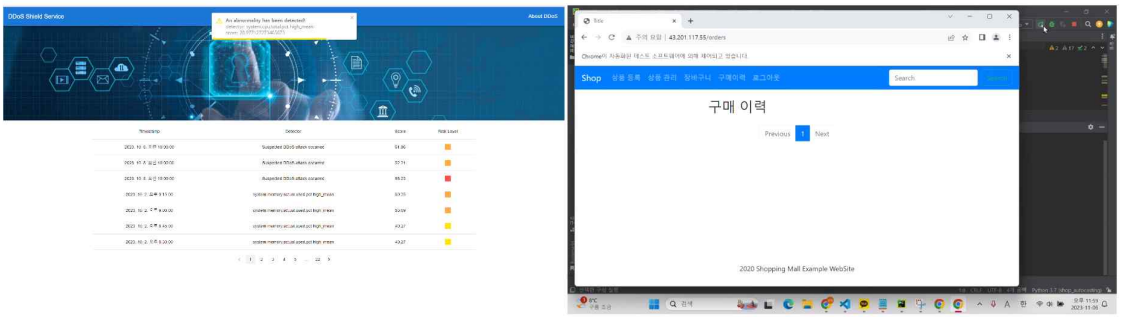
이상 탐지 알림 서비스를 작동시키기 위해 스프링부트를 이용하여 간단한 쇼핑몰 페이지를 만들어 해당 페이지에 직접 부하를 두어 접근량을 증가시켰다. 이상 탐지 알림 서비스에 적용된 개발 모델은 쇼핑몰 웹 페이지의 로그 데이터를 관찰하다가 서버 부하를 통해 발생된 급격히 증가하는 비이상적인 접근량을 탐지하여 사용자에게 이상 알림을 발생시킨다.



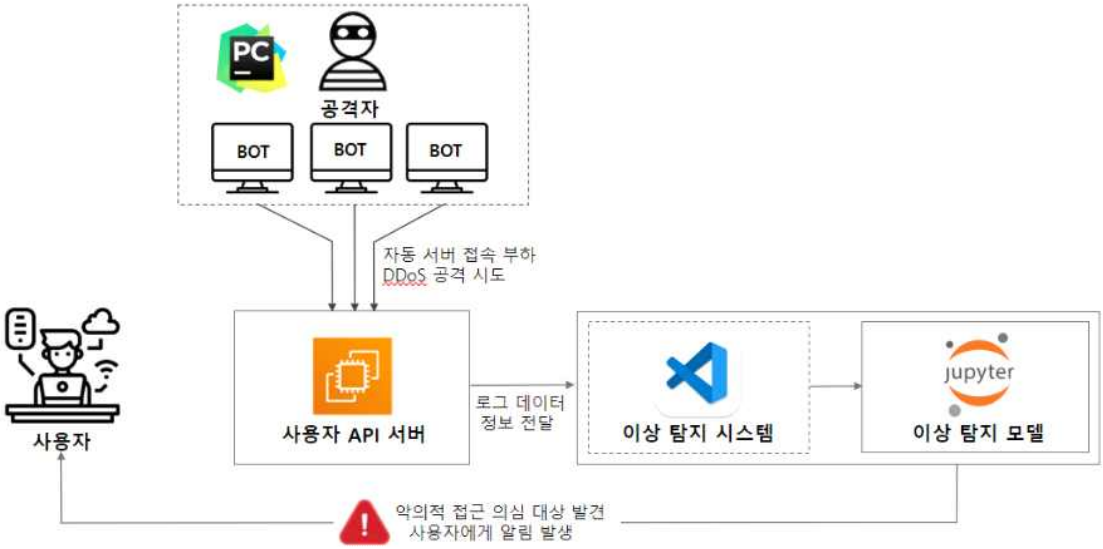
< 이상 탐지 알림 실제 화면>



< 부하 시스템 실행 과정 >



< 서버 부하 두기 및 시스템 실시간 알림 서비스 >


공격자
BOT
BOT
BOT
자동 서버 접속 부하
DDoS 공격 시도
사용자
사용자 API 서버
로그 데이터
정보 전달
이상 탐지 시스템
jupyter
이상 탐지 모델
악의적 접근 의심 대상 발견
사용자에게 알림 발생

< 전체 서비스 구성도 >

Ⅴ. 결론

1. 참고자료

[1] 나현정. (2003, December). 데이터 마이닝을 이용한 DDoS 공격 탐지 기법.
[2] 김진 & 오창석. (2013, May 8). 이상 접근 분석을 이용한 GET Flooding DDoS 공격 탐지.
[3] 박재연, 이송연, 이하은, & 이종우. (2017, December 14). 리눅스 아파치 웹 서버 실시간 로그 분석을 통한 공격 탐지 프로그램 개발.
[4] 유대성 & 오창석. (2004, November 17). 공격 탐지를 위한 트래픽 수집 및 분석 알고리즘.