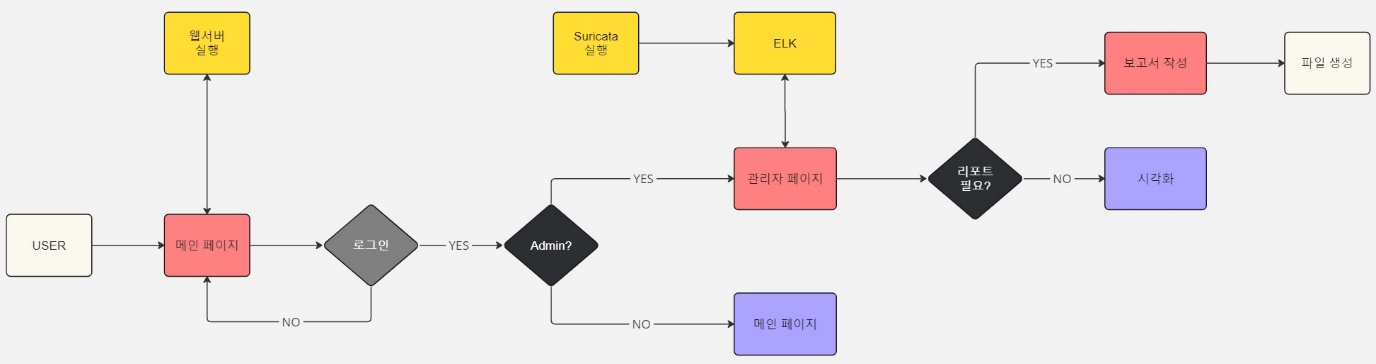
1. Flow Chart



2. 요구사항 명세서 (기능적/비 기능적)

A. 기능적

|  |  |  |  |
| --- | --- | --- | --- |
| **요구사항번호** | 기능-001 | **요구사항 명** | **Suricata 운영** |
| **요구사항설명** | 1. 방화벽 뒤에 Suricata를 설치하고 운영한다.  2. Suricata를 통해 네트워크 로그를 수집한다.  3. Rule을 설정하고 IDS 모드를 실행하여 관련 이벤트 로그를 수집한다. | | |
| **관련요구사항** | Suricata의 IDS 모드를 활용하여 침입 탐지를 우선한다. | | |
| **전제 조건** | 방화벽이 운영중이어야 한다. | **우선순위** | 2 |

|  |  |  |  |
| --- | --- | --- | --- |
| **요구사항번호** | 기능-002 | **요구사항 명** | **Elastic Stack 운영** |
| **요구사항설명** | 1. Suricata와 연동하여 Elastic Stack을 설치하고 운영한다.  2. 수집된 로그 데이터를 Elastic Stack을 활용하여 시각화한다.  3. Elastic Stack을 활용하여 필요한 데이터를 검색하고 분석한다. | | |
| **관련요구사항** | Nginx와 연동하여 웹서버에 Elastic Stack를 구현한다. | | |
| **전제 조건** | 웹서버가 운영중이어야 한다. | **우선순위** | 1 |

|  |  |  |  |
| --- | --- | --- | --- |
| **요구사항번호** | 기능-003 | **요구사항 명** | **Report 작성** |
| **요구사항설명** | 1. Elastic Stack을 활용하여 로그 관련 리포트를 작성한다.  2. 필요한 사항이 포함된 리포트 작성을 자동화 한다.  3. 웹페이지에 요약된 리포트 형식을 구현한다.  4. Elastic Stack의 시각화 데이터를 활용하여 리포트 작성에 추가한다. | | |
| **관련요구사항** | Elastic Stack을 활용한 지속적인 관제를 운영한다. | | |
| **전제 조건** | Elastic Stack 서비스를 제공받고 있다. | **우선순위** | 1 |

B. 비기능적

|  |  |  |  |
| --- | --- | --- | --- |
| **요구사항번호** | 비기능-001 | **요구사항 명** | **작동 환경** |
| **요구사항설명** | 1. Django WAS를 통하여 웹서버를 운영한다.  3. 로그 빅데이터 관제에 Elastic Stack을 이용한다.  2. 해당 플랫폼은 네이버 클라우드 Ubuntu 22.04 64bit에서 작동한다.  4. 해당 플랫폼은 Python 기반으로 하여 개발한다. | | |
| **관련요구사항** |  | | |
| **전제 조건** |  | **우선순위** |  |

3. WBS (Work Breakdown Structure)

|  |  |
| --- | --- |
| **케이스 ID : report-001 웹서버 운영** | |
| **사전조건** | 사용자가 운영중인 웹서비스에 접속한다. |
| 시나리오 | |
| **[주요 이벤트 경로]**  1. 사용자가 제공 중인 웹서비스에 접근한다.  2. 사용자가 원하는 뉴스 기사를 클릭한다.  3. 상세 페이지로 이동하여 관련 기사의 자세한 내용을 제공한다.  4. 사용자가 원하는 뉴스 기사를 검색한다.  5. 검색 결과 페이지로 이동하여 관련 검색 내용을 표시한다.  6. 사용자가 원하는 내용을 클릭하면 관련 상세 페이지로 이동한다.  **[대안 또는 예외 경로]**  1. 로그인을 시도할 경우에 로그인 페이지로 이동한다.  2. DB에 저장된 관리자 계정과 비교하여 로그인 성공/실패 여부를 제공한다. | |
| 사후조건 | 주기적인 업데이트를 통하여 사용자에게 새로운 기사를 제공한다.  검색을 통한 오래된 기사 검색 기능도 제공한다. |

|  |  |
| --- | --- |
| **케이스 ID : report-002 웹서버 로그인** | |
| **사전조건** | 사용자가 운영중인 웹서비스에 로그인을 시도한다. |
| 시나리오 | |
| **[주요 이벤트 경로]**  1. 사용자가 제공 중인 웹서비스 메인 페이지에서 로그인을 클릭한다.  2. 로그인 페이지로 이동한다.  3. 로그인 ID와 Password를 입력하고 로그인을 시도한다.  4. DB에 저장된 관리자 계정과 비교하여 로그인 성공/실패 여부를 제공한다.  A. 로그인 성공 시 관리자 메뉴가 제공되는 메인 페이지로 접속한다.  B. 로그인 실패 시 다음과 같은 메시지를 알려주고 사용자 메인 페이지로 이동한다.  i. 알림 메시지 : “관리자 계정을 입력하세요.”  **[대안 또는 예외 경로]**  1. 메인 페이지에서만 관리자 로그인 접속을 활성화 한다.  2. 차후 제공 예정인 서비스 등을 위하여 사용자 로그인/로그아웃 기능을 구현한다. | |
| 사후조건 | 관리자 메뉴를 통하여 뉴스 기사 올리기, 수정, 삭제가 가능하도록 한다.  관리자 메뉴를 통하여 웹서버 로그 데이터 확인과 리포트 작성이 가능하도록 연결한다. |

|  |  |
| --- | --- |
| **케이스 ID : report-003 Suricata 설치 및 운영** | |
| **사전조건** | 관리자가 Suricata를 실행한다. |
| 시나리오 | |
| **[주요 이벤트 경로]**  1. 웹서버에 Suricata를 설치한다.  2. Suricata의 기본 Rule을 Update 한다.  3. Suricata에 필요한 Custom Rules을 적용한다.  4. Suricata를 IDS 모드로 운영한다.  5. Suricata의 최근 로그를 지속적으로 모니터링 한다.  6. Suricata의 전체 로그 데이터를 수집, 분석한다.  **[대안 또는 예외 경로]**  1. 침입 방지가 필요할 경우 IPS 모드로 Suricata를 운영한다. | |
| 사후조건 | 지속적인 Log 모니터링을 위하여 항상 Suricata 운영을 유지한다.  주기적으로 빅데이터 Log 파일을 DB화 한다. |

|  |  |
| --- | --- |
| **케이스 ID : report-004 Elastic Stack 설치 및 운영** | |
| **사전조건** | 관리자가 Elastic Stack을 실행한다. |
| 시나리오 | |
| **[주요 이벤트 경로]**  1. 웹서버에 순서대로 Elasticsearch, Kibana, Logstash, Filebeat를 설치한다.  2. Filebeat 와 Suricata를 연동한다.  3. Elastic Stack에 설정한 ID/Password로 접속한다.  4. filebeat filter를 선택한다.  5. 실시간으로 Suricata가 수집한 Log 시각화 데이터를 확인한다.  6. Elastic을 활용하여 Log 데이터를 검색, 분석한다.  **[대안 또는 예외 경로]**  1. Kibana 기능을 nginx와 연동하여 접속 가능하게 한다.  2. Elastic Stack이 사용하는 Port들을 관리한다. | |
| 사후조건 | 리포트 작성을 위하여 대시보드 시각화 및 관련 통계 데이터를 공유받는다.  중요한 이벤트 Log는 DB화하여 Elasticsearch를 활용한다. |

|  |  |
| --- | --- |
| **케이스 ID : report-005 웹서버 로그 시각화** | |
| **사전조건** | 관리자가 Elastic Stack을 운영한다. |
| 시나리오 | |
| **[주요 이벤트 경로]**  1. Elastic Stack에 로그인하여 접속한다.  2. Filebeat로 받아온 Suricata Log 데이터를 확인한다.  3. 설정한 시각화 필드를 통해 Log 데이터를 분석, 시각화 한다.  **[대안 또는 예외 경로]**  1. 로그인한 관리자에 한하여 분석한 Log 데이터와 시각화 내용을 제공한다.  2. 필요한 필드를 반영하여 시각화 내용을 구성한다. | |
| 사후조건 | Suricata와 연동하여 관련 Log 데이터를 항상 모니터링 한다.  리포트 작성을 위하여 필요한 시각화 필드와 관련 내용을 공유한다. |

|  |  |
| --- | --- |
| **케이스 ID : report-006 로그 데이터 리포트 자동화** | |
| **사전조건** | 관리자가 Elastic Stack을 활용하여 보고서를 작성한다. |
| 시나리오 | |
| **[주요 이벤트 경로]**  1. 관리자가 Elastic Stack에 접속한다.  2. 실시간 수집되고 있는 Log 데이터를 확인하고 리포트 작성을 실행한다.  3. 웹페이지를 통해 자동으로 작성된 보고서 내용을 확인한다.  4. pdf 형식으로 저장을 원하는 경우 ‘리포트 저장’을 클릭한다.  5. 필요한 데이터가 포함된 다양한 보고서를 작성한다.  **[대안 또는 예외 경로]**  1. 로그인한 관리자 정보를 보고서 작성 시 포함한다.  2. Elastic Stack에서 수집, 분석된 구체적인 데이터도 제공 가능하다. | |
| 사후조건 | 필요한 리포트 작성을 위하여 지속적으로 Elastic Stack을 운영한다.  작성된 리포트 자료를 DB화하여 관리한다. |

4. 기능 / 기술 명세서





|  |  |
| --- | --- |
| 기능구현 ID | R001-001 |
| 목적 | 웹서버 구성 |
| - 네이버 클라우드를 사용하여 웹서버를 생성하고 설정함으로써, 웹 애플리케이션이 동작할 수 있는 서버 환경을 구축  > 네이버 클라우드 로그인 > ‘콘솔’  > 메뉴 - Services - Server 선택 - '+서버 생성'  - 서버 이미지 : ubuntu-22.04 / KVM  - 서버 설정  VPC : lab1-vpc (서버가 위치할 가상 네트워크 선택)  Subnet : lab1-vpc-web-subnet~ / Public (서버의 IP 주소 범위를 정의하는 서브넷 선택, Public 서브넷을 통해 공인 IP 할당. 공인 IP 필요시)  서버 스펙 : Standard / s2-g3(CPU 2개, 메모리 8GB)  요금제 : 월요금제  서버 개수 : default  서버 이름 : was001 / was002  Network Interface : ip 10.0.1.101/102  공인 IP : 새로운 공인 IP 할당  나머지 : default  - 스토리지 설정 : default  - 인증키 설정 : 새로운 인증키 생성 > ncp + 오늘날짜 (서버에 접근할 수 있는 보안 인증키를 새롭게 생성)  - 네트워크 접근 설정 : ACG > lab1-web-acg 선택 (서버에 접근할 수 있는 네트워크 규칙 설정)  - 확인 후 '서버 생성'  서버 생성 후 SSH를 통해 서버에 접속  -서버 생성 완료 후  > $ ssh root@223.130.147.99  > PW : root 비밀번호 입력  -서버 초기 설정  서버 업데이트 및 업그레이드  # apt update  # apt upgrade  >서버의 안정성과 보안성을 강화  NginX 설치  > NginX 웹서버 설치, 자동 실행 설정, IP 버전 설정 수정  # apt install -y nginx  # systemctl start nginx  # systemctl enable nginx  # systemctl status nginx  NginX의 기본 설정 파일 수정  > IPv4만 사용  # vi /etc/nginx/site-enabled/default : 23번째 줄 listen [::] 80~ 주석 처리  MariaDB 설치  > 데이터베이스 서버 설정 및 초기화 작업  # apt install mariadb-server  # systemctl status maridb  # mysql -u root -p : DB 접속  > GRANT ALL PRIVILEGES ON \*.\* to root@'%' identified by '123!@#';  (# CRTEATE USER rroot@'%' identified by '123!@#';)  > flush privileges;  네트워크 설정  - MariaDB 서버에 외부에서 접근할 수 있도록 네트워크 설정을 조정  > 네이버 클라우드 : ACG > 'lab1-web-acg' ACG 설정 : Inbound / 프로토콜 TCP, 접근 소스 0.0.0.0/0, 허용 포트 3306 추가  MariaDB 설정 파일 수정  - MariaDB 네트워크 설정 변경  # vi /etc/mysql/mariadb.conf.d/50-server.cnf : bind-address 항목 주석 처리  # systemctl restart mariadb  $ 보안 설정  SSH 설정 보안 강화  # vi /etc/ssh/sshd\_config  PermitRootLogin no (root 계정 SSH 접속 금지)  PasswordAuthentication no (비밀번호 인증 비활성화)  SSH 포트 변경 (예: Port 2222)  # systemctl restart sshd  $ UFW 방화벽 설정  # apt install ufw  기본 정책: 들어오는 트래픽 차단, 나가는 트래픽 허용  필요한 포트만 허용: SSH (2222/tcp), HTTP (80/tcp), HTTPS (443/tcp)  # ufw enable (방화벽 활성화)  # ufw status (상태 확인)  $ SSL 인증서 설치  Certbot 설치 및 SSL 인증서 발급  # apt install certbot python3-certbot-nginx  SSL 인증서 발급: # certbot --nginx -d yourdomain.com -d www.yourdomain.com  인증서 자동 갱신 설정 확인: # certbot renew --dry-run  -ELK 설치 및 설정  Elasticsearch, Logstash, Kibana 설치  # apt install openjdk-11-jdk  # wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.x.x-amd64.deb  # dpkg -i elasticsearch-8.x.x-amd64.deb  # systemctl start elasticsearch  # systemctl enable elasticsearch  # wget https://artifacts.elastic.co/downloads/logstash/logstash-8.x.x.deb  # dpkg -i logstash-8.x.x.deb  # systemctl start logstash  # systemctl enable logstash  # wget https://artifacts.elastic.co/downloads/kibana/kibana-8.x.x-amd64.deb  # dpkg -i kibana-8.x.x-amd64.deb  # systemctl start kibana  # systemctl enable kibana  ELK 설정  Elasticsearch 설정 파일 수정: /etc/elasticsearch/elasticsearch.yml  Logstash 설정 파일 작성 및 필터 정의  Kibana 설정 파일 수정: /etc/kibana/kibana.yml  로그 수집, 분석, 시각화를 위한 기본 설정 및 테스트 수행 | |
| 부가설명 | 보안 설정은 반드시 활성화하여 운영해야 하며, Elasticsearch와 Kibana의 보안을 위해 사용자 인증 및 TLS/SSL 암호화를 설정하는 것이 중요함.  Prometheus와 Grafana는 서버 모니터링 및 시각화를 위한 도구로, ELK와 함께 사용하는 것이 권장됨.  UFW 방화벽은 기본적으로 들어오는 트래픽을 차단하며, 필요한 포트만 허용하여 보안을 강화할 수 있음. |

|  |  |
| --- | --- |
| 기능구현 ID | R001-002 |
| 목적 | WAS 구성 |
| **WAS(Web Application Server) 환경 구축**  python 설치  Python3와 pip(Python 패키지 관리자) 설치  # apt install python3  # apt install python3-pip  # pip3 install --upgrade pip  (> py -m pip install --upgrade pip)  Django(웹 애플리케이션을 쉽게 개발할 수 있도록 도와주는 프레임워크) 설치  # pip3 install Django  # apt install python3-django  Django 프로젝트 디렉토리 생성 및 이동  # mkdir django-pjt  # cd django-pjt  # django-admin startproject netreport\_pjt (netreport\_pjt 프로젝트 생성)  Django 설정 변경  ALLOWED\_HOSTS 설정 수정 (개발 서버를 외부에서 접근 가능하게)  # cd netreport\_pjt  # vi netreport\_pjt/setting.py  > 28번째 줄 ALLOWED\_HOSTS = ['\*'] 변경  Django 개발 서버 실행  -웹 애플리케이션을 테스트할 수 있도록 개발 서버를 실행  # python3 manage.py runserver 0.0.0.0:8000 (모든 IP에서 접근 가능하도록 설정)  웹 브라우저를 통해 Django 애플리케이션에 접근  > 223.130.147.99:8000/ 열기 | |
| 부가설명 | Django 개발 서버는 개발 및 테스트 용도로만 사용하며, 실제 운영 환경에서는 WSGI 서버(예: Gunicorn)와 웹 서버(예: Nginx)와 함께 배포하는 것이 좋음.  ALLOWED\_HOSTS 설정을 '\*'로 변경하면 모든 도메인에서 접근이 가능하므로, 실제 배포 시에는 도메인 또는 IP 주소를 명시하여 보안을 강화해야 함.  Python과 Django의 버전 호환성을 확인하여 설치 및 운영 중 문제를 방지할 수 있음. |

|  |  |
| --- | --- |
| 기능구현 ID | R001-003 |
| 목적 | 웹 서비스 구성 |
| Django 애플리케이션 생성 및 설정  # cd django-pjt/netreport\_pjt  Django 앱을 생성하여 웹 서비스 구조를 마련  - report\_main, report\_list, report\_admin 세 개의 앱을 생성하여 프로젝트를 기능별로 나눈다.  # python manage.py startapp report\_main  # python manage.py startapp report\_list  # python manage.py startapp report\_admin  Django 설정 파일 수정  INSTALLED\_APPS 및 DATABASES 설정 변경  **# vi netreport\_pjt/settings.py**  INSTALLED\_APPS 항목에 'report\_main', 'report\_list', 'report\_admin' 추가  INSTALLED\_APPS = [  **'report\_main',**  **'report\_list',**  **'report\_admin',**  'django\_extensions',  'django.contrib.admin',  'django.contrib.auth',  'django.contrib.contenttypes',  'django.contrib.sessions',  'django.contrib.messages',  'django.contrib.staticfiles',  ]  템플릿 설정  - templates와 static 디렉토리를 사용할 수 있도록 각각의 설정 추가  > DIRS 항목에 ‘templates’ 디렉토리 추가  TEMPLATES = [  {  'BACKEND': 'django.template.backends.django.DjangoTemplates',  'DIRS': **[BASE\_DIR / 'templates'**],  'APP\_DIRS': True,  ...  ...  정적 파일 디렉토리 설정  >STATICFILES\_DIRS 항목에 'static' 디렉토리 추가  STATIC\_URL = 'static/'  STATICFILES\_DIRS = [  **BASE\_DIR / 'static',**  ]  ...  ...  언어 및 시간대 설정  - 한국어와 서울 시간대로 설정  >LANGUAGE\_CODE와 TIME\_ZONE 수정  LANGUAGE\_CODE = **'ko-kr'** # 'en-us'  TIME\_ZONE = **'Asia/Seoul'** # 'UTC'  ...  ...  MySQL 데이터베이스 연결 설정  - Django 프로젝트를 MySQL 데이터베이스와 연결할 수 있도록 설정  DATABASES 항목에서 MySQL 데이터베이스 연결 설정  DATABASES = {  'default': {  # 'ENGINE': 'django.db.backends.sqlite3',  # 'NAME': BASE\_DIR / 'db.sqlite3',  **'ENGINE': 'django.db.backends.mysql',**  **'NAME': 'django\_netreport', # 연동할 mysql db 이름**  **'USER': 'rroot', # db 접속 계정명**  **'PASSWORD': 'rroot123!@#', # 해당 계정 비밀번호**  **'HOST': '223.130.147.99',**  **'PORT': '3306',**  }  }  ...  ...  각 앱의 URL 설정  >'report\_main', 'report\_list', 'report\_admin'의 URLs 추가  **# vi netreport\_pjt/urls.py**  from django.contrib import admin  from django.urls import path, include  from django.conf import settings  from django.conf.urls.static import static  - 각 앱에 맞는 URL 경로로 접근할 수있도록 각 앱에 대한 경로를 urls.py에 설정  urlpatterns = [  path('main/', include('report\_main.urls')),  path('list/', include('report\_list.urls')),  path('accounts/', include('report\_admin.urls')),  path('admin/', admin.site.urls),  ] + static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT)  report\_main 앱의 View 작성  - Django의 View는 사용자가 요청한 데이터를 처리하고, 템플릿으로 전달하는 역할  >index 및 detail View 작성  **# vi report\_main/views.py**  from django.shortcuts import render  from django.http import HttpResponse  import random  from .models import Contents  from django.shortcuts import redirect  from .forms import ContentsForm  from django.contrib.auth.decorators import login\_required, permission\_required  # Create your views here.  def index(request):  texts = Contents.objects.all().order\_by("-id")  # r\_texts = reversed(texts)  foods = ['apple', 'banana', 'coconut']  pick = random.choice(foods)  info = {  'name': 'kildong',  'age': 20,  }  context = {  'pick': pick,  'foods': foods,  'info': info,  'texts': texts,  }    return render(request, 'report\_main/index.html', context)  def detail(request, pk):  text = Contents.objects.get(id=pk)  context = {  'text': text  }  return render(request, 'report\_main/detail.html', context)  **# vi report\_main/urls.py**  from django.urls import path  from . import views  app\_name = 'report\_main'  urlpatterns = [  path('', views.index, name='index'),  path('<int:pk>/', views.detail, name='detail'),  ]  report\_main 앱의 템플릿 작성  - HTML 템플릿을 작성하여 View에서 전달한 데이터를 화면에 표시  **# vi report\_main/templates/report\_main/index.html**  > 상용 템플릿 활용  {% extends 'base.html' %}  {% include 'title.html' %}  {% block content %}  {% load static %}  ...  ...  {% for cont in texts %}  <div class="image">  <h3>  <a href="#" class="tag category">Security</a>  <a href="{% url 'report\_main:detail' cont.pk %}" class="title">{{ cont.title }}</a>  <!-- <span class="tag">Offer</span> -->  </h3>  <a href="{% url 'report\_main:detail' cont.pk %}" class="image-wrapper background-image">  <img src="{{ cont.image.url }}" alt="">  </a>  </div>  <div class="meta">  <figure>  <i class="fa fa-calendar-o"></i>{{ cont.updated\_at }}  </figure>  </div>  <div class="description">  <p>{{ cont.content }}s</p>  </div>  {% endfor%}  ...  ...  {% endblock content %}  **# vi report\_main/templates/report\_main/detail.html**  {% extends 'base.html' %}  {% block content %}  <div class="article-title">  <h2><a href="#">{{ text.title }}</a></h2>  <div class="tags framed">  <a href="#" class="tag">Security</a>  <!-- <a href="#" class="tag">Design</a> -->  </div>  </div>  <div class="meta">  <figure>  <a href="#" class="icon">  <i class="fa fa-user"></i>  관리자  </a>  </figure>  <figure>  <i class="fa fa-calendar-o"></i>  {{ text.updated\_at }}  </figure>  </div>  ...  ...  {% endblock content %}  Django 개발 서버 실행  - 애플리케이션 실행  # python manage.py runserver 0.0.0.0:8000  웹 브라우저를 통해 Django 애플리케이션에 접근  > 223.130.147.99:8000 열기 | |
| 부가설명 | Django 앱을 생성할 때, 앱의 기능에 따라 적절히 분리하여 관리하는 것이 중요함.  MySQL 데이터베이스와의 연동을 위해 Django의 DATABASES 설정을 정확하게 해야 함.  Django 템플릿 언어를 사용하여 동적으로 웹 페이지를 생성할 수 있으며, HTML 코드 내에서 템플릿 태그를 활용하여 효율적인 페이지 구성이 가능함.  운영 환경에서는 개발 서버 대신 Nginx와 같은 웹 서버와 Gunicorn 같은 WSGI 서버를 이용해 배포해야 함. |

|  |  |
| --- | --- |
| 기능구현 ID | R002-001 |
| 목적 | 웹서버 로그인/로그아웃 |
| - AuthenticationForm을 사용해 로그인 기능을 구현  로그인 뷰 작성  사용자가 로그인하면 메인 페이지로 리다이렉트  **# vi report\_admin/views.py**  login() 함수 작성  로그인 요청 시 POST 방식으로 받은 데이터를 인증 후 로그인 처리  로그인 페이지 렌더링 시 AuthenticationForm을 사용  - 로그인 페이지의 HTML 파일인 login.html은 AuthenticationForm을 폼으로 사용해 로그인 UI를 구현  from django.shortcuts import render, redirect  from django.contrib.auth.forms import AuthenticationForm, PasswordChangeForm  from django.contrib.auth import login as auth\_login  from django.contrib.auth import logout as auth\_logout  def login(request):  # 이미 로그인한 경우 메인 페이지로 리다이렉트  if request.user.is\_authenticated:  return redirect('report\_main:index')    # POST 요청 처리  if request.method == 'POST':  form = AuthenticationForm(request, request.POST)  if form.is\_valid():  auth\_login(request, form.get\_user()) # 로그인 처리  return redirect('report\_main:index') # 로그인 성공 시 메인 페이지로 이동    else:  form = AuthenticationForm()  # 로그인 페이지 렌더링  context = { 'form': form }  return render(request, 'report\_admin/login.html', context)  logout() 함수 작성  사용자가 로그아웃하면 메인 페이지로 리다이렉트  def logout(request):  auth\_logout(request) # 로그아웃 처리  return redirect('report\_main:index') # 메인 페이지로 리다이렉트  URL 설정  로그인/로그아웃 URL 설정  - login/과 logout/ 경로에 대해 해당 뷰를 연결해서 사용자가 로그인과 로그아웃을 할 수 있게 만드는 과정  >login/ URL은 login() 뷰와 연결  logout/ URL은 logout() 뷰와 연결  **# vi report\_admin/urls.py**  from django.urls import path  from . import views  app\_name = 'report\_admin'  urlpatterns = [  path('login/', views.login, name='login'), # 로그인 URL  path('logout/', views.logout, name='logout') # 로그아웃 URL  ]  사용자 모델 설정  >기본 제공되는 AbstractUser를 상속받은 User 모델 작성  - 추가적인 필드나 기능을 추가할 수 있도록 확장가능  # vi report\_admin/models.py  from django.db import models  from django.contrib.auth.models import AbstractUser  # Create your models here.  # 사용자 모델 정의  class User(AbstractUser):  pass  로그인 폼을 구성하는 HTML 템플릿 작성  **# vi report\_admin/templates/report\_admin/login.html**  base.html을 상속받아 로그인 폼 생성  POST 요청으로 로그인 데이터 전송  Django의 AuthenticationForm을 사용하여 기본적인 로그인 폼을 구성하고, CSRF 보호를 위해 csrf\_token을 포함함.  {% extends 'base.html' %}  {% block content %}  <h1>로그인</h1>  <hr>  <form action="{% url 'report\_admin:login' %}" method="POST">  {% csrf\_token %}  {{ form.as\_p }}  <input type="submit" value="로그인"><br>  </form>  {% endblock content %}  기타 설정  Django의 기본 인증 시스템을 사용하여 로그인 및 로그아웃 구현  추가적인 사용자 모델 확장 시 User 모델을 기반으로 커스텀 가능 | |
| 부가설명 | 로그인 기능은 Django의 기본 AuthenticationForm을 사용하여 쉽게 구현 가능하며, 이 폼은 Django가 제공하는 인증 기능을 포함함.  로그인 폼에는 CSRF 보호를 위해 {% csrf\_token %}을 포함시켜 보안을 강화함.  운영 환경에서는 보안 인증을 강화하기 위해 HTTPS 및 추가적인 보안 설정을 고려해야 함. |

|  |  |
| --- | --- |
| 기능구현 ID | R003-001 |
| 목적 | Suricata 구성 |
| **(PPA 설치 -> Suricata를 안정적으로 운영하기 위함)**  \*PPA : Ubuntu의 기본 소프트웨어 저장소에서 제공되지 않는 소프트웨어나 최신 버전을 쉽게 제공  **> 시스템에 software-properties-common 패키지 설치, 이 패키지 안에 PPA설치**  # sudo apt install software-properties-common –y  **> Suricata의 안정적인 버전이 포함된 PPA를 추가**  # sudo add-apt-repository ppa:oisf/suricata-stable  **> 패키지 목록을 업데이트**  **# sudo apt update –y**  **(Suricata 설치)**  **> Suricata 설치 명령어**  **# sudo apt install suricata**  **> eve.json 로그관련 파일을(네트워크 로그) 쉽게 처리하고 필터링할 수 있도록 보조**  \*eve.json : Elasticsearch와와 연동되어 실시간으로 보안사항 분석과 시각화를 가능하게 하는 핵심 로그파일  # sudo apt install suricata jq  **> Suricata 서비스의 현재 상태를 확인**  # sudo systemctl status suricata  **(Suricata가 네트워크 인터페이스에서 패킷을 수집하고 처리하는 방식을 정의)**  **> Suricata가 고성능 네트워크 환경에서 패킷을 효율적으로 처리하기 위한 설정**  **> eth0 인터페이스에서 트래픽을 분석하며, 클러스터링과 멀티스레딩을 통해 트래픽을**  **병렬 처리하고 성능을 최적화**  **> 패킷 조각화 해제(Defrag)를 통해 완전한 트래픽 흐름을 분석, 메모리 매핑과**  **TPACKET v3 인터페이스를 활용하여 패킷 캡처 성능을 향상**  # vi /etc/suricata/suricata.yam  ...  ...  af-packet:  - interface: **eth0**  cluster-id: 99  cluster-type: cluster\_flow  defrag: yes  use-mmap: yes  tpacket-v3: yes  ... | |
| 부가설명 | 위의 과정들을 통해 suricata버전을 자동으로 최신화 할 시   * 지속적인 관리가 용이함 * 네트워크 패킷에 대한 효율적 처리 구성 * suricata 성능을 최적화하여 관리할 수 있음 |

|  |  |
| --- | --- |
| 기능구현 ID | R003-002 |
| 목적 | Suricata Rule 설정 |
| **(기본 Rule 설정)**  **> 현재 설정되어 있는 rule이 없음을 확인 가능**  # vi /var/lib/suricata/rules  **> rule 파일 생성: suricata.rules**  # sudo suricata-update  **(사용자 정의 Rule 설정)**  **> local.rules에서 아래의 예시와 같은 사용자 정의 rule 설정**  # vi /var/lib/suricata/rules/local.rules  **> TLS 트래픽을 특정 서버 포트(예: 443)에 대해서 차단하고, 그에 대한 로그를 남기는**  **설정**  drop tls any any-> any <차단할서버포트> (msg:"차단메시지 :)"; classtype:misc-activity; sid:1000015; priority:1;) | |
| 부가설명 | Suricata의 rule 구성방식은 많은 부분에 있어서 snort와 동일하거나 유사하나, 전부 동일하지 않은 점을 감안하여 rule 구성 시 테스트를 진행하여 적용완료 여부에 대한 체크가 중요 |

|  |  |
| --- | --- |
| 기능구현 ID | R003-003 |
| 목적 | 운영목적에 따른 Suricata 역할 부여 |
| **(IDS 운영: 침입 탐지모드)**  **> IDS 모드를 운영하기 위한 명령어 -i로 IDS 모드 식별**  # suricata -c /etc/suricata/suricata.yaml -i eth0  **(IPS 운영: 침입 방지모드)**  **> IPS 모드를 운영하기 위한 명령어 -q로 IPS 모드 식별**  # suricata -c /etc/suricata/suricata.yaml -q eth0 | |
| 부가설명 | 사용목적(네트워크 이상행동 탐지 및 방지)에 따라서 IDS 혹은 IPS로 설정하여 목적성에 맞게 suricata 운영 |

|  |  |
| --- | --- |
| 기능구현 ID | R003-004 |
| 목적 | Log 데이터 수집, 분석 |
| **(Suricata Log 데이터 모니터링)**  # /var/log/suricata/  - eve.json  \*eve.json : Elasticsearch와와 연동되어 실시간으로 보안사항 분석과 시각화를 가능하게 하는 핵심 로그파일  - fast.log  \*fast.log: 탐지된 보안 이벤트 및 경고 정보를 빠르고 간단하게 요약한 로그. 신속한 경고 모니터링 및 대응이 가능  - stats.log  \*stats.log: Suricata의 성능과 상태 정보를 기록하는 로그. 성능 모니터링 및 트러블슈팅.  **(eve.json 파일 내 로그 구성 확인)**  **> JSON 포멧으로 편집된 log 파일**  # vi /var/log/suricata/eve.json  {  "timestamp": "2024-08-17T01:02:42.528502+0900",  "flow\_id": 840373234764068,  "in\_iface": "eth0",  "event\_type": "tls",  "src\_ip": "10.0.1.101",  "src\_port": 39402,  "dest\_ip": "151.101.198.49",  "dest\_port": 443,  "proto": "TCP",  "pkt\_src": "wire/pcap",  "tls": {  "subject": "CN=cdn.fwupd.org",  "issuerdn": "C=BE, O=GlobalSign nv-sa, CN=GlobalSign Atlas R3 DV TLS CA 2024 Q2",  "serial": "01:1B:8A:E8:D6:6C:74:C6:4F:E2:3B:B9:AF:8B:AB:28",  "fingerprint": "1e:12:13:c4:e1:cf:2b:7b:06:73:6d:fe:9a:df:2d:ca:1e:b5:4e:a7",  "sni": "cdn.fwupd.org",  "version": "TLS 1.2",  "notbefore": "2024-04-10T19:52:04",  "notafter": "2025-05-12T19:52:03"  }  }{  "timestamp": "2024-08-17T01:02:49.408586+0900",  "event\_type": "stats",  "stats": {  "uptime": 8,  "capture": {  "kernel\_packets": 1407,  "kernel\_drops": 0,  "errors": 0,  "afpacket": {  "busy\_loop\_avg": 0,  "polls": 945,  "poll\_signal": 0,  "poll\_timeout": 213,  "poll\_data": 732,  "poll\_errors": 0,  "send\_errors": 0  }  },  ...  ...  **> 위 이벤트의 내용은 Suricata가 네트워크 트래픽을 모니터링하고 TLS 세션과 시스템 성능 통계를 로그로 남긴 결과** | |
| 부가설명 | Eve.json, fast.log 및 stats.log 파일 확인 시   * suricata의 성능에 대한 확인 및 효율성을 재고할 수 있음 * 네트워크 트래픽의 상세내용을 ELK Stack에 연동하여 확인 및 대응 가능 |

|  |  |
| --- | --- |
| 기능구현 ID | R004-001 |
| 목적 | Elastic Stack 구성 |
| **Elasticsearch 설치 (자동 or 수동)**  > **(자동) Elasticsearch 패키지 설치 명령어**  # curl -fsSL https://artifacts.elastic.co/GPG-KEY-elasticsearch |sudo gpg --dearmor -o /usr/share/keyrings/elastic.gpg  # sudo apt-get update && sudo apt-get install elasticsearch  > **(수동) Elasticsearch 패키지 설치 명령어**  # wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.15.1-amd64.deb  # sudo dpkg -i elasticsearch-8.15.1-amd64.deb  **> 설치된 Elast+icsearch를 수동으로 실행**  # bin/elasticsearch  **> Elasticsearch 설정 파일 수정**  # vi /etc/elasticsearch/elasticsearch.yml  ...  ...  # ---------------------------------- Network -----------------------------------  #  # By default Elasticsearch is only accessible on localhost. Set a different  # address here to expose this node on the network:  #  **> 네트워크 설정을 0.0.0.0으로 변경하여 외부 접근 허용**  **network.host: 0.0.0.0**  ...  ...  **> 노드 간 통신을 위해 사용(모든 IP 주소에서의 통신을 허용)**  **discovery.seed\_hosts: "0.0.0.0"**  ...  ...  **> Elasticsearch 서비스 시작 및 시스템 부팅 시 자동 시작 설정**  # sudo systemctl start elasticsearch  # sudo systemctl enable elasticsearch  **> 외부에서 Elasticsearch 클러스터에 접근할 수 있는지 확인**  # curl -X GET "223.130.147.99:9200"  **Kibana 설치**  **> Kibana 패키지 설치 명령어**  # sudo apt install kibana  **> Kibana 서비스 시작 및 시스템 부팅 시 자동 시작 설정**  # sudo systemctl start kibana  # sudo systemctl enable kibana  **Nginx 리버스 프록시 설정**  **> Nginx 서버의 IP 또는 도메인으로 접속하면 Nginx가 Kibana 서버로 요청을 전달**  **> Nginx 설정 파일 수정**  # sudo vi /etc/nginx/sites-available/default  server {  **> Nginx가 포트 80(HTTP 기본 포트)에서 요청을 수신하도록 설정**  listen 80;  **(리버스 프록시 설정)**  **> 해당 서버에 할당된 도메인 또는 IP를 지정(\_는 기본으로 모든 도메인과 IP에 대해 적용하도록 설정)**  server\_name \_;  **> Nginx가 /(루트 경로)로 들어오는 모든 요청을 처리하도록 지정**  location / {  **> Nginx가 수신한 요청을 Kibana가 실행 중인 서버로 전달**  proxy\_pass http://localhost:5601;  **(추가 헤더 설정)**  **> HTTP 버전을 1.1로 설정**  proxy\_http\_version 1.1;  **> Kibana와의 연결에서 프로토콜 업그레이드를 지원**  proxy\_set\_header Upgrade $http\_upgrade;  **> 웹소켓 연결을 위한 설정**  proxy\_set\_header Connection 'upgrade';  **> 원본 요청의 호스트 정보를 Kibana로 전달**  proxy\_set\_header Host $host;  **> 캐시를 우회하도록 설정**  proxy\_cache\_bypass $http\_upgrade;  } }  **> Nginx 설정을 다시 불러옴(Nginx 재시작없이 변경된 설정 적용)**  # systemctl reload nginx  **Logstash 설치**  **> Logstash를 사용하진 않지만 연동을 위해 설치, Filebeat와 Elasticsearch와의 통합을 위해 입출력 설정을 구성**  > **Logstash 설치 명령어**  # sudo apt install logstash  **> Logstash의 입력 설정을 위한 파일 생성하거나 수정**  # sudo vi /etc/logstash/conf.d/beats-input.conf  **> beats 플러그인을 사용하여 데이터 수집 도구(Filebeat)에서 데이터를 수신하도록 설정하고 5044 포트에서 데이터 수신**  input {  beats {  port => 5044  } }  **> 출력 설정을 위한 파일 생성하거나 수정(Logstash가 처리한 데이터를 Elasticsearch로 전송하는 방식)**  # sudo vi /etc/logstash/conf.d/elasticsearch-output.conf  output {  if [@metadata][pipeline] {  elasticsearch {  **> Logstash가 데이터를 보낼 Elasticsearch의 호스트와 포트를 지정**  hosts => ["localhost:9200"]  manage\_template => false  **> 수신된 데이터는 일자별 인덱스로 저장**  index => "%{[@metadata][beat]}-%{[@metadata][version]}-%{+YYYY.MM.dd}"  **> Elasticsearch 파이프라인을 지정하여 추가 처리**  pipeline => "%{[@metadata][pipeline]}"  }  } else {  elasticsearch {  hosts => ["localhost:9200"]  manage\_template => false  index => "%{[@metadata][beat]}-%{[@metadata][version]}-%{+YYYY.MM.dd}"  }  } }  **> 설정 파일 검증(오류가 있으면 로그로 출력)**  # sudo -u logstash /usr/share/logstash/bin/logstash --path.settings /etc/logstash -t  **> Logstash 서비스 자동 시작 설정 및 실행**  # sudo systemctl enable logstash  # sudo systemctl start logstash | |
| 부가설명 | 1. 보안을 위한 security 설정 필요  1) **Elasticsearch 보안 설정**   * **관리자 계정 암호 설정**   # sudo bin/elasticsearch-setup-passwords interactive   * **TLS 설정(통신 암호화)** (#elasticsearch.yml)   xpack.security.transport.ssl.enabled: true xpack.security.http.ssl.enabled: true  2) **Kibana 보안 설정**   * **Kibana ID, Password 설정**(#kibana.yml)   elasticsearch.username: "kibana\_system"  elasticsearch.password: "비밀번호"   * **Kibana HTTPS 활성화**(#kibana.yml)   \* 자체 서명 인증서나 공인 인증서를 사용할 수 있음  server.ssl.enabled: true  server.ssl.certificate: /path/to/your/certificate.pem  server.ssl.key: /path/to/your/certificate.key  3) **Nginx 리버스 프록시 보안 설정**   * **ID, Password 및 SSL 설정**(#etc/nginx)   # sudo htpasswd -c /etc/nginx/.htpasswd user1  사용자 및 비밀번호 설정 후 Nginx 설정에서 적용, SSL 내용도 추가하기  4) **Logstash 보안 설정**   * **ID, Password 및 SSL 설정** (#logstash.yml)   output {  elasticsearch {  ....  user => "elastic"  password => "비밀번호"  ssl => true  }  2. Elastic Stack 구성 시 참고사항  1) **Elasticsearch 인덱스 관리 :** 인덱스 생명 주기 관리(ILM)를 설정하여 오래된 인덱스를 자동으로 삭제하거나 보관  2) **모니터링 및 알림 설정 :** Elastic Stack에서 제공하는 Watcher나 Alerting 기능 활용  3) **Elastic Stack 업그레이드 관리** : 정기적으로 업그레이드하여 최신 보안 패치와 성능 개선을 적용 (\*업그레이드 시에는 반드시 백업을 수행하고, 테스트 환경에서 먼저 적용한 후 운영 환경에 배포) |

|  |  |
| --- | --- |
| 기능구현 ID | R004-002 |
| 목적 | Filebeat 와 Suricata 연동 |
| **filebeat 설치**  **> 운영중인 엘라스틱에 접속**  > 223.130.147.99 로그인  > 메인 페이지 ‘Add integrations’ > ‘Suricata’ 검색  > Suricata integration 페이지 하단 ‘Also available in Beats’ 메뉴 선택  **> Filebeat 다운로드 및 설치** (Suricata Logs)  # curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.15.0-amd64.deb  # sudo dpkg -i filebeat-8.15.0-amd64.deb  **> Filebeat 설정 수정**  # vi /etc/filebeat/filebeat.yml  ...  ...  **(수집한 로그 데이터를 Elasticsearch로 전송하는 설정)**  **> 로그를 전송할 Elasticsearch 호스트와 포트를 지정**  output.elasticsearch:  # Array of hosts to connect to.  hosts: ["**223.130.147.99:9200**"]    # Performance preset - one of "balanced", "throughput", "scale",  # "latency", or "custom".  **> 성능 설정 ("balanced"는 균형 잡힌 성능을 제공)**  preset: balanced    # Protocol - either `http` (default) or `https`.  #protocol: "https"    # Authentication credentials - either API key or username/password.  #api\_key: "id:api\_key"  **> 인증을 위한 사용자 이름과 비밀번호를 입력**  **username: "elastic"**  **password: "엘라스틱PW"**  ...  > **Filebeat에 Suricata 모듈을 활성화하여 Suricata 로그를 수집할 수 있도록 설정**  # sudo filebeat modules enable suricata  **> Suricata 모듈 설정 파일을 수정하여 로그 수집을 활성화**  # vi /etc/filebeat/modules.d/suricata.yml  ...  ...  **> Suricata의 EVE JSON 로그를 활성화하여 수집**  - module: suricata  # All logs  eve:  enabled: **true**  ...  **> Filebeat가 Kibana 대시보드와 Elasticsearch 템플릿을 설정**  # sudo filebeat setup  **> Filebeat 서비스 시작**  # sudo service filebeat start  **> 대시보드를 이용해 Suricata 이벤트를 모니터링**  ‘Check data’ 확인 후 ‘Suricata Events Overview’로 이동하여 확인 | |
| 부가설명 | 1. Filebeat와 Suricata 연동 과정 유의사항  1) **Filebeat 버전 호환성 :** 버전 불일치로 인해 데이터 전송이나 인덱싱 과정에서 오류가 발생할 수 있으므로 Filebeat 버전과 Elasticsearch 및 Suricata의 버전 간의 호환성을 반드시 확인  2) **필수 설정 파일 확인 :** filebeat.yml에서 hosts, username, password 설정이 잘못되면 Elasticsearch에 로그가 수집되지 않을 수 있으므로 입력된 호스트와 자격증명 정보를 정확히 입력  3) **포트 및 방화벽 확인 :** Elasticsearch 포트(기본 9200)와 Filebeat 포트(기본 5044)가 제대로 열려 있는지 확인  2. 발생할 수 있는 에러 및 해결 방법  1) **로그가 Elasticsearch에 나타나지 않는 문제 :** Filebeat가 정상적으로 실행되고 있음에도 Suricata 로그가 Elasticsearch에서 검색되지 않는 경우.   * **원인**: Filebeat와 Elasticsearch 간의 통신 문제가 있거나 Filebeat가 Suricata 로그 파일을 올바르게 읽지 못하는 경우 * **해결 방법**: #sudo journalctl -u filebeat 또는 #filebeat logs 명령어로 로그 확인하고 Suricata 로그 파일 경로가 올바르게 설정되었는지 확인   2) **Filebeat 서비스가 시작되지 않는 문제**   * **원인**: filebeat.yml 설정 파일에 문법 오류 확인하기 * **해결 방법**: YAML 파일의 들여쓰기나 콜론(:)이 잘못될 경우 서비스가 정상적으로 시작되지 않을 수 있음   filebeat test config 명령으로 설정 파일 테스트 필요  3. 설정 시 참고사항  1) **Suricata 로그 경로 확인 :** 기본적으로 /var/log/suricata/eve.json에 로그가 저장되지만, 환경에 따라 경로가 다를 수 있음   * Suricata.yml에서 Suricata의 로그 경로 확인 후 Filebeat 설정에서 paths 항목에 해당 경로를 정확히 입력   2) **Filebeat 설정 테스트** **:** Filebeat의 설정을 수정한 후 반드시 설정 테스트 진행하기   * sudo filebeat test config 명령어 사용   3) **Filebeat의 모니터링 및 로그 확인 :** Filebeat는 실행 상태나 오류 메시지를 systemctl 명령어를 통해 기록하며, 이를 정기적으로 점검하여 정상 작동하고 있는지 확인하기   * sudo journalctl -u filebeat 명령어 사용   4) **Suricata 모듈 데이터 파이프라인 확인** : Suricata 로그가 특정 파이프라인을 통해 Elasticsearch로 전송되기 때문에 데이터 파이프라인 설정을 확인하는 것이 중요   * filebeat setup 명령어 사용 |
| 기능구현 ID | R004-003 |
| 목적 | Elastic 시각화 구성 |
| **(Elastic 메인 메뉴로 이동)**  Elastic 왼쪽 상단의 메인 메뉴를 열고 대시보드를 선택한다.  > 경로 : Analytics – Dashboards  **(Suricata 대시보드 검색)**  찾기 메뉴에서 “Suricata”를 검색하고, 검색한 내용 중 Filebeat 와 연동된 Events Overview를 선택한다.  > Search : “Suricata” - [Filebeat Suricata] Events Overview  **(대시보드 편집)**  **> 편집을 시작하려면 Edit를 선택하고, 다양한 시각화 옵션을 추가**  > Edit - Create visualization / Add panel / Add from library / Control  **(대시보드 저장 및 공유)**  > Save > Switch to view mode – Full screeen / Share : Links, Embed  **(시간 필터 설정)**  **> 대시보드의 시간 범위를 설정할 때, Absolute, Relative, Now 등의 옵션을 선택할 수 있으며 필요할 때 Refresh 버튼을 눌러 데이터를 최신으로 갱신**  > Date - Absolute / Relative / Now > Refresh | |
| 부가설명 | 1. 대시보드 설정 시 유의사항   1. **적절한 시각화 유형 선택**: 데이터를 시각화할 때 데이터의 목적에 맞는 시각화 유형을 선택하는 것이 중요   ex) 라인 차트: 시간에 따른 변화 (네트워크 트래픽 변화)  막대, 파이 차트 : 시간에 따른 변화 (보안 이벤트 발생 비율)     1. **필터 사용의 중요성**: 모든 데이터를 시각화하면 대시보드가 복잡해지거나 응답 속도가 느려짐으로 필터를 사용하여 필요한 데이터만 보여주는 것이 중요   2. 시간 범위 설정 시 참고사항   1. **시간 범위 설정에 따른 데이터 변화**: Elastic에서 데이터를 분석할 때 Absolute, Relative, Now 등의 시간 범위를 설정하여 과거와 현재 데이터를 비교할 수 있음. Relative로 설정하면 실시간 데이터와 과거 데이터를 비교 분석하는 데 유리 2. **실시간 데이터 업데이트**: Now 옵션을 설정하면 실시간 데이터를 볼 수 있음   3. 대시보드 편집 및 시각화 패널 추가 팁   1. **Add from Library**: 자주 사용하는 시각화 패널은 저장해 두고, 편리하게 재사용 가능 2. **데이터 간 연관성 분석** : 하나의 패널에서 발생한 보안 이벤트를 클릭하면 관련된 IP 트래픽, DNS 요청 등이 다른 패널에서 함께 갱신되도록 대시보드 설정 가능   4. 대시보드 공유 시 주의사항   1. **대시보드 공유 범위:** 적절한 권한 설정 필요, 반드시 보안 그룹을 설정하고, 비인가자가 접근할 수 없도록 제한   5. Suricata 데이터 시각화 유용한 팁   1. **이벤트 중첩 및 필터링** : 여러 종류의 이벤트가 동시에 발생할 수 있기 때문에, 시각화할 때 중첩된 이벤트를 필터링하여 특정 유형의 보안 위협만을 분석   ex) severity나 category 필드를 필터링하여, 심각한 보안 위협만 따로 분석하는 대시보드를 구성 |

|  |  |
| --- | --- |
| 기능구현 ID | R005-001 |
| 목적 | Elastic 로그인 |
| **> Security 설정을 위해 Elasticsearch 설정파일을 연다.**  > # sudo vi /etc/elasticsearch/elasticsearch.yml  **> 보안 설정의 두가지 항목을 변경하여 활성화 한다.**  > # Enable security features  xpack.security.enabled: **true**  # Enable encryption and mutual authentication between cluster nodes  xpack.security.transport.ssl:  enabled: **true**  verification\_mode: certificate  keystore.path: certs/transport.p12  truststore.path: certs/transport.p12  **> Elasticsearch 서비스를 재시작 한다.**  > # sudo systemctl restart elasticsearch  **> 기본 암호 생성을 위하여 Elasticsearch 명령어를 실행한다.**  > # /usr/share/elasticsearch/bin/elasticsearch-setup-passwords auto  **키바나 설정을 변경하여 보안 접속을 위한 유저 ID/PW 활성화한다.**  > # sudo vi /etc/kibana/kibana.yml  ...  ...  elasticsearch.username: "**kibana**"  elasticsearch.password: "**kibana**"  ...  **> Kibana 서비스를 재시작한다.**  > sudo systemctl restart kibana  **> 설정한 ID/PW로 Elastic 접속한다.** | |
| 부가설명 | 반드시 **보안 설정**을 활성화하여 운영한다.   * 암호 생성 시 .../elasticsearch-setup-passwords **interative** 옵션을 실행하여 직접 입력할 수 있다. |

|  |  |
| --- | --- |
| 기능구현 ID | R005-002 |
| 목적 | Log 데이터 확인, 검색 |
| **> Elastic Stack에 접근한다.**  > 223.130.147.99  **> 관리자 계정의 ID/PW를 입력하여 로그인 한다.**  **> 왼쪽 상단의 메인 메뉴을 열고 Log 검색창을 활성화한다.**  > Analytics – Discover  **> 기본 모듈인 filebeat를 선택하고 필요한 필터 등을 설정한다.**  > Data views : “filebeat-\*” - Add filter / KQL syntax – field names  **> 필요한 Log 데이터의 기간을 정하고, Refresh 하여 검색 내용을 확인한다.**  > 기간 : Absolute / Relative / Now > Refresh | |
| 부가설명 | 1. **Share 메뉴**   * 검색 내용의 Link 추가하거나 CSV 파일로 저장이 가능하다.   2. **Alerts 메뉴**   * 필요한 로그에 대한 알림을 받을 수 있도록 설정이 가능하다. |

|  |  |
| --- | --- |
| 기능구현 ID | R005-003 |
| 목적 | Log 데이터 분석, 시각화 |
| **>Elastic Stack에 접근한다.**  > 223.130.147.99  **> Elastic 왼쪽 상단의 메인 메뉴를 열고 대시보드를 선택한다.**  > Analytics – Dashboards  **> (1) 찾기 메뉴에서 “Suricata”를 검색하고, 검색한 내용 중 Filebeat 와 연동된 Events Overview를 선택한다.**  > Search : “Suricata” - [Filebeat Suricata] Events Overview  **> 리포트 작성과 관련하여 커스텀 된 Log 필드 내용을 확인한다.**  > 활성화 필드 : Activity Types over Time [Filebeat Suricata],  Top Hosts Generating Events [Filebeat Suricata],  Event Types [Filebeat Suricata],  Top Network Protocols [Filebeat Suricata],  Top Connection Source Countries [Filebeat Suricata],  Alerts - Top Source Countries [Filebeat Suricata],  Events [Filebeat Suricata],  Host Stats [Filebeat Suricata],  Alerts [Filebeat Suricata]  **> 필요한 Log 값에 대한 filter, 기간 설정 등을 통해 관련 내용만을 시각화할 수 있다.**  > Fliter (KQL syntax) / 기간 : Absolute / Relative / Now > Refresh  **> (2) 찾기 메뉴에서 “Suricata”를 검색하고, 검색한 내용 중 Filebeat 와 연동된 Alert Overview를 선택한다.**  > Search : “Suricata” - [Filebeat Suricata] Alert Overview  **> 리포트 작성과 관련하여 커스텀 된 Alerts 대시보드 필드를 확인한다.**  > 활성화 필드 : Top Alerting Hosts [Filebeat Suricata],  Top Alert Signatures [Filebeat Suricata],  Alerts - Top Destination Countries [Filebeat Suricata],  Alerts - Top Source Countries [Filebeat Suricata],  Alert - Source Location [Filebeat Suricata],  Alerts [Filebeat Suricata]  **> Filter, 기간 등을 설정하여 필요한 Log 나 Alert의 관련 내용만을 시각화할 수 있다.**  > Fliter (KQL syntax) / 기간 : Absolute / Relative / Now > Refresh | |
| 부가설명 | 1. **Share 메뉴**   * 시각화 내용의 Links 를 추가하거나, 선택한 항목을 포함하여 대시보드에 삽입할 수 있다.   2. **Edit 메뉴**   * 새로운 시각화 대시보드를 만들거나 기존의 필드를 수정할 수 있다. |

|  |  |
| --- | --- |
| 기능구현 ID | R006-001 |
| 목적 | Report 자동화 웹페이지 구성 |
| **> 관리자 페이지 메뉴를 통해 로그 모니터링/리포트 주소에 접근할 수 있도록 구성한다.**  > # vi report\_main/templates/report\_main/index.html  ...  ...  {% if user.is\_authenticated %}  <li class="nav-item">  <a href="**http://223.130.147.99/app/dashboards#/view/78289c40-86da-11e8-b59d-21efb914e65c-ecs**" target="\_blank" class="btn btn-success text-caps btn-rounded btn-framed">**Control**</a>  </li>  <li class="nav-item active has-child">  <a class="nav-link" href="">**REPORT**</a>  <ul class="child">  <li class="nav-item">  <a href="{% url '**report\_list:list**' %}" class="nav-link">**REPORT 1**</a>  </li>  ....  **> 엘라스틱 검색/분석 성능을 활용한 documentary를 작성하여 리포트 웹페이지 구성한다.**  > # vi report\_list/views.py  from django.shortcuts import render, redirect  **import elasticsearch**  **from elasticsearch import Elasticsearch**  import datetime  from django.contrib.auth.decorators import login\_required, permission\_required  ...  ...  @permission\_required('report\_main.can\_view\_content')  def index(request):  now = datetime.datetime.now().date()  day\_one = datetime.timedelta(days=1)  yesterday = now - day\_one  name = request.user.last\_name + request.user.first\_name  email = request.user.email  **es = Elasticsearch(**  **"http://223.130.147.99:9200", api\_key="ZGlGRHM1RUIyVEUtRmQxMkZPVmc6Y2JVS1JRbGRSZC15ZURBYzZfZmlrUQ==",**  **)**  **doc1 = {**  **"track\_total\_hits": True,**  **"query": {**  **"match": {"@timestamp": yesterday}**  **},**  **}**  ...  ...  **res1 = es.search(index=".ds-filebeat-8.15.0-2024.08.17-000001", body=doc1)**  ...  ...  datas = res1['hits']['total']  ...  ...  context = {  'res': datas['value'],  'avg\_res': round(datas['value']/24/60, 1),  'date': yesterday,  ...  ...  }  return render(request, 'report\_list/index.html', context)  ...  **> 웹페이지 주소를 연결한다.**  > # vi report\_list/urls.py  from django.urls import path  from . import views  app\_name = 'report\_list'  urlpatterns = [  **path('', views.index, name='list'),**  ...  ...  ]  **> 웹페이지에 Context를 통해 받은 data를 활용하여 Report 자동화를 구성한다.**  > # vi report\_list/templates/report\_list/index.html  {% extends 'base.html' %}  {% block content %}  <table>  <h1>네트워크 보안 관제 Daily Report</h1></table>  <br>  ...  ...  <h3><p class = "bold">네트워크 정보</p></h3>  네트워크 토폴로지 : <br>  주요 네트워크 프로토콜 및 서비스 : 80 Port / HTTP / Web Service<br>  주요 서버 및 장비 상태 : <font color="green">그린</font><br>  <hr>  <h3><p class = "bold">트래픽 분석 정보</p></h3>  주요 트래픽 서비스 : **{{ server }}**<br>  주요 패킷 사이트 : KR / US<br>  평균 트래픽 대역 : **{{ avg\_res }}** C/M<br>  트래픽 이상 징후 : <font color="green">**그린**</font><br>  <hr>  <h3><p class = "bold">로그 분석 정보</p></h3>  Daily 로그 총량 : **{{ res }}** 개<br>  의심스러운 활동 및 이벤트 : <font color="green">**그린**</font><br>  <br>  ...  ...  {% endblock content %}  **> 웹서버에 접근하여 관리자로 로그인한다.**  > 223.130.147.99:8000/main/  **> Report 메뉴를 선택하여 자동화된 보고서 웹페이지에 접근한다.**  > 223.130,147.99:8000/list/ | |
| 부가설명 | 1. 고도화를 통해 메뉴에서 원하는 주제의 리포트를 선택할 수 있게 구성한다.  2. 권한 설정을 관리하여 관리자 계정과 일반 사용자를 분리한다. |

|  |  |
| --- | --- |
| 기능구현 ID | R006-002 |
| 목적 | Report 자동화 고도화 - 저장하기, 리포트 형식 선택 |
| **> 작성된 Daliy 보고서를 파일로 생성하고 DB에 저장한다.**  > # vi report\_list/views.py  ...  ...  **import pdfkit**  ...  def makepdf(request):  options = {'quiet': ''} # wkhtmltopdf 출력값을 화면에 표시하지 않는다.  config = pdfkit.configuration(wkhtmltopdf = '/usr/bin/wkhtmltopdf')  **pdfkit.from\_url("http://223.130.147.99:8000/list/", "out.pdf", options = options, configuration = config)**    return redirect('report\_list:list')  ...  **> 다양한 주제/목적의 보고서를 구성한다.**  > # vi report\_list/views.py  ...  ...  **def list2(request):**  ...  ...  **def list3(request):**  ...  ...  **> 각각의 보고서 양식들과 웹페이지 주소를 연결한다.**  > # vi report\_list/urls.py  urlpatterns = [  **path('', views.index, name='list1'),**  **path('list2/', views.list2, name='list2'),**  ...  ...  ]  **> 웹서버 메인 페이지에서 관리자 계정으로 접속하여 원하는 리포트 형식을 선택할 수 있게 메뉴를 구성한다.**  > # vi report\_main/templates/report\_main/index.html  {% extends 'base.html' %}  {% include 'title.html' %}  {% block content %}  {% load static %}  ...  ...  <li class="nav-item active has-child">  <a class="nav-link" href="">REPORT</a>  <ul class="child">  <li class="nav-item">  <a href="{% url 'report\_list:list' %}" class="nav-link">**REPORT 1**</a>  </li>  <li class="nav-item">  <a href="#" class="nav-link">**REPORT 2**</a>  </li>  <li class="nav-item">  <a href="#" class="nav-link">**REPORT 3**</a>  </li>  <li class="nav-item">  <a href="#" class="nav-link">**REPORT 4**</a>  </li>  </ul>  </li>    ...  ...  {% endblock content %}  **> 관리자 계정으로 로그인하여 메뉴를 통해 보고서2에 접근한다.**  > 223.130.147.99:8000/list2/ | |
| 부가설명 | 1. Elasticsearch 검색, 분석 기능을 활용한 docu.를 작성하여 리포트 구성 시 원하는 data를 얻는다.  2. 작성된 보고서를 DB화하여 관리하고, 필요시 언제나 확인할 수 있게 한다. |

5. ERD (데이터 모델링/모식도)