

수업 내용 정리 보고서

작성일: 2026년 7월 22일

챕터 구성

- Nmap 스캔 패턴 분석 및 Suricata 탐지 룰 설계
- Python 기반 네트워크 포트 스캐닝 도구 구현 및 최적화
- EIGRP 라우팅 프로토콜과 DUAL 알고리즘의 메커니즘
- 네트워크 보안 및 라우팅 환경 운영 점검 리스트

Nmap 스캔 패턴 분석 및 Suricata 탐지 룰 설계

네트워크 침투의 가장 초기 단계인 정찰(Reconnaissance) 과정에서 공격자는 **Nmap**과 같은 도구를 사용하여 대상 네트워크의 살아있는 호스트, 열려 있는 포트, 그리고 운영 중인 서비스의 종류를 파악합니다. 이러한 스캔 행위는 네트워크 보안 장비인 **Suricata(IDS/IPS)**에 의해 탐지될 수 있습니다. 효과적인 탐지를 위해서는 단순한 패킷의 존재 여부를 넘어, 스캐너가 생성하는 고유한 페이로드(Payload), 반복적인 행위 패턴, 그리고 그에 따른 응답(Response)의 상관관계를 분석하여 룰(Rule)을 설계해야 합니다.

1. UDP 스캔 탐지 패턴 분석 (-sU)

UDP(User Datagram Protocol)는 연결 지향적이지 않은 비연결형 프로토콜이기 때문에, TCP 스캔보다 탐지가 까다롭습니다. Nmap의 UDP 스캔은 크게 세 가지 관찰 패턴을 보입니다.

- 빈 UDP 패킷 반복:** 특정 포트를 확인하기 위해 데이터 크기가 0인(dsize:0) 패킷을 연속적으로 전송합니다.
- 고정 페이로드 사용:** 스캔 과정에서 특정 크기의 고정된 데이터(예: 40-byte)를 반복적으로 사용하는 특징이 있습니다. 이는 단순한 통신과 구분할 수 있는 매우 강력한 시그니처입니다.
- ICMP 응답 기반 탐지:** 닫힌 UDP 포트로 패킷을 보냈을 때, 대상 호스트가 응답하는 **ICMP Destination Port Unreachable (Type 3, Code 3)** 메시지를 추적하여 스캐너의 위치를 역으로 파악합니다.

UDP 탐지 룰 설계 예시

효율적인 운영을 위해 탐지 신뢰도(C Confidence)에 따라 룰을 단계별로 구성합니다.

SID	탐지 대상	핵심 조건	신뢰도
3001021	빈 UDP 반복	dsize:0, 30초 내 10회 이상	중간
3001022	Nmap 고정 40B 페이로드	dsize:40, 특정 Hex 패턴 일치	높음
3001023	ICMP 3/3 응답	icmp type:3, code:3, 30초 내 10회	보조

```
# SID 3001022: Nmap 고정 40-byte payload 탐지 룰 예시
alert udp any any -> any any ( dsize:40; content:"|3E EC E3 CA 00 00 00 00 00 00 00 02 00 BC 61 4E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00|"; offset:0; depth:40; detection_filter:track by_src,count 10,seconds 30; sid:3001022; rev:1;)
```

2. IP 프로토콜 및 TCP 플래그 스캔 탐지

공격자는 프로토콜 번호를 바꾸어가며 스캔하거나(IP Protocol Scan), TCP 헤더의 플래그(Flag) 조합을 조절하여 방화벽의 정책을 탐색합니다.

IP 프로토콜 스캔 (-sO)

IP 헤더의 프로토콜 필드를 조사하는 스캔입니다. 단순히 데이터 크기가 0인 패킷을 탐지할 경우, 정상적인 ACK 패킷과 충돌하여 **오탐(False Positive)**이 발생할 위험이 큼니다. 따라서 임계치(Threshold) 설정이 매우 중요합니다.

- 주의사항:** dsize:0 단독 사용 시 위험하므로, 동일 출발지에서 짧은 시간 내에 발생하는 대량의 패킷(예: 30초 내 90회)을 묶어 탐지해야 합니다.

ACK 및 Window 스캔 (-sA, -sW)

이 스캔들은 완전한 연결을 맺지 않고, 패킷의 플래그 상태에 따른 응답(ACK 또는 RST)을 통해 포트의 필터링 상태를 확인합니다. 탐지 전략은 '**발생량의 급증(Burst)**'에 초점을 맞춥니다.

SID	스캔 유형	탐지 로직	설정값(예시)
3001041	ACK sweep (ACK flag)	동일 출발지에서 대량의 ACK 패킷 발생	count 100, seconds 1
3001042	RST burst (RST flag)	동일 목적지로 대량의 RST 패킷 수신	count 100, seconds 1

3. 서비스 프로브 탐지 (Service Probing)

포트가 열려 있음을 확인한 후, 공격자는 해당 포트에서 어떤 서비스가 실행 중인지 확인하기 위해 특정 문자열이 포함된 요청을 보냅니다. 이는 단순한 네트워크 스캔을 넘어 애플리케이션 계층까지 침투하려는 시도입니다.

- nmaplowercheck:** HTTP 서비스에 GET /nmaplowercheck 요청을 보내 응답을 확인합니다. (SID 3001052)
- Encoded Trinity:** 인코딩된 URI 경로(예: /nice%20ports%2C/Tri%6Eity.txt%2ebak)를 사용하여 보안 장비의 우회를 시도합니다. (SID 3001051)

실무 팁: 향후 탐지 성능을 높이기 위해서는 단순 텍스트 매칭을 넘어 HTTP Method나 URI를 별도의 **Sticky Buffer**로 지정하여 정규화(Normalization)에 대한 내성을 강화해야 합니다.

4. 스캐너 구현 관점에서의 이해 (Python 기반)

Suricata가 탐지하는 패턴은 실제 스캐너가 어떻게 동작하는지를 이해할 때 더욱 명확해집니다. 파이썬을 이용한 포트 스캐너 구현 시, 성능과 정확도를 위해 다음과 같은 기술적 요소들이 적용됩니다.

- 멀티스레딩(Multi-threading):** 단일 스레드는 속도가 매우 느리므로 Thread 또는 ThreadPoolExecutor를 사용하여 수많은 포트를 동시에 검사합니다.
- 동기화(Synchronization):** 여러 스레드가 동시에 결과 파일에 기록할 때 데이터가 깨지지 않도록 Lock을 사용하여 파일 접근을 제어합니다.
- 타임아웃(Timeout) 설정:** 응답이 없는 포트에서 무한정 대기하는 것을 방지하기 위해 `settimeout()`을 사용하여 스캔 속도를 유지합니다.

```
# 스레드 풀을 이용한 효율적인 스캔 구현 예시
from concurrent.futures import ThreadPoolExecutor, as_completed
import socket

def ConnScan(Target_Host, Target_Port):
    try:
        # 타임아웃을 설정하여 스캔 효율성 증대
        with socket.create_connection((Target_Host, Target_Port), timeout=2):
            return f"[{Target_Host}:{Target_Port}] Open"
    except OSError:
        return None

# max_workers를 통해 스레드 개수를 제어하여 시스템 부하 조절
with ThreadPoolExecutor(max_workers=100) as executor:
    # 스캔 작업 제출 및 결과 수집 과정...
    pass
```

[종합] 탐지 전략 및 운영 가이드

성공적인 탐지 운영을 위해서는 '시그니처 → 행위 → 응답' 순으로 탐지의 강도를 높여가는 전략이 필요합니다.

탐지 정확도(Detection Accuracy) = 특이도(Specificity) × 시간창(Time-window) × 방향성(Directionality)

운영 우선순위 (Deployment Decision)

- 즉시 배포 (High Confidence):** 고유한 페이로드가 확인되는 룰 (예: Nmap 40B UDP payload, nmaplowercheck URI)은 오탐 가능성이 매우 낮으므로 즉시 차단 후보로 활용합니다.
- 환경별 튜닝 (Medium Confidence):** 빈 UDP 반복, ICMP 응답, ACK/RST 버스트 등은 네트워크 환경에 따라 정상 트래픽과 충돌할 수 있으므로 실제 트래픽을 관찰하며 임계치(count, seconds)를 재조정해야 합니다.
- 집중 검증 (Low Confidence):** IP 프로토콜 스캔이나 HTTP 400 응답 버스트는 광범위한 탐지가 가능하지만 오탐 위험이 높으므로, 경보(Alert) 수준에서 먼저 수집하며 신뢰도를 검증합니다.

실무 점검 체크리스트

- [] 탐지 룰의 임계치가 실제 네트워크의 정상적인 대량 트래픽과 충돌하지 않는가?
- [] UDP 스캔 탐지 시, 단순히 `dsize:0`만 보고 있지는 않은가? (반드시 반복 횟수와 결합 필요)
- [] TCP 플래그 탐지 시, `track by_src`(공격자 추적)와 `track by_dst`(피해자 관찰)를 목적에 맞게 구분하였는가?
- [] HTTP 프로브 탐지 시, 인코딩된 URI에 대한 대응이 고려되었는가?

Python 기반 네트워크 포트 스캐닝 도구 구현 및 최적화

네트워크 보안 진단의 기초가 되는 포트 스캐닝(Port Scanning)은 대상 호스트의 어떤 서비스 포트가 열려 있는지 확인하여 공격 표면(Attack Surface)을 분석하는 과정입니다. 본 장에서는 Python의 `socket` 모듈을 활용하여 기초적인 스캐너부터 성능 최적화된 멀티스레딩 스캐너까지 구현하는 과정과, 이러한 스캐닝 행위가 네트워크 보안 시스템(IDS/IPS)에서 어떻게 탐지되는지에 대해 학습합니다.

1. 포트 스캐닝의 기본 원리와 구현

포트 스캐닝의 핵심은 대상 IP와 포트 번호로 TCP 연결을 시도하여, 연결 성공 여부에 따라 포트의 상태를 판단하는 것입니다. Python에서는 `socket` 라이브러리를 통해 이를 구현할 수 있습니다.

TCP 연결 기반 스캐닝 (TCP Connect Scan)

가장 기본적인 방식은 `socket.connect()` 함수를 사용하는 것입니다. 대상 포트가 열려 있으면 연결이 성립되고, 닫혀 있으면 `OSError` 또는 `ConnectionRefusedError`와 같은 예외가 발생합니다.

- 작동 원리:** 3-Way Handshake 과정을 통해 완전히 연결을 맺는 방식입니다.
- 특징:** 구현이 매우 간단하지만, 연결 기록(Log)이 대상 서버에 남기 때문에 스텔스(Stealth)성이 낮습니다.

2. 스캐닝 도구의 단계별 최적화 과정

단순한 순차적 루프 방식의 스캐닝은 수만 개의 포트를 확인하는 데 시간이 너무 오래 걸립니다. 이를 해결하기 위해 단계별로 최적화를 진행하였습니다.

단계 1: 기본 멀티스레딩 구현 및 한계

threading 모듈을 사용하여 여러 포트를 동시에 스캔하도록 구현했습니다. 하지만 초기 구현에서는 다음과 같은 문제점이 발견되었습니다.

- **CPU 과점유**: 스레드 생성 속도가 제어되지 않아 while True 루프 내에서 CPU 사용량이 100%에 도달하는 현상이 발생했습니다.
- **데이터 경합(Race Condition)**: 여러 스레드가 동시에 result.txt 파일에 쓰기를 시도하면서 파일 내용이 섞이거나 누락될 위험이 있었습니다.
- **타임아웃 부재**: 응답이 없는 포트에서 프로그램이 무한정 대기하는 현상이 발생했습니다.

단계 2: 안정성 및 동기화 적용

위의 문제점을 해결하기 위해 다음과 같은 최적화 기법을 적용했습니다.

- **타임아웃 설정**: connskt.settimeout(2)를 추가하여 응답이 없는 포트는 2초 후 자동으로 포기하게 설정했습니다.
- **상호 배제(Mutual Exclusion)**: threading.Lock()을 도입하여 파일 쓰기 시 한 번에 하나의 스레드만 접근하도록 제어함으로써 데이터 무결성을 보장했습니다.
- **부하 조절**: time.sleep(0.01)을 추가하여 스레드 생성 루프의 CPU 점유율을 낮추었습니다.
- **데이터 타입 준수**: Python 3의 특성에 맞게 send(b'hello')와 같이 데이터를 바이트(bytes) 형태로 전송하도록 수정했습니다.

단계 3: ThreadPoolExecutor를 이용한 고성능 구현

수동으로 스레드를 생성하고 관리하는 대신 concurrent.futures.ThreadPoolExecutor를 사용하여 효율적인 스레드 풀(Thread Pool) 기반 스캐너를 구현했습니다.

최적화 핵심: max_workers 옵션을 통해 동시에 실행될 스레드 수를 제한함으로써 시스템 자원 고갈을 방지하고, as_completed()를 통해 완료된 스캔 결과부터 즉시 처리하여 응답성을 높였습니다.

3. 실무적 기능 확장: 입력 검증 및 범위 지정

실제 도구로 활용하기 위해 사용자로부터 IP 주소와 포트 범위를 입력받는 기능을 추가했습니다. 이때 잘못된 입력으로 인한 프로그램 충돌을 막기 위해 정규 표현식(Regular Expression)과 ipaddress 모듈을 활용했습니다.

- **IP/도메인 검증**: re.compile을 통해 숫자 형식의 IP 주소인지, www.로 시작하는 도메인 형태인지 구분하여 입력값을 필터링합니다.
- **포트 범위 지정**: 0-65535와 같은 형태의 입력을 받아 int형으로 변환 후 루프를 돌리는 유연한 범위를 설정했습니다.

4. 보안 관점에서의 분석: 스캐닝 탐지와 대응

Python으로 구현한 스캐너가 실제로 작동할 때, 네트워크 보안 장비(예: Suricata)는 다음과 같은 패턴을 근거로 이를 탐지합니다. 개발자는 이를 통해 자신의 도구가 어떤 흔적을 남기는지 이해해야 합니다.

탐지 시그니처 분석

탐지 대상	주요 탐지 조건(Suricata 룰 기준)	분석 및 의미
빈 UDP 스캔	dsizе:0 + count 10, seconds 30	데이터가 없는 UDP 패킷이 짧은 시간 내 반복적으로 발생할 때 탐지
Nmap 고유 패턴	특정 40바이트 페이로드(Hex 값)	도구 고유의 시그니처(Payload)를 기반으로 정확하게 탐지
ACK/Window 스캔	flags:A 또는 flags:R burst	비정상적으로 많은 ACK 요청이나 RST 응답이 발생할 때 탐지
서비스 프로브	GET /nmaplowercheck 등 특정 URI	단순 포트 확인을 넘어 서비스 버전을 확인하려는 시도를 문자열 기반으로 탐지

탐지 회피와 탐지 정확도의 관계

보안 시스템은 단순히 패킷 하나를 보는 것이 아니라 '시그니처 → 행위 → 응답'의 순서로 신뢰도를 높입니다. 예를 들어, 포트 스캔 요청(Request)과 그에 따른 ICMP Destination Port Unreachable 응답을 하나의 사건으로 묶어 분석하면 오탐(False Positive)을 줄이고 정확도를 높일 수 있습니다.

5. 요약 및 점검 리스트

포트 스캐닝 도구를 구현하고 운영할 때 반드시 확인해야 할 사항입니다.

- **성능 점검**: 스레드 수(max_workers)를 너무 높게 설정하여 대상 서버에 DoS(서비스 거부) 공격과 같은 영향을 주지는 않는가?
- **안정성 점검**: 타임아웃 설정이 되어 있어 네트워크 지연 상황에서도 프로그램이 멈추지 않는가?
- **동기화 점검**: 멀티스레드 환경에서 파일 쓰기 시 Lock을 사용하여 결과와 데이터가 손상되지 않는가?
- **윤리 및 법적 준수**: 허가받지 않은 대상에 대한 스캐닝은 공격 행위로 간주될 수 있으므로, 반드시 테스트 환경(Sandbox)에서 수행하고 있는가?

EIGRP 라우팅 프로토콜과 DUAL 알고리즘의 메커니즘

네트워크의 효율적인 데이터 전달을 위해 라우터들은 서로 최적의 경로 정보를 교환합니다. 이번 학습에서는 시스코(Cisco)의 독자적인 라우팅 프로토콜인 EIGRP(Enhanced Interior Gateway Routing Protocol)와, 루프(Loop) 없는 빠른 경로 계산을 보장하는 핵심 엔진인 DUAL(Diffusing Update Algorithm) 알고리즘을 심도 있게 다룹니다.

1. EIGRP의 정의 및 프로토콜 특성

EIGRP는 하나의 자율 시스템(AS, Autonomous System) 내부에서 경로 정보를 교환하는 IGP(Interior Gateway Protocol)입니다. 흔히 Distance Vector와 Link-State의 장점을 결합한 **Advanced Distance Vector** 프로토콜로 분류됩니다.

- 전송 프로토콜:** IP Protocol 88을 사용합니다.
- 멀티캐스트 주소:** 이웃 라우터를 탐색할 때 224.0.0.10 주소를 사용합니다.
- 핵심 차별점:** 단순한 거리(Distance)와 방향(Vector) 정보만 전달하는 기존 방식과 달리, DUAL 알고리즘을 통해 매우 빠른 수렴(Convergence) 속도와 루프 방지 능력을 갖추고 있습니다.

라우팅 프로토콜은 운용 범위에 따라 IGP와 EGP로 나뉩니다. EIGRP는 내부 네트워크 최적화에 집중하는 IGP이며, 서로 다른 AS 간의 경로를 교환하는 BGP와 같은 EGP와는 그 목적과 동작 방식이 엄격히 구분됩니다.

2. EIGRP의 데이터 관리: 3대 핵심 테이블

EIGRP는 경로 정보를 체계적으로 관리하기 위해 세 가지 독립적인 데이터베이스(테이블)를 유지합니다. 이 테이블들의 관계를 이해하는 것이 EIGRP 운영의 시작입니다.

- Neighbor Table (이웃 테이블):** 현재 인접 관계를 형성하여 통신 중인 이웃 라우터들의 목록입니다. `show ip eigrp neighbors` 명령어로 확인합니다.
- Topology Table (토폴로지 테이블):** 이웃으로부터 학습한 모든 가능한 경로 정보를 담고 있습니다. 현재 사용 중인 최적 경로뿐만 아니라 백업 경로 정보도 포함됩니다. `show ip eigrp topology` 명령어로 확인합니다.
- Routing Table (라우팅 테이블):** 토폴로지 테이블에서 선정된 최적의 경로만을 선별하여 실제 패킷 전달에 사용하는 테이블입니다. `show ip route eigrp` 명령어로 확인합니다.

3. 경로 결정의 기준: 복합 메트릭(Composite Metric)

EIGRP는 단순히 '몇 번 거쳤는가(Hop Count)'를 따지는 RIP와 달리, 다양한 경로 품질 요소를 조합하여 **메트릭(Metric)**을 계산합니다. 이를 통해 훨씬 정교한 경로 선택이 가능합니다.

기본적으로 **대역폭(Bandwidth)**과 **지연 시간(Delay)**을 주요 요소로 사용하며, 계산식은 다음과 같습니다.

$$\text{Metric} = 256 \times (\text{Bandwidth 항목} + \text{Delay 항목})$$

이때 각 요소의 가중치를 조절하는 값이 **K 값(K-values)**입니다. 실무에서는 안정성을 위해 기본 설정인 K1(Bandwidth)과 K3(Delay)만을 사용하는 것이 일반적입니다.

항목	기본 K 값	역할
Bandwidth (대역폭)	K1 = 1	경로 내 최소 대역폭 반영
Load (부하)	K2 = 0	링크의 현재 부하 상태 반영
Delay (지연)	K3 = 1	경로의 누적 지연 시간 반영
Reliability (신뢰성)	K4, K5 = 0	링크의 오류 발생 빈도 반영

4. 네이버 형성 및 패킷 통신 메커니즘

라우터가 이웃(Neighbor) 관계를 맺기 위해서는 몇 가지 조건이 반드시 일치해야 합니다. 만약 조건이 맞지 않으면 네이버 관계가 형성되지 않으므로 troubleshooting 시 가장 먼저 확인해야 할 포인트입니다.

- 동일한 AS 번호:** 서로 다른 AS 번호를 가진 라우터는 통신하지 않습니다.
- 동일 서브넷:** 직접 연결된 인터페이스의 서브넷이 일치해야 합니다.
- K 값의 일치:** 계산 방식이 같아야 하므로 K 값이 동일해야 합니다.
- 인증 정보 일치:** 보안 설정 시 Key-chain 정보가 일치해야 합니다.

이웃 관계가 형성되면, EIGRP는 다섯 가지 종류의 패킷을 통해 정보를 교환합니다. 특히 Update, Query, Reply 패킷은 신뢰성 있는 전송을 위해 **RTP(Reliable Transport Protocol)**를 통해 ACK(수신 확인)을 주고받습니다.

5. DUAL 알고리즘: 루프 없는 빠른 경로 계산

EIGRP의 가장 강력한 특징은 **DUAL 알고리즘**을 통한 신속한 장애 복구입니다. DUAL은 'Feasibility Condition(FC)'이라는 조건을 통해 경로의 루프 가능성을 수학적으로 차단합니다.

핵심 용어 정의

- **FD (Feasible Distance):** 현재 라우터에서 목적지까지의 전체 메트릭(최적값).
- **RD (Reported Distance):** 이웃 라우터가 나에게 알려준 목적지까지의 메트릭.
- **Successor (서세서):** DUAL이 선정한 현재의 최적 경로.
- **Feasible Successor (FS, 피저블 서세서):** 최적 경로는 아니지만, 루프가 없음이 증명된 백업 경로.

Feasibility Condition (FC): “*이웃이 보고한 거리(RD)가 나의 현재 최적 거리(FD)보다 작아야 한다*”는 조건입니다. 이 조건이 만족되면 해당 이웃은 루프를 형성하지 않는다는 것이 보장되므로, 장애 발생 시 즉시 백업 경로로 전환할 수 있습니다.

장애 발생 시 동작 흐름

경로에 문제가 생겼을 때, EIGRP는 상태에 따라 두 가지 방식으로 대응합니다.

1. **FS가 존재하는 경우:** 즉시 새로운 Successor로 승격시켜 통신 중단을 최소화합니다. (Passive 상태 유지)
2. **FS가 없는 경우:** 경로를 **Active** 상태로 전환하고, 주변 이웃들에게 새로운 경로를 묻는 **Query**를 전송합니다. 모든 이웃으로부터 **Reply**를 받은 후에야 다시 안정적인 **Passive** 상태로 돌아갑니다.

이 과정에서 응답이 오지 않아 경로 계산이 지연되는 현상을 **SIA(Stuck In Active)**라고 합니다. 이는 네트워크 부하나 불안정한 링크로 인해 발생하며, 결국 해당 이웃과의 관계가 끊어질 수 있는 위험한 상태입니다.

6. 경로 최적화 및 제어 기법

네트워크 설계 목적에 따라 경로의 우선순위를 조정하거나 부하를 분산할 수 있습니다.

- **UCMP (Unequal Cost Multi-Path):** variance 명령어를 사용하여 메트릭이 다른 경로라도 특정 조건($FD \leq \text{Successor FD} \times \text{variance}$)을 만족하면 동시에 사용하여 부하를 분산할 수 있습니다.
- **Offset-list:** 특정 경로의 메트릭을 인위적으로 높여(주로 Delay 항목에 추가) 해당 경로를 덜 선호하게 만듭니다.
- **Manual Summary:** 세부 경로를 요약하여 라우팅 테이블 크기를 줄이고 네트워크 안정성을 높입니다. 이때 요약 경로에 포함되지만 상세 경로가 없는 트래픽이 루프를 돌지 않도록 **Null0** 인터페이스로 드랍하는 것이 중요합니다.
- **Filtering:** distribute-list나 route-map을 사용하여 특정 경로가 광고되거나 수신되는 것을 제어합니다.

실무 및 시험 대비 점검 리스트

EIGRP를 운영하거나 관련 시험을 준비할 때, 다음 항목들을 반드시 점검하십시오.

1. **네이버가 맺어지지 않을 때:** AS 번호, 서브넷, K 값, 인증(Key-chain) 설정을 순차적으로 확인하십시오.
2. **수렴 속도가 느릴 때:** Feasible Successor(FS)가 존재하도록 토폴로지가 구성되어 있는지, 혹은 SIA 현상이 발생하고 있지는 않은지 확인하십시오.
3. **경로가 의도치 않게 선택될 때:** 메트릭 계산 방식(Bandwidth/Delay)과 Offset-list 설정, 혹은 Redistribution 시 부여된 Seed Metric을 검토하십시오.
4. **요약 경로 설정 시:** 요약 경로(Summary Address)가 생성될 때 자동으로 만들어지는 Null0 경로가 루프 방지 역할을 하고 있는지 확인하십시오.

네트워크 보안 및 라우팅 환경 운영 점검 리스트

본 보고서는 네트워크 보안의 핵심인 침입 탐지(Intrusion Detection)와 네트워크 가용성을 결정짓는 라우팅 프로토콜(Routing Protocol)의 안정적 운영을 위한 점검 항목을 정리한다. 스캐닝 도구의 동작 원리를 이해하여 탐지 룰을 설계하는 방법과, EIGRP와 같은 고급 거리 벡터(Advanced Distance Vector) 프로토콜의 경로 계산 메커니즘을 점검하는 절차를 포함한다.

1. 침입 탐지 시스템(IDS) 운영 및 스캔 패턴 점검

네트워크 스캐닝(Scanning)은 공격 전 단계에서 타겟의 활성 포트와 서비스를 확인하는 행위이다. 이를 효과적으로 차단하기 위해서는 단순한 시그니처 탐지를 넘어, 행위(Behavior)와 응답(Response)의 상관관계를 분석하는 체계적인 접근이 필요하다.

1.1 스캔 유형별 탐지 전략

탐지의 정확도는 '**특이도(Specificity) × 시간창(Time Window) × 방향성(Directionality)**'의 조합으로 결정된다. 단순한 패킷 하나가 아닌, 동일 출발지(Source)에서 특정 시간 동안 발생하는 반복적인 패턴을 식별해야 한다.

- **UDP 스캔 탐지:** 빈 UDP 패킷(dsize:0)의 반복, 또는 Nmap 특유의 고정된 40-byte 페이로드(Payload)를 확인한다. 또한, 닫힌 포트에 대한 응답인 **ICMP Destination Port Unreachable**(Type 3, Code 3) 패킷이 스캐너에게 돌아오는지를 관찰하여 스캔 여부를 확증한다.
- **ACK/Window 스캔 탐지:** TCP 플래그가 ACK(A) 또는 RST(R)로 구성된 패킷이 단시간에 대량 발생(Burst)하는 경우를 점검한다. 이는 네트워크의 필터링 상태를 확인하려는 시도일 가능성이 높다.
- **서비스 프로브(Service Probing) 탐지:** 특정 도구(예: nmaplowercheck)가 사용하는 고유한 HTTP URI나 인코딩된 경로(예: Trinity Probe)를 시그니처로 활용한다.

1.2 탐지 룰 튜닝 및 오탐(False Positive) 관리

모든 탐지 룰이 즉시 차단(Drop) 대상이 될 수는 없다. 운영 환경에서는 다음과 같은 단계적 배포 전략을 권고한다.

1. **우선순위 배포**: 40-byte 고정 페이로드나 특정 URI와 같이 오탐 가능성이 매우 낮은 **고유 시그니처**를 우선적으로 차단한다.
2. **임계치(Threshold) 조정**: 빈 UDP 반복이나 ACK/RST 버스트의 경우, 실제 트래픽 패턴을 분석하여 **count**와 **seconds** 값을 재조정한다. 예를 들어, 정상적인 대용량 트래픽이 RST 응답을 유발하지 않는지 확인해야 한다.
3. **상관분석 활용**: 단일 패킷 경보보다는 '**요청(Request) + 응답(ICMP/RST/HTTP 400)**'을 하나의 사건(Event)으로 묶어 탐지 신뢰도를 높인다.

2. 스캐닝 도구 구현 및 자원 관리 점검

포트 스캐닝 도구를 직접 구현하거나 분석할 때는 멀티스레딩(Multi-threading) 환경에서의 자원 경합과 효율적인 스레드 관리가 핵심이다.

2.1 효율적인 스레드 모델링

단순히 스레드를 생성하는 것을 넘어, 시스템 부하를 방지하고 데이터 무결성을 보장하는 설계가 필요하다.

- **자원 경합 방지(Locking)**: 여러 스레드가 동시에 파일(예: result.txt)에 결과를 기록할 경우 데이터가 깨질 수 있으므로, **Lock** 객체를 사용하여 쓰기 작업을 동기화해야 한다.
- **CPU 점유율 관리**: 스레드 개수가 임계치(예: 500개)를 초과할 경우, `time.sleep()`을 통해 CPU가 100% 점유되는 것을 방지하는 제어 로직이 포함되어야 한다.
- **고급 스레드 관리**: 실무적으로는 `ThreadPoolExecutor`를 사용하는 것이 권장된다. 이는 스레드 풀을 통해 작업(Task)을 관리하므로, 개별 스레드를 직접 생성하고 관리하는 것보다 훨씬 안정적이고 효율적이다.

3. 라우팅 환경(EIGRP/DUAL) 운영 안정성 점검

네트워크 경로의 안정성은 라우팅 프로토콜이 장애를 얼마나 빠르게 감지하고, 루프(Loop) 없이 대체 경로를 계산하느냐에 달려 있다. EIGRP 환경에서는 **DUAL(Diffusing Update Algorithm)**의 동작 상태를 점검하는 것이 필수적이다.

3.1 DUAL 알고리즘 및 경로 상태 점검

EIGRP는 **Feasibility Condition(FC)**을 통해 루프 없는 경로를 보장한다. 즉, 이웃이 광고한 거리(Reported Distance, RD)가 현재 나의 최적 거리(Feasible Distance, FD)보다 작아야 안전한 백업 경로로 간주한다.

- **Successor**: 현재 라우팅 테이블에 설치된 최적 경로.
- **Feasible Successor (FS)**: FC를 만족하여 즉시 사용 가능한 백업 경로.

운영자는 라우터의 상태가 **Passive**(정상 계산 완료)인지, **Active**(경로 재계산 중)인지를 반드시 확인해야 한다. 만약 경로 장애 시 FS가 없다면 라우터는 Query를 전송하며 Active 상태로 전환된다. 이때 적절한 응답을 받지 못하면 **SIA(Stuck In Active)** 상태에 빠져 인접 관계가 끊어질 수 있으므로 주의가 필요하다.

3.2 경로 제어 및 최적화 설정 점검

네트워크 트래픽 엔지니어링을 위해 다음과 같은 설정값이 의도대로 동작하는지 점검한다.

점검 항목	목적 및 동작 원리	주의사항
Offset-list	특정 경로의 Metric(주로 Delay)을 인위적으로 증가시켜 경로 우선순위 조정.	경로를 차단하는 것이 아니라 선호도를 낮추는 것임.
Manual Summary	인터페이스 단위로 경로를 요약하여 라우팅 테이블 크기를 줄이고 업데이트 범위를 축소.	요약 경로 생성 시 Null0 경로가 생성되어 루프를 방지하는지 확인.
Variance	서로 다른 Metric을 가진 경로도 동시에 사용(UCMP)할 수 있도록 허용.	반드시 Feasibility Condition을 만족하는 경로에만 적용됨.
Distribute-list	ACL이나 Route-map을 사용하여 특정 경로의 광고/수신을 필터링.	Route-map 사용 시 마지막의 암묵적 Deny를 고려하여 구성.

[종합] 운영 환경 핵심 점검 체크리스트

보안 운영(Security Operations)

- [] Suricata 룰의 임계치(Count/Seconds)가 실제 트래픽 환경에 맞춰 튜닝되었는가?
- [] 고유 페이로드(40-byte 등)를 기반으로 한 차단 룰이 우선 적용되고 있는가?
- [] 스캔 탐지 시 단순 패킷 발생뿐만 아니라 ICMP/RST 등의 응답 상관관계를 확인하는가?

라우팅 운영(Routing Operations)

- [] `show ip eigrp neighbors`를 통해 Hello/Hold 타이머와 인접 관계를 확인하였는가?
- [] `show ip eigrp topology`에서 주요 경로가 **Passive** 상태를 유지하고 있는가?
- [] 장애 발생 시 FS(Feasible Successor)가 존재하여 빠른 수렴(Convergence)이 가능한가?
- [] 대규모 네트워크에서 Query 전송 범위가 과도하여 **SIA** 발생 위험이 있지는 않은가?

- [] 인증(Key-chain) 및 필터링(Distribute-list)이 보안 정책에 맞게 설정되었는가?

추가 실습 화면 데이터