2025 월간 위협 분석 보고서

웹 로그 분석을 통한 악성행위 판단 방안

PLAINBIT 사이버위협대응센터
인텔리전스팀

※ 본 보고서는 2025년 8월 국가사이버안보센터(NCSC) 합동분석협의체를 통해 발간되었습니다.

2025 기술 가이드

웹 로그 분석을 통한 악성행위 판단 방안

Contents

01 개요

웹 서버는 조직의 내부 시스템과 외부 네트워크가 맞닿는 접속 지점(Entry Point)으로 대부분의 사이버 공격이 최초로 시도되는 대상 중 하나이다. 특히, HTTP 기반 통신은 인증, 데이터 송수신, API 연동 등 다양한 서비스 흐름을 포함하고 있어 공격자는 이 영역을 활용해 취약점 스캐닝, 인가 우회, 명령 실행 등 다양한 위협 행위를 시도한다.

이러한 상호작용은 웹 서버의 액세스 로그 및 에러 로그에 기록된다. 웹 로그에는 요청 IP, 경로, 파라미터, 응답 코드, 응답 시간 등의 정보가 포함되어 있으며, 이를 기반으로 공격자의 정찰 행위부터 침투 및 내부 이동에 이르는 시퀀스를 추적할 수 있다.

그러나 침해사고 분석을 전문적으로 하는 경우를 제외하곤, 실무 영역에선 다음과 같은 어려움이 존재한다.

- 웹 로그는 정상 트래픽과 공격 트래픽이 혼재되어 있어 공격식별이 어려움
- 기본 설정만으로는 공격 분석에 필요한 정보가 불충분하게 기록되는 경우가 발생
- 로그 분석은 단순 검색 수준을 넘어 공격 시나리오 재구성과 이상 행위 식별 능력이 요구

이 문서는 위와 같은 실무적 어려움을 해결하고자 작성된 웹 로그 기반 위협 식별 및 분석 방안이다. 단순히 로그 형식을 설명하는 수준을 넘어서, 공격 유형별 분석 방안, 명령어 기반 필터링 기법 등을 제시하며, 이를 통해 웹 서버를 통한 공격 여부를 초기에 파악하여 피해 최소화, 대응 시간 확보, 피해 범위 파악 등의 효과를 확보할 수 있는 기반이 될 것이다.

02 웹 로그의 구조와 수집 방식

1. 웹 로그(Weblog)

웹 로그(Weblog)는 웹 서버에서 발생하는 요청과 응답에 대한 상세한 정보를 기록한 데이터이다. 사용자의 접속 시도, 요청한 리소스, HTTP 메서드, 응답 코드, 사용자 에이전트 등 다양한 정보가 포함된다. 이 로그는 정상적인 운영 상태의 추적뿐만 아니라, 이상 행위 또는 공격 시도의 단서를 파악하는 데 매우 중요한 역할을 한다.

웹 로그는 웹 서버의 종류에 따라 형식이 다르며, 대표적으로 Apache HTTP Server, Nginx, Microsoft IIS 등이 있다. 본 문서에서는 Apache와 IIS 로그를 중심으로 분석 방안을 설명한다.

2. Apache 웹 서버

Apache HTTP Server는 Apache Software Foundation에서 개발 및 유지 관리하는 오픈소스 크로스플랫폼 웹 서버 소프트웨어이다. 오랜 기간 전 세계 웹 서버 시장에서 가장 높은 점유율을 차지해 왔으며, 현재까지도 널리 사용되고 있다. 높은 안정성과 성능, 풍부한 기능, 뛰어난 확장성을 특징으로 한다. 유연성은 로그 설정에도 그대로 적용되어, 관리자가 원하는 형식과 수준으로 로그를 기록하고 관리할 수 있는 강력한 기능을 제공한다.

(1) 로그 파일 종류 및 경로

Apache는 주로 두 가지 종류의 로그를 생성하며, 기본 경로는 운영체제 및 설치 방식에 따라 다르다.

① Access Log (접근 로그)

Access Log	서버에 들어오는 모든 HTTP 요청을 기록한다. 누가, 언제, 무엇을 요청했고, 서버가 어떻게 응답했는지 알 수 있다.	
파일명	access.log 또는 access_log (loglotate가 설정된 경우 일시가 접두/접미어로 붙을 수 있음)	
경로	Debian / Ubuntu	/var/log/apache2/
	RHEL / CentOS	/var/log/httpd/

② Error Log (오류 로그)

Error Log	서버 운영 중 발생하는 오류나 진단 정보를 기록한다. 설정 오류, 스크립트 실행 오류, 클라이언트의 비정상적인 요청 종료 등을 파악하는 데 중요하다.
파일명	error.log 또는 error_log (loglotate가 설정된 경우 일시가 접두/접미어로 붙을 수 있음)
경로	Access Log와 동일 경로

(2) 로그 포맷

Apache는 httpd.conf 또는 apache2.conf 설정 파일의 LogFormat 지시어를 사용하여 로그 형식을 자유롭게 정의할 수 있다.
기본 지시어로는 "Common Log Format"과 "Combined Log Format"으로 설정되어 있다.

① Common Log Format (CLF)
가장 기본적인 형식으로 다음과 같이 정의된다.

형식	"%h %l %u %t \"%r\" %>s %b" common
예시	127.0.0.1 - - [10/Oct/2023:10:55:36 +0000] "GET /index.html HTTP/1.1" 200 2326
필드	**설명**
%h	원격 호스트 (클라이언트 IP 주소)
%l	원격 로그 이름 (보통 -로 표시)
%u	인증된 사용자 이름 (보통 -로 표시)
%t	요청 시간 (예: [10/Oct/2023:10:55:36 +0000])
%r	요청 라인 (예: "GET /index.html HTTP/1.1")
%>s	최종 상태 코드 (예: 200, 404)
%b	응답 크기 (바이트 단위)

② Combined Log Format

CLF에 Referer와 User-Agent 정보가 추가된, 더 상세한 형식이다.

형식	LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
예시	192.168.1.1 - - [11/Jul/2024:14:20:11 +0900] "GET /index.html HTTP/1.1" 200 1234 "http://example.com" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) ..."156
추가 필드	**설명**
%{Referer}i	사용자가 이 페이지로 오기 전에 머물렀던 이전 페이지 주소
%{User-Agent}i	클라이언트의 브라우저 정보

③ Error Log Format

오류 로그는 접근 로그와 같이 형식을 자유롭게 정의하기보다는, LogLevel 지시어를 통해 기록되는 메시지의 심각도 수준을 제어하는 것이 일반적이다. 로그의 각 항목은 서버의 문제 해결에 중요한 정보를 제공한다.

형식	[Timestamp] [Module:Level] [ProcessID:ThreadID] [Client IP:Port] Message
예시	[Tue Jul 11 14:30:00.123456 2024][core:error] [pid 12345:tid 67890] [client 192.168.1.2:12345] AH00124: Request exceeded the limit of 10 internal redirects ⋯ necessary.
추가 필드	**설명**
[Timestamp]	이벤트가 발생한 정확한 시간
[Module:Level]	로그 메시지를 생성한 모듈(예: core, mpm_prefork, proxy_http 등) 로그의 심각도 수준(예: debug, info, notice, warn, error, crit, alert 등)
[ProcessID: ThreadID]	이벤트를 처리한 서버 프로세스 ID와 스레드 ID. 시스템 문제를 추적하는 데 사용
[Client IP:Port]	오류를 유발한 요청을 보낸 클라이언트의 IP 주소와 포트 번호
Message	실제 오류 메시지 내용 AH로 시작하는 코드는 Apache 고유의 오류 코드로

3. IIS 웹 서버

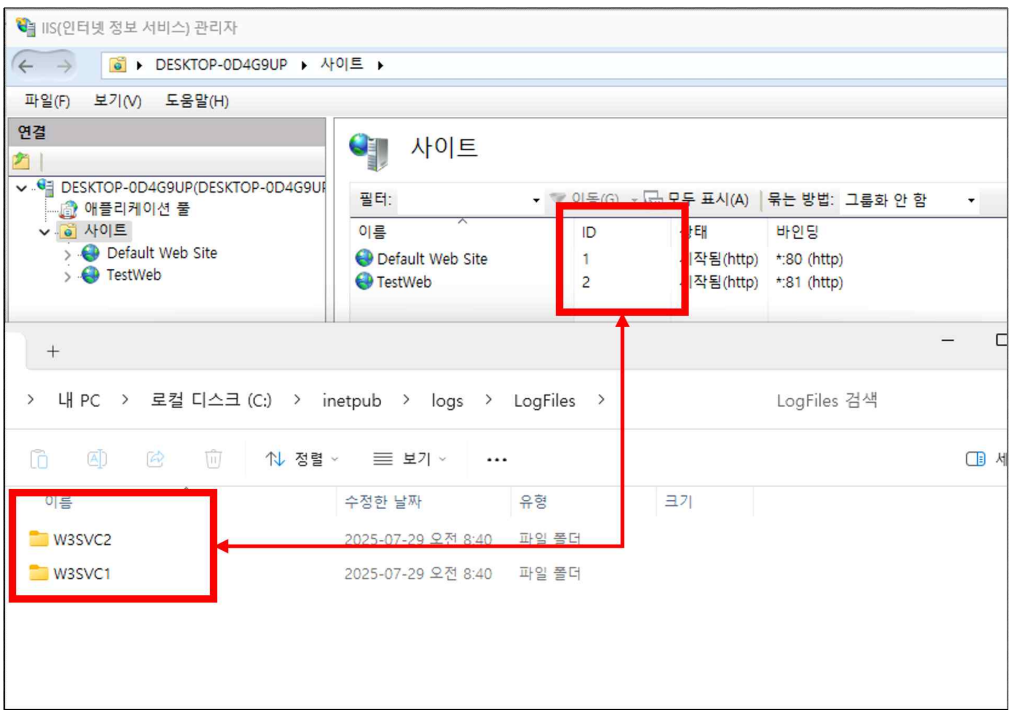
IIS(Internet Information Services)는 Microsoft Windows Server에 기본 포함된 웹 서버로, Windows 환경과 높은 통합성을 제공한다. Active Directory와 연동하여 사용자 인증 및 권한 관리를 중앙 집중화할 수 있으며, Windows 인증 기반의 보안 구성이 용이하다. 관리자는 GUI 기반의 IIS Manager를 통해 웹 사이트 설정, 인증 방식, 로깅 등을 직관적으로 제어할 수 있다. 또한 W3C 로그 포맷을 기반으로 상세한 요청 정보를 기록하며, .NET 기반 웹 애플리케이션의 호스팅에 최적화되어 있다.

(1) 로그 파일 및 경로

IIS 로그 파일은 기본적으로 사이트별로 생성되며, 경로는 다음과 같다.

로그 경로	%SystemDrive%\inetpub\logs\LogFiles\W3SVC<사이트ID>\
설명	• <사이트ID>: 각 웹 사이트에 부여된 고유 번호 (예: W3SVC1, W3SVC2) • 파일명: u_exYYMMDD.log 형식으로 매일 새로운 로그 파일이 생성 • Access, Error 로그 구분 없이 한 파일에 저장

가상 디렉터리가 다수일 경우, IIS 관리자의 사이트 메뉴에서 각 사이트 ID를 통해 로그가 저장되어있는 디렉터리를 식별할 수 있다.



(2) 로그 포맷

형식	date time s-ip cs-method cs-uri-stem cs-uri-query s-port cs-username c-ip cs(User-Agent) cs(Referer) sc-status sc-substatus sc-win32-status time-taken
예시	2024-07-11 05:30:15 172.20.1.10 GET /index.html - 80 - 192.168.10.5 Mozilla/5.0… http://previous.com/ 200 0 0 15
필드	**설명**
date	날짜
time	시간 (UTC 기준)
s-ip	서버 IP 주소
cs-method	요청 방식 (예: GET, POST))
cs-uri-stem	요청 URI (경로)
cs-uri-query	쿼리 문자열 (예: id=1)
s-port	서버 포트
cs-username	인증된 사용자 이름
c-ip	클라이언트 IP 주소
cs(User-Agent)	사용자 에이전트
cs(Referer)	이전 페이지 주소
sc-status	프로토콜 상태 코드 (예: 200, 404)
sc-substatus	하위 상태 코드
sc-win32-status	Win32 상태 코드
time-taken	time-taken

4. 웹 로그 수집 방안

로그를 분석 시스템으로 수집함으로써, 관리자 또는 보안 담당자는 개별 서버에 접근하지 않고도 전체 인프라의 행위 흐름을 파악할 수 있으며, 빠르고 체계적인 위협 탐지 및 대응이 가능해진다. 웹 로그를 수집하는 방법은 다음과 같은 방법들을 활용할 수 있다.

(1) 에이전트(Agent) 기반 수집

각 웹 서버에 로그 수집 에이전트(예: Filebeat, Fluentd, Splunk Universal Forwarder)를 설치하여 로그 파일을 실시간으로 감시하고 중앙 로그 서버로 전송하는 방식이다. 실시간 분석에 가장 효과적이다.

- 장점: 실시간 수집, 안정적인 전송, 다양한 로그 형식 처리 가능
- 단점: 모든 서버에 에이전트 설치와 관리 필요

(2) 스크립트 기반 수집

Cron(Linux)이나 작업 스케줄러(Windows)를 사용하여 특정 시간에 로그 파일을 읽어 중앙 서버로 전송하는 스크립트(예: Shell Script, Python, PowerShell)를 실행하는 방식이다.

- 장점: 에이전트 설치 없이 간단하게 구현 가능
- 단점: 실시간 수집이 어렵고, 스케줄링 간격 동안의 로그는 지연되어 수집될 수 있음

(3) Syslog 전송

웹 서버가 로그를 파일에 기록하는 대신, Syslog 프로토콜을 사용하여 네트워크를 통해 직접 중앙 로그 서버로 전송하는 방식이다. Apache는 mod_syslog 모듈을 통해 이를 지원할 수 있다.

- 장점: 표준 프로토콜을 사용하여 호환성이 높고, 실시간 전송 가능
- 단점: 네트워크 부하가 발생할 수 있으며, UDP 전송 시 로그 유실 가능성 존재

03 웹 로그를 통한 악성 행위 판단

수집된 웹 로그를 분석하여 악성 행위를 탐지하는 것은 보안 관제 및 침해사고 대응의 핵심적인 활동이다. 공격자는 다양한 기법을 사용하여 시스템을 공격하며, 일부 공격은 웹 로그에 흔적을 남긴다. 해당 흔적을 통해 공격 인지, 공격 영향(Impact) 파악 등을 판단할 수 있다.

(1) 웹 스캔 및 정찰(Reconnaissance) 공격 탐지

공격자는 본격적인 공격에 앞서, 웹 사이트의 구조를 파악하고 취약점을 찾기 위해 자동화된 스캐닝 도구를 사용한다. 이때 공격 방법에 따른 특징이 웹 로그에 기록되는데, 다음과 같은 탐지 지표로서 해당 특징을 활용할 수 있다.

탐지 지표	① 짧은 시간 동안 특정 IP 주소에서 대량의 요청이 발생하는 경우 ② 존재하지 않는 파일이나 디렉터리에 대한 요청(404 Not Found 응답)이 다수 발생하는 경우 ③ User-Agent 필드에 sqlmap, nmap, Nikto, 등과 같은 스캐너 도구의 이름이 포함된 경우 ④ 관리자 페이지(예: /admin, /manager/html)나 설정 파일 (예: /.git/config, /.env)에 대한 무단 접근 시도

로그 예시 (Apache)

```
242.41.4.1 - - [11/Jul/2024:10:00:05 +0900] "GET /admin.php HTTP/1.1" 404 499 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
242.41.4.1 - - [11/Jul/2024:10:00:06 +0900] "GET /config.json HTTP/1.1" 404 504 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
242.41.4.1 - - [11/Jul/2024:10:00:07 +0900] "GET /.git/config HTTP/1.1" 404 411 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
242.41.4.1 - - [11/Jul/2024:10:00:08 +0900] "GET /.git/config.bak HTTP/1.1" 404 411 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
242.41.4.1 - - [11/Jul/2024:10:00:08 +0900] "GET /.git/configure HTTP/1.1" 404 412 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
242.41.4.1 - - [11/Jul/2024:10:00:09 +0900] "GET /.git/conf HTTP/1.1" 404 411 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
242.41.4.1 - - [11/Jul/2024:10:00:09 +0900] "GET /.git/config.txt HTTP/1.1" 404 500 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
```

로그 예시를 분석해보면, 공격자 IP(242.41.4.1)가 Nmap Scripting Engine을 사용하여 웹 서버의 구조를 파악하고 민감한 설정 파일을 찾으려는 정찰 활동으로 판단할 수 있다.
다음과 같은 단계로 분석을 수행할 수 있다.

① 공격자 정보 식별

- 공격 IP: 242.41.4.1
- 공격 도구: Nmap Scripting Engine (User-Agent 필드에서 확인)
- 공격 시간: 10:00:05 시점부터 약 4초간 집중적으로 수행되었다
  (실제 공격에선 더 긴 시간 공격이 수행되었을 수 있음)

② 공격자의 목적 파악

- GET /admin.php
  관리자 페이지 존재 여부를 확인하려 시도했다.
- GET /config.json
  웹 애플리케이션의 설정 파일 존재 여부를 확인하려 시도했다.
- GET /.git/config 및 관련 변형 요청들 (.git/config.bak, .git/configure 등)
  Git 저장소 설정 파일(.git/config)이 외부에 노출되어 있는지 확인하려 시도했다.
  만약 이 파일이 노출되면 소스 코드 유출로 이어질 수 있다.

③ 피해 여부 판단

- 모든 요청에 대해 서버는 404 Not Found 응답 코드를 반환했다.
  이는 공격자가 찾으려던 파일들이 해당 경로에 존재하지 않았음을 의미한다.
  따라서 이 특정 공격 시퀀스로 인한 직접적인 피해(정보 유출 등)는 발생하지 않았다고 판단할 수 있다.
- 만약 여기서 200 OK 응답 코드가 있었다면, 해당 파일이 유출되었음을 의미하므로 침해 사고가 발생한 것으로 간주하고 대응해야 한다.

④ 분석을 위한 커맨드 예시

특정 IP의 모든 활동 확인	Linux (Bash)	grep '242.41.4.1' access.log
	Windows (Powershell)	Select-String -Path u_ex*.log -Pattern '242.41.4.1'
Nmap 스캐너를 사용한 모든 IP 확인	Linux (Bash)	grep 'Nmap Scripting Engine' access.log | awk '{print $1}' | sort -u
	Windows (Powershell)	Select-String -Path u_ex*.log -Pattern 'Nmap Scripting Engine' | ForEach-Object { ($_.Line.Split(' ')[8]) } | Get-Unique
config 파일 접근 시도 탐지	Linux (Bash)	grep 'config' access.log
	Windows (Powershell)	Select-String -Path u_ex*.log -Pattern 'config'

(2) SQL 인젝션(Injection) 공격 탐지

공격자는 웹 애플리케이션의 입력 파라미터에 악의적으로 조작된 SQL 쿼리를 삽입함으로써, 데이터베이스에 직접 접근하여 정보를 조회하거나 조작하는 공격을 수행한다. 이러한 공격은 일반적으로 입력값에 대한 검증이 부족하거나, 쿼리 작성 시 사용자 입력이 동적으로 포함되는 구조에서 발생한다. SQL Injection은 인증 우회, 민감정보 유출, 데이터 변조 등 다양한 침해 행위로 이어질 수 있으며, 웹 애플리케이션에 대해 치명적인 위협 중 하나로 분류된다.

탐지 지표	① URL의 쿼리 스트링(Query String)이나 POST 데이터에 SQL 구문 또는 주석(', " , --, #, /* … */)이 포함된 경우 ② UNION, SELECT, FROM, WHERE, OR 1=1, SLEEP() 등 SQL 예약어나 함수가 포함된 경우 ③ 데이터베이스 에러와 관련된 500 Internal Server Error 응답이 발생하는 경우

로그 예시 (IIS)
2024-07-11 01:15:30 172.21.13.2 GET /products.aspx id=1'+or+'1'='1'-- - 80 - 192.168.10.10 Mozilla/5.0+... - 500 0 0 120
2024-07-11 01:15:31 172.21.13.2 POST /login.aspx username=admin&password=password'+or+'1'='1' - 80 - 192.168.10.10 Mozilla/5.0+... - 200 0 0 50
2024-07-11 01:15:32 172.21.13.3 GET /search.aspx query=test%27%20UNION%20SELECT%20null,@@version-- - 80 - 192.168.10.11 Mozilla/5.0+... - 200 0 0 70
2024-07-11 01:15:33 172.21.13.4 GET /data.aspx id=1%20AND%20(SELECT%20SLEEP(5)) - 80 - 192.168.10.12 Mozilla/5.0+... - 200 0 0 5100
2024-07-11 01:15:34 172.21.13.5 GET /news.aspx category=1%20AND%201=CAST(0x7e AS NUMERIC) - 80 - 192.168.10.13 Mozilla/5.0+... - 500 0 0 100
2024-07-11 01:15:35 172.21.13.6 GET /item.aspx item_id=1%20AND%20(SELECT%20COUNT(*)%20FROM%20information_schema.tables)>0 - 80 - 192.168.10.14 Mozilla/5.0+... - 200 0 0 60
2024-07-11 01:15:36 172.21.13.7 GET /user.aspx user_id=1%20AND%20SUBSTRING(VERSION(),1,1)='5' - 80 - 192.168.10.15 Mozilla/5.0+... - 200 0 0 80

로그 예시는 다양한 SQL 인젝션 공격 시도를 보여준다. 다음과 같은 단계로 분석을 수행할 수 있다.

① 공격자 정보 식별

- 공격 IP: 172.21.13.2부터 172.21.13.7까지 여러 IP에서 공격이 시도되었다.
  이는 단일 공격자가 IP를 변경하며 시도하거나, 여러 공격자가 동시에 공격을 시도하는 상황일 수 있다.
- 공격 시간: 01:15:30부터 약 6초간 집중적으로 수행되었다.
- 사용 도구: 특정 도구의 User-Agent는 없지만, 패턴화된 쿼리 스트링으로 미루어 자동화된 도구(예: sqlmap)를 사용했을 가능성이 크다.

② 공격자의 목적 파악

- id=1'+or+'1'='1'--
  인증 우회 또는 모든 데이터 조회를 시도하는 전형적인 불리안 기반(Boolean-based) SQL 인젝션이다. --는 뒤의 SQL 구문을 주석 처리하여 오류를 방지한다.
- username=admin&password=password'+or+'1'='1'
  로그인 페이지에서 인증을 우회하여 관리자 권한을 획득하려는 시도이다.
- query=test%27%20UNION%20SELECT%20null,@@version--
  UNION SELECT를 사용하여 데이터베이스 버전 정보(@@version)를 탈취하려는 시도이다. 이는 정보 수집 단계에서 자주 사용된다.
- id=1%20AND%20(SELECT%20SLEEP(5))
  Time-based Blind SQL Injection 시도로, 응답 시간 지연을 통해 특정 조건의 참/거짓 여부를 판단하여 정보를 추출하려는 시도이다.
- category=1%20AND%201=CAST(0x7e AS NUMERIC)
  Error-based SQL Injection 시도로, 데이터베이스 오류 메시지를 강제로 발생시켜 오류 메시지 내에 포함된 정보를 탈취하려는 시도이다. 0x7e는 ~ 문자를 의미하며, 이를 숫자로 변환하려 할 때 오류가 발생한다.
- item_id=1%20AND%20(SELECT%20COUNT(*)%20FROM%20information_schema.tables)>0
  블라인드 SQL 인젝션으로, information_schema.tables 테이블의 존재 여부를 확인하여 데이터베이스 구조를 파악하려는 시도이다.
- user_id=1%20AND%20SUBSTRING(VERSION(),1,1)='5'
  블라인드 SQL 인젝션으로, 데이터베이스 버전의 첫 글자가 '5'인지 확인하는 등 특정 문자열을 추측하여 정보를 탈취하려는 시도이다.

③ 피해 여부 판단

- id=1'+or+'1'='1'-- (sc-status: 500)
  서버가 500 Internal Server Error를 반환했다. 이는 공격 쿼리로 인해 데이터베이스에서 예외가 발생했음을 의미한다. 직접적인 정보 유출은 없었을 가능성이 높지만, 공격자는 이 오류 응답을 통해 해당 파라미터가 SQL 인젝션에 취약하다는 중요한 정보를 얻었을 것이다.
- username=admin&password=password'+or+'1'='1' (sc-status: 200)
  200 OK가 반환되었다. 이는 로그인 우회에 성공했을 가능성이 매우 크므로 즉각적인 조사가 필요하다.
- query=test%27%20UNION%20SELECT%20null,@@version-- (sc-status: 200)
  200 OK가 반환되었다. 이는 데이터베이스 버전 정보가 공격자에게 노출되었을 가능성이 크다.
- id=1%20AND%20(SELECT%20SLEEP(5)) (sc-status: 200, time-taken: 5100)
  200 OK가 반환되었지만, 응답 시간(time-taken)이 5100ms (5.1초)로 비정상적으로 길다. 이는 공격이 성공하여 데이터베이스 지연이 발생했음을 의미하며, 공격자는 이를 통해 타이밍 기반 인젝션이 가능함을 확인할 수 있다.
- category=1%20AND%201=CAST(0x7e AS NUMERIC) (sc-status: 500)
  500 Internal Server Error가 반환되었다. 이는 오류 기반 SQL 인젝션 시도가 성공하여 데이터베이스 오류 메시지가 공격자에게 노출되었을 가능성이 크다. 오류 메시지 내에 민감한 정보(예: 테이블명, 컬럼명)가 포함될 수 있다.
- item_id=1%20AND%20(SELECT%20COUNT(*)%20FROM%20information_schema.tables)>0 (sc-status: 200)
  200 OK가 반환되었다. 이는 불리언 기반 블라인드 SQL 인젝션 시도가 성공하여 information_schema.tables 테이블의 존재 여부가 확인되었음을 의미한다. 공격자는 이 방식으로 데이터베이스 스키마를 추측할 수 있다.
- user_id=1%20AND%20SUBSTRING(VERSION(),1,1)='5' (sc-status: 200)
  200 OK가 반환되었다. 이는 블라인드 SQL 인젝션 시도가 성공하여 데이터베이스 버전의 일부 정보가 확인되었음을 의미한다. 공격자는 이 방식으로 데이터베이스의 정확한 버전을 알아낼 수 있다.

(3) 무차별 대입 및 크리덴셜 스터핑 공격 탐지

무차별 대입 공격은 특정 계정에 대해 가능한 모든 비밀번호 조합을 시도하여 로그인을 알아내는 공격이며, 크리덴셜 스터핑은 다른 곳에서 유출된 대량의 ID/비밀번호 목록을 가지고 특정 사이트에 대입하여 로그인을 시도하는 공격이다. 두 공격 모두 대량의 로그인 시도를 유발한다는 공통점이 있다.

탐지 지표	① 짧은 시간 동안 특정 IP 또는 여러 IP에서 로그인 페이지 (예: /login.php, /auth)로의 POST 요청이 대량으로 발생하는 경우 ② 특정 IP에서 다수의 로그인 실패(응답 코드 401 Unauthorized 또는 로그인 페이지로 재요청) 후 성공(응답 코드 302 Found 리다이렉션 또는 응답 사이즈의 변화)이 관찰되는 경우 ③ 정상적인 사용자의 활동으로 보기 어려운, 비정상적으로 많은 수의 로그인 시도가 탐지되는 경우

로그 예시 (Apache)
185.191.171.10 - - [11/Jul/2024:11:25:01 +0900] "POST /login.php HTTP/1.1" 200 1150 "http://victim.com/login.php"
185.191.171.20 - - [11/Jul/2024:11:25:02 +0900] "POST /login.php HTTP/1.1" 200 1150 "http://victim.com/login.php"
185.191.171.51 - - [11/Jul/2024:11:25:02 +0900] "POST /login.php HTTP/1.1" 200 1150 "http://victim.com/login.php"
185.191.171.200 - - [11/Jul/2024:11:25:03 +0900] "POST /login.php HTTP/1.1" 200 1150 "http://victim.com/login.php"
185.191.171.12 - - [11/Jul/2024:11:25:04 +0900] "POST /login.php HTTP/1.1" 200 1150 "http://victim.com/login.php"
대량의 로그인 실패 로그
185.191.171.10 - - [12/Jul/2024:14:27:30 +0900] "POST /login.php HTTP/1.1" 302 20 "http://victim.com/login.php"

로그 예시에선 여러 IP(185.191.171.0/32 대역)를 동원하여 로그인 페이지에 대량의 요청을 보내는 무차별 대입 공격 혹은 크리덴셜 스터핑 공격의 전형적인 로그 패턴이 확인된다. 두 개의 공격 방식을 웹 로그만으로는 구분하기 힘들지만, 최종적으로 공격자가 정상 계정의 확보 및 인증을 목표로 하는 점이 일치한다.

① 공격자 정보 식별

- 공격 IP: 185.191.171.10을 포함한 다수의 IP가 동원되었다. 이는 탐지를 피하기 위해 IP를 분산시키는 봇넷을 사용했을 가능성을 확인할 수 있다.
- 공격 시간: [11/Jul/2024:11:25:01] ~ [12/Jul/2024:14:27:30] 간 약 하루에 걸쳐 공격이 진행되었다.

② 공격자의 목적 파악

- 모든 요청이 로그인 페이지인 /login.php에 POST 메소드로 집중되어 있다. 이는 명백히 계정 접근 권한을 탈취하려는 시도이다.
- 여러 IP가 동원된 점으로 보아, 단일 계정을 노리는 무차별 대입 공격보다는 다수의 계정 정보를 대입하는 크리덴셜 스터핑 공격일 가능성이 크다.

③ 피해 여부 판단

- 대부분의 요청은 응답 코드 200 OK와 함께 응답 크기 1150을 반환했다.
  "아이디 또는 비밀번호가 틀렸습니다"와 같은 메시지가 포함된 로그인 실패 페이지를 의미하는 것으로 추정할 수 있다. 같은 페이지에 대한 요청과 같은 응답으로 보아, 지속해서 로그인에 시도 후 실패한 것으로 판단할 수 있다.
- 하지만 마지막 로그에서 185.191.171.10 IP의 요청은 응답 코드 302 Found와 응답 크기 20을 반환했다. 이는 일반적으로 로그인 성공 후 메인 페이지나 대시보드로 리다이렉션(Redirection)될 때 나타나는 패턴이다.
- 따라서, 이 302 응답은 공격자가 최소 하나 이상의 계정 정보를 탈취하여 로그인에 성공했음을 의미하므로, 심각한 침해사고로 간주하고 즉시 대응해야 한다.

④ 분석을 위한 커맨드 예시

로그인 페이지 요청 IP (Top 10)	Linux (Bash)	grep "POST /login.php" access.log | awk '{print $1}' | sort | uniq -c | sort -nr | head -n 10
	Windows (Powershell)	Select-String -Path 'u_ex*.log' -Pattern "POST /login.php" | Group-Object -Property {($_.Line.Split(' ')[8])} | Sort-Object Count -Descending | Select-Object -First 10
IP별 로그인 요청 횟수 조회	Linux (Bash)	grep "POST /login.php" access.log | awk '{print $1}' | sort | uniq -c | sort -nr
	Windows (Powershell)	Select-String -Path 'u_ex*.log' -Pattern "POST /login.php" | Group-Object -Property {($_.Line.Split(' ')[8])} | Sort-Object Count -Descending

(4) 웹쉘(Web Shell) 업로드 및 실행 탐지

웹쉘(Web Shell)은 공격자가 대상 웹 서버의 제어권을 획득하거나 지속적인 접근을 확보하기 위해 사용하는 악성 스크립트 파일이다. 이는 일반적으로, .php, .jsp, .asp, .aspx 등 서버 측 스크립트 언어로 작성되며, 웹 서버에 업로드되었을 경우 HTTP 요청을 통해 서버에서 임의의 시스템 명령을 원격으로 실행할 수 있도록 한다.

탐지 지표	① 업로드 시도: 파일 업로드 기능이 있는 페이지(예: /upload.php)에 .php, .jsp, .asp 등 웹 스크립트 파일이 업로드되는 경우. 또는 이중 확장자(.php.jpg, .asp;.txt)를 사용하여 우회를 시도하는 경우. POST 요청의 응답 사이즈가 비정상적으로 큰 경우도 의심해볼 수 있다. ② 실행 시도: 업로드된 웹쉘 파일에 cmd, shell, exec, system, passthru 등 명령 실행과 관련된 파라미터가 전달되는 경우。 ③ 비정상적인 파일명: shell.php, c99.php, r57.php 등 잘 알려진 웹쉘 파일명이 탐지되는 경우

로그 예시 (Apache)
21.52.124.10 - - [11/Jul/2024:12:05:10 +0900] "GET /board/write.php HTTP/1.1" 200 150 "-" "Mozilla/5.0 ..."
21.52.124.10 - - [11/Jul/2024:12:05:16 +0900] "POST /board/upload.php HTTP/1.1" 200 243 "http://victim.com/board/write.html" "Mozilla/5.0 ..."
n회 POST 요청 반복
21.52.124.10 - - [11/Jul/2024:12:06:11 +0900] "POST /board/upload.php HTTP/1.1" 200 243 "http://victim.com/board/write.html" "Mozilla/5.0 ..."
21.52.124.10 - - [11/Jul/2024:12:06:17 +0900] "GET /board/uploads/common.php HTTP/1.1" 200 150 "http://victim.com/board/upload.php" "Mozilla/5.0 ..."
21.52.124.10 - - [11/Jul/2024:12:06:22 +0900] "GET /board/uploads/common.php?cmd=whoami HTTP/1.1" 200 50 "-" "Mozilla/5.0 ..."
21.52.124.10 - - [11/Jul/2024:12:06:25 +0900] "GET /board/uploads/common..php?cmd=dir HTTP/1.1" 200 50 "-" "Mozilla/5.0 ..."
21.52.124.10 - - [11/Jul/2024:12:06:30 +0900] "GET /board/uploads/common..php?cmd=ifconfig HTTP/1.1" 200 50 "-" "Mozilla/5.0 ..."

로그 예시에선 클라이언트에서 서버에 파일을 업로드할 수 있는 포인트(write.php, upload.php)를 공격자가 접근한 것으로 확인된다. 공격자(21.52.124.10)는 해당 포인트를 이용해 웹쉘을 업로드하여 실행한 내역을 확인할 수 있다.

① 공격 흐름 분석

- 공격자 IP(21.52.124.10)가 /board/write.php에 접근하여 upload.php에 POST 요청을 보내 무언가를 업로드한 패턴이 확인된다. upload.php에 다수의 요청이 반복되었으며, 최종적으로 서버의 응답 코드가 200이었으며, 이는 업로드가 성공했음을 의미한다.
- upload.php에 다수의 요청이 반복됨을 통해 공격자는 악성 파일 업로드를 위한 우회 방법을 다수 시도한 것으로 판단할 수 있다.
  (예: 확장자 우회, MIME TYPE 변경, 이미지 웹쉘 업로드 등)
  로그만으로는 업로드된 파일명을 알 수 없지만, 파일 업로드 기능에 대한 POST 요청은 항상 주의 깊게 살펴봐야 한다.
- 동일 IP가 /board/uploads/common.php라는 파일을 GET 방식으로 요청하면서 cmd=whoami라는 파라미터를 전달했다. 이는 전형적인 웹쉘 실행 패턴이다. whoami는 현재 웹 서버가 어떤 사용자 권한으로 동작하는지 확인하는 명령이다.
- whoami와 동일한 시스템 명령어인 dir, ifconfig 등도 전달한 내역도 확인된다.
  예시 로그 외 웹셀의 종류마다 지원하는 기능과 파라미터가 상이하니 전체적인 동작 구조와 웹셀의 의미를 파악하고 있어야 한다.

② 피해 여부 판단

- 웹쉘 실행 요청에 대해 서버가 200 OK로 응답했다. 이는 웹쉘이 성공적으로 실행되었고, whoami 명령어 등의 결과(예: www-data, apache)가 공격자에게 전송되었음을 의미한다.
- 이는 서버의 제어권 일부가 공격자에게 넘어갔음을 의미하며, 공격자는 이를 발판으로 내부 시스템 정보를 수집하거나, 데이터베이스에 접근하거나, 다른 시스템으로 추가 공격을 감행할 수 있다. 이는 웹 서버를 장악 후 내부 자산으로 침투할 수 있는 상황이다.

③ 분석을 위한 실제 커맨드 예시

업로드 요청(성공) 검색	Linux (Bash)	grep -i "POST /board/upload.php.* 200 " access.log
	Windows (Powershell)	Select-String -Path 'u_ex*.log' -Pattern "POST /board/upload.php.* 200 "
의심 파라미터 및 주요 웹셸 조회 이력 검색	Linux (Bash)	grep -i -E "(cmd=|exec=|shell=|system=|passthru=|c99.php|r57.php)" access.log
	Windows (Powershell)	Select-String -Path 'u_ex*.log' -Pattern "(cmd=|exec=|shell=|system=|passthru=|c99.php|r57.php)"
업로드 폴더 내 의심 파일 조회	Linux (Bash)	find /board/uploads -type f \(-name "*.php" -o -name "*.jsp" -o -name "*.asp" -o -name "*.aspx" -o -name "*.phtml" \) -ls
	Windows (Powershell)	Get-ChildItem -Path \board\uploads -Include "*.php", "*.jsp", "*.asp", "*.aspx" -Recurse -File

(5) 경로 조작(Path Traversal) 공격 탐지

경로 조작(Path Traversal)은 공격자가 웹 애플리케이션의 파일 경로 관련 입력값을 조작하여, 애초에 접근이 허용되지 않은 서버 내부의 민감한 파일 또는 디렉터리에 접근하는 공격 기법이다. 이 공격은 주로 URL 또는 파라미터 내 파일명을 조작하거나, 디렉터리 상위 경로를 의미하는 ../(또는 %2e%2e/, %252e%252e/ 등 URL 인코딩된 우회 형태)를 삽입함으로써 발생한다.

탐지 지표	① URL에 상위 디렉터리로 이동하려는 시도(../ 또는 ..%2F, .. 등)가 포함된 경우 ② /etc/passwd, C:\Windows\System32\drivers\etc\hosts 등 시스템 파일에 대한 접근 시도가 포함된 경우

로그 예시 (Apache)
1.24.124.1 - - [11/Jul/2024:11:30:40 +0900] "GET /detail.php?file=..%2F..%2F..%2F..%2Fetc%2Fpasswd HTTP/1.1" 200 503 "-" "Mozilla/5.0 ..."
1.24.124.1 - - [11/Jul/2024:11:30:41 +0900] "GET /images.php?img=..%c0%af..%c0%af..%c0%afboot.ini HTTP/1.1" 404 498 "-" "Mozilla/5.0 ..."
1.24.124.1 - - [11/Jul/2024:11:30:42 +0900] "GET /view.php?page=../../../../../../../../../../../windows/win.ini HTTP/1.1" 500 850 "-" "Mozilla/5.0 ..."
1.24.124.1 - - [11/Jul/2024:11:30:43 +0900] "GET /app/file?path=....//....//....//etc/hosts HTTP/1.1" 403 495 "-" "Mozilla/5.0 ..."

로그는 동일한 IP의 공격자가 우회 기법을 사용하여 경로 조작 공격을 시도하는 것을 보여준다.

① 공격자의 목적 파악

- file=..%2F..%2F..%2F..%2Fetc%2Fpasswd
  URL 인코딩된 ../를 반복 사용하여 리눅스/유닉스 시스템의 사용자 계정 정보 파일인 /etc/passwd에 접근하려 시도했다.
- img=..%c0%af..%c0%af..%c0%afboot.ini
  일반적인 ../ 필터링을 우회하기 위해 비표준 URL 인코딩(%c0%af)을 사용하여 윈도우 시스템의 부팅 설정 파일인 boot.ini에 접근하려 시도했다.
- page=../../../../../../../../../../../../windows/win.ini
  ../를 매우 많이 사용하여 웹 루트 디렉터리부터 최상위 디렉터리까지 거슬러 올라가 윈도우 설정 파일인 win.ini에 접근하려 시도했다.
- path=..../...../...../etc/hosts
  ../ 필터링을 우회하기 위해 ..../ 와 같은 변형된 패턴을 사용하여 시스템의 호스트 파일인 /etc/hosts에 접근하려 시도했다.

② 피해 여부 판단

- /etc/passwd 요청 (응답 코드 200)
  서버가 200 OK를 반환했다. 이는 /etc/passwd 파일의 내용이 공격자에게 유출되었음을 의미한다.
- boot.ini 요청 (응답 코드 404)
  서버가 404 Not Found를 반환했다. 이는 해당 경로에 파일이 존재하지 않음을 의미하므로 공격은 실패했다.
- win.ini 요청 (응답 코드 500)
  서버가 200 OK를 반환했다. 이는 애플리케이션 레벨에서 비정상적인 요청으로 인해 오류가 발생했음을 의미하며, 공격은 실패했을 가능성이 크다.
- /etc/hosts 요청 (응답 코드 403)
  서버가 403 Forbidden을 반환했다. 이는 파일은 존재하지만 접근 권한이 없어 실패했음을 의미한다.

③ 분석을 위한 실제 커맨드 예시

경로 조작 패턴 탐지	Linux (Bash)	grep -i -E "(\.\./|\.\\|%2e%2e%2f|%2e%2e%5c|%c0%af)" access.log
	Windows (Powershell)	Select-String -Path 'u_ex*.log' -Pattern "(\.\./|\.\\|%2e%2e%2f|%2e%2e%5c|%c0%af)"
IP별 공격 횟수 집계	Linux (Bash)	grep -i -E "(\.\./|\.\\|%2e%2e%2f|%2e%2e%5c|%c0%af)" access.log | awk '{print $1}' | sort | uniq -c | sort -nr
	Windows (Powershell)	Select-String -Path 'u_ex*.log' -Pattern "(\.\./|\.\\|%2e%2e%2f|%2e%2e%5c|%c0%af)" | ForEach-Object { ($_.Line.Split(' ')[8]) } | Group-Object | Sort-Object Count -Descending

04 로그 분석 방법론 및 도구

효과적인 로그 분석을 위해서는 체계적인 방법론과 적절한 도구의 활용이 필수적이다.

(1) 분석 방법론

① 기준선 설정 (Baseline Establishment)

- 설명: 평상시 시스템의 정상적인 로그 패턴을 파악하고 기준선을 설정하는 것은 이상 징후 탐지의 가장 기본이 되는 단계이다. "정상"이 무엇인지 알아야 "비정상"을 식별할 수 있기 때문이다.
- 방법: 최소 수일에서 수 주간의 로그를 수집하여 시간대별 평균 요청 수, 주요 접속 국가, 상위 요청 페이지, 주된 User-Agent, 정상적인 응답 코드(2xx, 3xx)의 비율 등을 통계적으로 분석하여 기준선을 수립한다.

예시

- 평일 오전 9시부터 오후 6시까지 요청이 집중되고, 그 외 시간에는 요청이 뜸하다.
- 주요 접속 국가는 대한민국(KR)이며, 다른 국가에서의 접속은 드물다.
- 가장 많이 요청되는 페이지는 /main, /list, /login이다.
- 404 Not Found 오류는 시간당 평균 5회 미만으로 발생한다.

② 통계 기반 분석 (Statistical Analysis)

- 설명: 기준선과 비교하여 현재 로그 데이터가 통계적으로 유의미한 차이를 보이는지 분석하는 방법이다. 특정 지표의 급증이나 급감은 공격의 징후일 수 있다.
- 방법: 특정 필드를 기준으로 로그를 집계하여 기준선과 비교하고 이상 징후를 파악한다.

예시

- IP 주소 기준: 특정 IP에서 평소보다 수백, 수천 배 많은 요청이 발생한다면 웹 스캐닝이나 무차별 대입 공격을 의심할 수 있다.
- HTTP 상태 코드 기준: 4xx (클라이언트 오류)나 5xx (서버 오류) 코드의 발생 빈도가 갑자기 급증했다면, 이는 각각 웹 스캐닝/경로 조작 공격이나 SQL 인젝션/애플리케이션 오류를 의미할 수 있다.
- 요청 시간 기준: 새벽 시간대와 같이 평소 요청이 거의 없던 시간에 요청이 급증하는 경우, 이는 자동화된 공격일 가능성이 크다.
- 응답 크기 기준: 특정 요청에 대한 응답 크기가 비정상적으로 크거나 작다면, 이는 데이터 유출이나 공격 실패를 나타낼 수 있다.

③ 시그니처 기반 분석 (Signature-based Analysis)

- 설명: 이미 알려진 공격 패턴(시그니처)의 목록을 만들어두고, 로그에서 해당 패턴과 일치하는 내용을 탐지하는 가장 직접적인 분석 방법이다.
- 방법: 정규 표현식(Regular Expression)을 사용하여 로그의 특정 필드(주로 URL, User-Agent, POST 데이터)에서 공격 키워드를 검색한다.

예시
• SQL 인젝션: UNION SELECT, or 1=1, SLEEP() • 경로 조작: ../, ..\, /etc/passwd • 웹쉘 실행: cmd=, exec= • 스캐너: Nmap, Nikto, Acunetix

④ 상관관계 분석 (Correlation Analysis)

- 설명: 단일 시스템의 로그만으로는 파악하기 어려운 복합적인 공격의 전체 흐름을 재구성하기 위해, 여러 소스의 로그를 시간 순서에 따라 연계하여 분석하는 고급 분석 기법이다.
- 방법: 웹 서버 로그뿐만 아니라 웹방화벽(WAF), 침입 탐지 시스템(IDS/IPS), 데이터베이스, 운영체제(OS), 프록시 서버 등 여러 장비와 시스템의 로그를 통합 로그 관리 시스템(ELK, Splunk 등)으로 모아 분석한다.

예시
• 웹쉘 공격: WAF 로그에서 파일 업로드 탐지 -> 웹 서버 로그에서 해당 파일에 대한 POST 요청 확인 -> OS 로그에서 해당 파일 생성 기록 확인 -> 웹 서버 로그에서 웹쉘 실행 파라미터와 함께 GET 요청 확인 -> OS 로그에서 자식 프로세스(예: whoami) 생성 기록 확인. 이 모든 과정을 시간 순으로 연결하여 공격의 전체 흐름을 파악한다. • HTTP 리퀘스트 스머글링: 프록시 서버 로그에는 정상적인 POST 요청 하나만 기록되었지만, 거의 동일한 시간에 웹 서버 로그에는 해당 POST 요청과 함께 출처를 알 수 없는 악성 GET

(2) 분석 활용 도구

① 통합 로그 관리 시스템:

- ELK Stack (Elasticsearch, Logstash, Kibana): 대용량 로그를 수집, 저장, 분석, 시각화할 수 있는 오픈소스 솔루션이다.
- Splunk: 강력한 검색 및 분석 기능을 제공하는 상용로그 관리 플랫폼이다.

② 커맨드 라인 도구:

- grep: 특정 문자열이나 정규 표현식 패턴을 검색한다.
- awk: 로그 파일을 필드별로 분리하고 가공하는 데 유용하다.
- sort, uniq, wc: 로그를 정렬하고, 중복을 제거하며, 통계를 내는 데 사용한다.

05 결론

본 문서는 웹 로그의 구조부터 시작하여 실제 공격 사례를 기반으로 한 탐지 및 분석 방법론까지 다루었다. 단순히 특정 공격의 패턴을 암기하는 것을 넘어, 로그에 남겨진 흔적을 통해 공격자의 의도를 파악하고 피해 범위를 추론하며, 체계적으로 대응하는 역량을 기르는 것을 목표로 한다.

- 보안 관제 및 침해사고 대응: 실제 보안 관제 업무나 침해사고 발생 시, 이 문서를 참고하여 공격 유형을 신속하게 판단하고, 분석 커맨드를 활용하여 즉각적인 조치를 수행하는 데 활용할 수 있다.
- 훈련 및 교육 자료: 내부 보안 팀원이나 개발자를 대상으로 한 보안 교육에서 실제 로그 예시와 분석 방법을 보여주는 실습 자료로 활용하여, 이론이 아닌 실무 중심의 역량을 강화할 수 있다.
- 탐지 룰 개발: 여기에 제시된 공격 패턴과 키워드를 기반으로 통합 로그 관리 시스템(SIEM, ELK 등)의 탐지 룰을 개발하거나 고도화하여, 향후 유사한 공격에 대한 자동 경고 및 대응 체계를 구축할 수 있다.

결론적으로, 웹 로그는 모든 공격과 시스템 활동의 가장 기본적인 증거가 기록물이다. 본 문서가 단순히 지식의 나열로 끝나지 않고, 여러분이 로그 데이터 속에서 의미 있는 단서를 찾아내고 분석할 수 있는 발판이 되기를 바란다. 꾸준한 학습과 실습을 통해 악성행위를 판단하는 능력을 갖추게 될 때, 비로소 보이지 않는 위협으로부터 내부 자산을 안전하게 지킬 수 있을 것이다.