https://doi.org/10.7840/kics.2024.49.10.1397

Linux 로그 분석을 통한 공격자 차단 시간의 동적 설정 방법 연구

조 진 용◆, 박 주 원*, 김 승 해*, 조 부 승*

A Study on Dynamic Configuration of Attacker Blocking Time through Linux Log Analysis

Jinyong Jo◆, Ju-Won Park*, Seung-Hae Kim*, Buseung Cho*

요 약

네트워크 방화벽은 외부의 보안 공격으로부터 내부 자원을 보호하는 핵심 보안 요소이지만, 방화벽 규칙이 증가하면 성능이 저하되고 관리 부담이 늘어나는 문제가 있다. 특히, 소규모 방화벽은 수용 가능한 규칙의 수가 제한되어 있어 장기간 지속되는 다수의 공격을 효과적으로 방어하기 어렵다. 본 연구는 방화벽 규칙의 유효 시간을 동적으로 설정할 수 있는 MALP(More Attack, Longer Penalty) 알고리즘을 제안하여 소규모 방화벽이 갖는 규칙 수용 공간의 부족과 성능 저하 문제를 완화하고자 한다. 제안한 알고리즘은 호스트 기반 침입탐지 시스템에서 수집한 Linux 로그 데이터를 분석하여 얻은 공격 유형별 공격 차단 간격 등의 공격 특성을 활용하여 차단 시간을 설정하고, 벌칙 시간에 추가 공격이 탐지되면 차단 시간을 늘려가는 방식으로 동작한다. 모의 환경에서 수행된 테스트 결과 제안한 알고리즘의 성능 이득은 약 81.22%로, 적은 수의 필터 규칙으로도 보안 공격을 효과적으로 차단할 수 있었다.

키워드 : 로그 분석, 방화벽, 보안 자동화, ACL 설정, 공격 방어
Key Words : Log analysis, firewall, security automation, ACL configuration, attack protection

ABSTRACT

Network firewalls are a key security element that protects internal resources from security attacks. However, as the number of firewall rules increases, the performance of firewalls degrades and the management burden increases. In particular, small-capacity firewalls can accommodate only a limited number of rules, making it difficult to defend against persistent attacks. This study proposes the MALP (More Attack, Longer Penalty) algorithm that can dynamically set the attack blocking time to mitigate the lack of rule capacity and performance degradation problems. MALP adaptively increases the blocking time when subsequent attacks are detected during the penalty time. It leverages data features such as attack intervals, obtained by Linux logs collected from a host-based intrusion detection system. Computer simulation showed that the proposed algorithm yielded an 81.22% performance improvement, effectively blocking attacks with a small number of firewall rules.

※ 본 연구는 한국과학기술정보연구원(K24L4M1C1)의 지원으로 수행되었습니다.
◆ First Author : Korea Institute of Science and Technology Information, jiny92@kisti.re.kr, 정회원
* Korea Institute of Science and Technology Information, juwon.park@kisti.re.kr; shkim@kisti.re.kr; bscho@kisti.re.kr, 정회원
논문번호 : 202405-093-B-RU, Received May 8, 2024; Revised June 12, 2024; Accepted July 2, 2024

Ⅰ. 서 론

보안 공격이 더 복잡해지고 지능적으로 변하고 있지만, 데이터 유출 사고의 약 80%는 무작위 대입 공격(Brute-force attack)과 같은 일상적인 공격으로 인해 발생한 것으로 알려져 있다[1]. 네트워크 방화벽은 경계선 방어를 위한 보안의 핵심 요소로서 외부 공격으로부터 내부의 디지털 자원을 보호한다. 하지만 방화벽은 규칙 모음(Rule set)의 규모, 설정 오류의 존재 여부, 주석의 여부에 따라 관리 비용이 증감한다[2].

규칙 모음의 수는 방화벽의 패킷 포워딩 성능에도 영향을 미친다. 대표적인 소프트웨어 방화벽인 iptables와 nftables의 경우, 규칙 모음의 수가 증가하면 TCP 처리량(Throughput)이 저하된다. 체인의 종류(예, INPUT, OUTPUT, FORWARD 등)나 서버의 사양에 따라 성능 차이는 있겠지만 iptables는 약 8,000에서 28,000개의 규칙모음부터 성능 감쇄가 발생한다[3]. 상용 방화벽도 ACL(Access Control List) 필터 규칙(Filter rule)의 수가 증가하면 패킷 손실량이 증가하는 경향이 있다[4]. 패킷 스위칭 칩을 탑재하지 않은 방화벽들은 소프트웨어가 ACL 규칙[1]을 처리해야 하므로 규칙의 증가가 패킷 손실을 유발하거나 높은 지연 변이(Jitter)를 발생시킬 수 있다[5].

특히, 시그니처(Signature) 또는 머신러닝(ML, Machine Learning)을 기반으로 공격자를 탐지하는 웹 방화벽은 오탐(False alarm)을 피하기 어렵다[6,7]. 잦은 오탐은 사용자의 불만을 야기하고 서비스의 신뢰도를 하락시키는 등 운영 측면에서 부정적인 영향을 미칠 수 있다. 또한 HTTPS나 SSHD와 같이 암호화된 전송 채널을 사용하는 경우, 패킷에 대한 가시성을 확보하기 위해서는 OSI(Open Systems Interconnection) 7계층에서 동작하는 고가의 장치가 필요하다. 즉, 3/4 계층에서 동작하는 방화벽은 패킷의 내용을 검사할 수 없는 문제가 있다.

본 연구는 방화벽이 다수의 ACL 규칙을 가질 때 발생하는 규칙 저장 공간의 부족, 성능 저하 및 관리 어려움을 완화하기 위해 공격자 차단 규칙의 유효 시간을 동적으로 결정할 수 있는 MALP(More Attack, Longer Penalty) 알고리즘을 제안한다. 제안한 알고리즘은 호스트 기반 침입탐지 시스템(HIDS, Hosted-based Intrusion Detection System)에서 축적한 HTTP, SSHD, Mail(Postfix) 서비스의 로그 데이터를 분석해 얻은 공격 유형별 차단 간격과 같은 데이터 특성(Feature)과 벌칙 시간(Penalty time)이라는 개념을 도입하여 공격의 차단 시간을 설정한다. 벌칙 시간에 공격이 탐지되면 차단 시간을 증가시키는 방식이다.

본 연구의 기여점은 다음과 같다. 첫째, 방화벽의 성능 저하와 규칙 저장 공간의 부족 문제를 해결하기 위해 ACL 규칙의 유효 시간을 이용하는 첫 번째 연구이다. 제안한 알고리즘은 소규모 방화벽을 활용하는 서비스 환경에서 HTTP, SSHD, Mail에 대한 보안 공격을 비용 효율적으로 방어하는 데 활용될 수 있다. 둘째, 알고리즘의 실용 가능성(Feasibility)을 높이기 위해 웹 응용 서비스를 제공하고 있는 실제 서비스 운영 환경에서 1년 간 수집한 로그 데이터를 분석했다. 분석한 데이터는 ML/DL(Deep Learning)의 특성으로 활용될 수 있다.

본 논문은 다음과 같이 구성된다. 제2장에서 연구 동기와 HIDS에 대해서 설명하고 제3장은 관련 연구를 소개하고 본 연구의 제한 사항을 기술한다. 로그 데이터의 수집 방법과 분석 결과는 각각 제4장과 제5장에서 제시한다. 제6장에서 MALP 알고리즘을 설명하고 성능을 평가한다. 마지막으로 제7장에서 결론을 맺는다.

Ⅱ. 배 경

본 연구는 오탐 문제를 완화하고 암호화된 전송 채널을 사용하는 패킷에 대해서 가시성을 높이기 위해 공개 소스 방화벽 소프트웨어인 OSSEC(Open Source HIDS SECurity[8]) HIDS를 사용했다. 본 장에서는 연구 동기와 OSSEC의 구조를 소개한다.

2.1 연구 동기

우리는 2016년부터 사용자 인증인가 체계인 T&I 기반(Trust and Identity infrastructure)을 운영해 왔다. T&I 기반은 학·연 기관과 서비스제공자의 인증인가 체계를 표준화하여 기관 간 경계 없는 웹 통합인증 환경을 구현한다. T&I 기반은 12개의 가상 서버로 구성되어 있으며 인증인가 소프트웨어의 제공을 위해 HTTP 서비스와 Mail 서비스가 구동되고 있다. 또한 서버의 관리운영을 위해 SSHD 서비스가 활용되고 있다.

사용자 인증인가 정보를 처리해야 하는 T&I 기반의 경계선 방어를 위해 소규모 통합 위협관리(UTM, Unified Threat Management) 시스템을 활용했다. UTM은 방화벽, IPS(Intrusion Protection System), VPN(Virtual Private Network), Content/URL (Uniform Resource Locator) 필터링 등의 기능을 단일 장치에 통합한 보안 장치이다. 구축 초기의 T&I 기반은

1) 본 논문에서 ACL 규칙은 방화벽 규칙과 동일한 개념으로 사용한다.

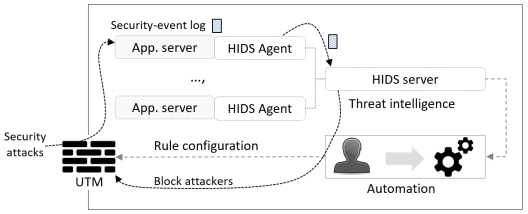

그림 1. 연구 목적의 개략도.
Fig. 1. High-level overview of research goal.

그림 1의 UTM과 인증인가 소프트웨어가 구동되는 응용서버(App. server)로만 이루어져 있었다. 그러나 UTM의 IPS에서 발생하는 오탐으로 인해 사용자가 인증인가 소프트웨어에 접근하지 못하는 상황이 빈번히 발생했다. 오탐 문제를 완화하기 위해 XSS(Cross Site Scripting) 공격과 SQL 주입 공격의 탐지를 제외한 IPS의 모든 기능을 비활성화하고, HIDS를 구축하여 비활성화된 IPS의 기능을 보완하려 했다.

구축된 HIDS는 탐지한 공격을 일정 시간 동안만 차단하므로 지속 시간이 긴 공격은 효과적으로 대응하지 못했다. 지속적으로 공격하는 공격자를 방어하기 위해 운영자가 HIDS의 로그를 분석하여 공격자를 식별하고 ACL 규칙을 수동으로 설정해야 했기 때문에, 운영자의 업무 부담이 많이 증가하는 문제가 발생했다.

업무 부담을 줄이기 위해 로그 분석과 공격자 식별 및 ACL 규칙 설정을 자동화하는 Python 스크립트를 개발하여 활용했다. 하지만 구축된 UTM이 수용할 수 있는 ACL 규칙의 수에 제한이 있어 자동화 된지 수개월 후부터는 규칙 설정이 불가능해졌다. ACL 규칙의 수용 용량 제한 문제를 해결하기 위해 선입선출 방식으로 규칙을 설정하도록 스크립트를 수정했으나, 장기간 지속되는 공격에 대해서는 여전히 대응이 어려웠다.

결과적으로 UTM에 설치해야 하는 규칙의 수를 줄이기 위해 ACL 규칙의 유효 시간을 동적으로 결정하는 방안을 연구하게 되었다.

2.2 OSSEC HIDS

OSSEC은 공개 소스 보안 소프트웨어로써 로그 분석, 파일 무결성(Integrity) 검사, 악성 소프트웨어(Rootkit) 탐지 및 실시간 알림과 공격 대응(Active response) 등의 기능을 제공한다[8,9]. 시스템 로그를 분석해 침입을 탐지하므로 LIDS(Log-based IDS)로도 불린다. PCRE(Perl Compatible Regular Expressions) 정규식과 공격의 강도 즉, 특정 시간 동안 집계된 동일한 공격 유형(Pattern)의 공격 횟수를 이용해 공격을 탐지한다. 로그의 수집, 분석, 경고 및 대응 과정이 목적 서버(OSSEC 에이전트가 설치된 서버)에서 실행되는 LM(Local Model) 구조와 목적 서버와 제어 서버가 분리된 ASM(Agent/Server Model) 구조로 구분할 수 있다. 본 연구에서는 다수의 목적 서버를 효과적으로 관리하기 위해 ASM 구조를 채택했다.

그림 2는 ASM 구조에서 로그의 처리 과정을 보여준다. 목적 서버에 설치된 OSSEC 에이전트는 로그 수집기(Log collector)를 통해 syslog, Apache access 등의 로그를 수집하고 제어 서버(OSSEC 서버)에게 전송한다.

제어 서버는 수집한 모든 로그를 archives.log 파일에 저장한다. 디코더(Decoder)는 PCRE 정규식을 이용하여 공격의 유형을 파악하고 공격 시간, 공격자의 IP 주소, 호스트명, 프로그램명(예, Apache) 등의 세부 정보를 추출한다. 분석기(Analyzer)는 공격의 세부 정보가 탐지 규칙(Rule)에 부합하는지 확인하고 세부 정보와 공격의 심각도(Severity level) 등 추가 정보를 alerts.log 파일에 저장한다. 심각도가 6 이상인 공격에 대해서 제어 서버가 공격 대응을 명령하면 목적 서버는

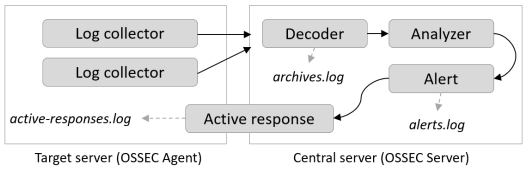

그림 2. 로그 처리 절차.
Fig. 2. Flow of processing logs.

표 1. 로그 처리 절차의 예
Table 1. Example of processing a log

raw log	Apr 18 00:02:11 localhost sshd[12005]: refused connect from 120.*.*.32 (120.*.*.32)
rule	<rule id="2503" level="6"> <pcre2>^refused connect from|</pcre2> <description>[description]</description> <group>access_denied,</group> </rule>
alerts	** Alert [event_id]: 2024 Apr 18 00:02:13 ([victim]) [IP addr. of victim]->[log file] Rule: 2503 (level 6) -> '[description].' Src IP: 120.*.*.32 Apr 18 00:02:11 localhost sshd[12005]: refused connect from 120.*.*.32 (120.*.*.32)
active response	2024 Apr 18 00:02:13 KST 2024 [script a] add - 120.*.*.32 [event_id] 2503 2024 Apr 18 00:12:13 KST 2024 [script d] delete - 120.*.*.32 [event_id] 2503

iptables나 TCPWrapper(예, hosts.allow 및 hosts.deny)와 같은 소프트웨어 방화벽을 이용해 공격을 차단한다.

표 1은 목적 서버에서 제어 서버에게 보낸 원본 로그가 공격으로 간주되어 보안 대응이 이루어지는 과정을 예시한다. 디코딩 과정은 생략했다. 디코더가 해석한 원본 로그가 'refused connect from'으로 시작한다. 공격 탐지 규칙 2503에 정의된 PCRE 정규식 즉, 'refused connect from으로 메시지가 시작'에 해당되므로, 규칙에 정의된 심각도 6을 부여하고 120.*.*.32를 공격자로 분류했다. 심각도가 6 이상이므로 제어 서버는 목적 서버에 공격 대응을 명령한다. 목적 서버는 [script a]를 실행해 공격자를 방화벽에 등록([script a] add)해 차단하고 사전에 설정된 시간(예, 600초)이 되면 해당 공격자를 차단 해제([script d] delete)한다.

Ⅲ. 관련 연구 및 제한 사항

우리가 아는 바로는 로그 분석을 통해 방화벽 규칙의 적용 시간을 자동으로 설정하는 연구는 아직 없는 것으로 파악된다. 본 장에서는 유관 연구를 소개하고 연구의 한계 사항을 기술한다.

3.1 관련 연구

로그를 활용한 이상(Anomaly) 탐지[10,11]나 이종 방화벽 간의 규칙 설정을 자동화하기 위한 언어 모델[12,13]은 보안 이벤트의 탐지나 방화벽과 관련해 잘 알려진 연구 주제들이다. 특히, 최근에는 로그 정보의 의미 파악(Semantic analysis)[14], 구문 분석(Log parsing)의 자동화[15], 거짓 양성(False positive)의 완화[16] 등 다양한 분야에서 ML/DL을 활용하고 있다. 하지만 본 연구에서는 오탐을 줄이기 위해 PCRE 정규식을 활용하여 로그를 분석하고, 정의된 규칙에 따라 공격을 탐지한다.

구문 해석이나 로그 분석과 관련해 본 연구와 유사성을 갖는 연구를 살펴본다. 유관 연구[17]는 SSH 무차별 대입 공격을 가하는 공격자들의 위협 수준을 등급화하기 위해 네트워크 라우터에서 수집한 공격의 빈도와 지속 시간 및 공격 국가를 데이터 특성으로 활용했다. 본 연구는 공격의 차단 시간을 예측하기 위해 공격의 심각도(L^s)와 차단 간격(I^b)을 특성으로 활용했으며, SSH 뿐만 아니라 HTTP와 Mail 서비스에 대한 공격에도 적용 가능하다는 점에서 유관 연구와 차이가 있다.

두 번째 유관 연구[18]는 소프트웨어 정의 네트워크(SDN, Software Defined Network)에서 HIDS의 보안 알림 메시지를 활용하여 공격을 차단하는 방법을 제안했다. HIDS와 연동된 SDN 제어기가 스위치의 플로우 테이블에 ACL 규칙을 적용하여 공격을 차단한다. HIDS를 활용해 공격을 차단하고자 한 방법은 본 연구와 유사하다. 하지만 본 연구는 HIDS의 보안 알림 메시지를 이용하지 않고 HIDS의 로그 데이터를 분석해 공격 차단 시간을 동적으로 조절한다는 점에서 차이가 있다.

3.2 연구의 한계

본 연구는 ACL 규칙의 유효 시간을 동적으로 설정할 수 있는 MALP 알고리즘을 제안하여 장기간 지속되는 보안 공격을 효과적으로 방어할 수 있음을 보이고, 성능 검증을 통해 실용 가능성을 입증한다. 하지만 연구에서 활용한 HIDS의 제약 사항과 데이터의 수집 환경 등으로 인해 기능과 성능에 다음과 같은 한계가 있을 수 있다.

첫째, 제안한 MALP 알고리즘은 보안 공격의 탐지에 사용된 HIDS 즉, OSSEC의 공격 탐지 성능에 영향을 받는다. 예를 들어, 아파치 400 오류(Apache 400 Bad Request)를 발생시키는 일부 웹 스캔 공격은 공격 간격을 조절해 OSSEC의 탐지 규칙을 지능적으로 회피하는 경향을 보였다. 회피된 공격이 분석한 데이터를 왜곡할 수 있다. 둘째, 공격에 대해서 방어의 적시성이 매우 중요한(Time-critical) 경우에는 HIDS를 방화벽과 연동하여 방어하는 방식은 효과적이지 못하다. 특히, HIDS가 ASM 구조를 가질 때는 공격 탐지 시간과 방어 시간 간에 수 초 이상의 시간차가 발생할 수 있다. 즉, 공격이 탐지된 이후에도 상당 시간 동안 공격을 방어하지 못하는 상황이 초래된다.

수집하여 분석한 로그의 종류(syslog, Apache access, mail 등)가 현재 운영 중인 T&I 기반에 종속되어 있어, 새로운 유형의 공격 로그가 발견되면 추가적인 데이터 분석이 필요하다. 또한 분석 결과의 일반성을 확인하기 위해서 분석에 사용하지 않은 별도의 로그 데이터를 수집하고 본 연구의 내용과 동일한 공격 특성이 나타나는지 검증할 필요가 있다.

Ⅳ. 연구 방법

4.1 데이터의 수집

T&I 기반을 구성하는 12개의 목적 서버와 추가로 구축된 1개의 제어 서버에서 로그를 수집했다. 총 13개의 서버 중에 SSHD, HTTP, Mail(Postfix) 서비스를 제공하는 서버는 각각 1개, 13개, 3개이다. 모든 서버는 Linux 운영체제를 탑재하고 있고 대한민국에 위치한다.

로그 분석을 위해 목적 서버와 제어 서버로부터 총

표 2. active-responses.log의 형태
Table 2. Format of active-responses.log

<timestamp><script_name><action><user><IP><alert_id><rule_id>[additional_data]
Mon Mar 25 07:58:35 KST 2024 [script a] add - 120.*.*.32 1711320485.1476072 30107

13개 active-responses.log 파일을 취합했다. active-responses.log는 제어 서버가 목적 서버에게 전달한 접근 차단 및 허용 명령의 로그를 표 2에 보이는 형태로 저장한다. HTTP와 Mail 서비스는 2023년 1월부터 12월까지, SSHD는 동년 7월부터 12월까지의 로그를 수집했다. SSHD 서비스의 로그를 확보하기 위해 해당 기간 동안 TCPWrapper를 설정하지 않았고 OSSEC HIDS로만 공격을 방어했다.

표 2의 <IP>는 공격자의 IP 주소이고 <rule_id>는 OSSEC에서 정의한 공격 규칙이다. 예를 들어, 공격 규칙 30107은 'Code Red' 공격을 의미한다. 본 연구에서 사용된 일부 공격 규칙들을 서비스별로 분류하면 표 3과 같다. OSSEC은 공격의 유형과 빈도에 따라 심각도(표 3의 level)를 결정한다. 심각도가 6 이상인 공격은 즉시 차단되고 6 미만인 공격은 특정 시간 동안 13회 이상 탐지되면 차단한다. 차단 시간은 600초로 설정했다.

표 4. CSV 파일의 형태
Table 4. Format of CSV file

<timestamp>,<attacker>,<victim>,<rule_id>,<group>,<attack_country>,...,<attack_weekday>

active-responses.log 파일에 포함된 개별 이벤트 로그는 데이터 분석을 위해 표 4과 같은 형태의 CSV(Comma-Separated Values) 파일로 재구성했다. 획득한 공격 규칙을 기준으로 서비스 그룹(<group>)을 'http', 'sshd', 'mail'로 분류했으며 GeoLite2 데이터베이스[19]를 이용해 IP 주소로부터 국가 정보(<attack_country>)를 획득했다. 접근이 허용된 IP 주소에서 발생하는 보안 이벤트(예, 로그인 실패 등)와 중복되는 공격 차단 로그는 CSV 파일에서 제거했다. OSSEC ASM 구조에서는 메시지의 왕복 지연과 서버 내 처리 지연으로 인해 즉, 제어 서버가 공격 차단을 지시한 시간과 목적 서버가 차단 규칙을 적용한 시간차로 인해 동일한 차단 로그가 중복될 수 있다.

표 3. rule ID의 설명
Table 3. Description of rule ID

	rule ID	level	description
http	31151	10	Multiple web server 400 error codes from same source ip
	31104	6	Common web attack
	31516	6	Suspicious URL access
	31164	6	SQL injection attempt
	31515	6	PHPMyAdmin scans (looking for setup.php)
	31103	6	SQL injection attempt
	31105	6	XSS (Cross Site Scripting) attempt
	31106	6	A web attack returned code 200 (success)
	31508	6	Blacklisted user agent (known malicious user agent)
	31162	10	Multiple web server 500 error code (Internal Error)
	31533	10	High amount of POST requests in a small period of time (likely bot)
	30107	6	Code Red attack
	31152	10	Multiple SQL injection attempts from same source ip
	30117	10	Invalid URI, file name too long
	30116	10	Multiple Invalid URI requests from same source
sshd	5551	10	Multiple failed logins in a small period of time
	5712	10	SSHD brute force trying to get access to the system
	5706	6	SSH insecure connection attempt (scan)
	5703	10	Possible breakin attempt (high number of reverse lookup errors)
	5758	8	Maximum authentication attempts exceeded
	2502	10	User missed the password more than one time
mail	3301	6	Attempt to use mail server as relay (client host rejected)
	3302	6	Rejected by access list (Requested action not taken)
	3357	10	Multiple SASL authentication failures
	3353	10	Multiple attempts to send e-mail from invalid/unknown sender domain
	3355	10	Multiple attempts to send e-mail to invalid recipient or from unknown sender domain

V. 데이터 분석

데이터 분석을 위해 Python Pandas[20] 라이브러리를 이용했다. 서비스 유형별, 국가별, 공격 규칙별로 공격을 분석했지만, MALP 알고리즘의 데이터 특성으로 국가별 공격 형태는 활용하지 않았기 때문에 국가별 공격 형태에 대해서는 본 논문에서 다루지 않는다.

표 5는 서비스 유형별 피공격자(목적 서버)에 대한 공격 및 공격자의 수, 단수 차단(Single block)의 횟수와 복수 차단(Multiple blocks)의 횟수를 보여준다. 단수 차단은 공격자의 공격 기간이 짧아 로그 데이터의 수집 기간 동안 한 번만 차단된 경우이다. 피공격자를 구분하지 않고 공격자를 기준으로 분석한 경우와 공격자와 피공격자를 1:1로 묶어 분석한 경우를 각각 교차 공격(Cross attack)과 비교차 공격(Non-cross)으로 정의했다. 피공격자를 구분하지 않으면 공격자가 여러 피공격자를 공격할 수 있으므로 복수 차단의 가능성이 커진다. 반면에 피공격자를 구분하면 공격자가 여러 피공격자를 공격한 경우는 고려하지 않으므로 단수 차단의 빈도가 높아진다.

고유 공격자(Unique attacker)는 IP 주소를 기반으로 분류했다. 본 연구에서 SSHD 서비스를 제공하는 서버는 하나이므로 즉, 하나의 피 공격자만 존재하므로 교차 공격에 대한 통계는 수집할 수 없다.

SSHD 서비스에 대한 로그 수집 기간이 HTTP나 Mail 서비스에 비해 0.5배 적음에도 불구하고 공격이 차단된 횟수(C^b)는 SSHD가 압도적으로 많은 것을 확인할 수 있다. SSHD 서비스에 대한 C^b는 HTTP 서비스에 비해 약 16.5배 많았다. C^b가 많다는 것은 공격의 횟수가 많았음을 의미한다. HTTP 서비스에 대한 공격에서는 하나의 공격자가 약 4.06번 차단되었지만, SSHD는 약 12.12회 차단되어 약 2.98배 많았다. SSHD 서비스에 대한 복수 차단의 비율은 약 91.8%였다. 종합해 보면 SSHD 서비스에 대한 공격은 HTTP와 비교해서 공격 강도와 지속성이 매우 높음을 알 수 있다.

비교차 공격의 경우에 HTTP의 단수 차단과 복수 차단의 비율이 각각 약 54.4%와 45.6%로써 공격의 지속성이 SSHD에 비해 낮은 것으로 볼 수 있다. 교차 공격 분석에서 복수 차단의 비율은 71.1%였다. 즉, 공격의 지속성은 SSHD에 비해 낮지만, 공격이 지속되면 공격의 강도가 증가하거나 일부 공격자들의 공격 강도가 매우 높은 것으로 유추된다.

5.1 서비스 유형별 공격 분석

표 6은 총 6,085 개의 고유 공격자가 SSHD 서비스를 공격하기 위해 사용한 공격 규칙의 수(C^r), 공격의 지속 시간(duration, D^b), C^b(blocks)를 보여준다. D^b의 중위값($Q2$)이 0인 것으로 보아 다수의 공격자들이 1회 차단되면 공격을 중단하는 것으로 볼 수 있다. 그러므로 모든 공격자들에 대해서 ACL 규칙을 장기간 유지하는 것은 비효율적일 수 있다. 임의의 고유 공격자는 평균 13.29회 차단되었으며 이상치(Outlier)를 제외한 제3사분위값($Q3$)은 4회였다. 표 7의 최대치(max)는 이상치를 포함한 최댓값이다. D^b의 $Q3$는 4,226초(1.17 시간)이지만 평균은 약 9.2e+05 초(약 256.94 시간)로써 특정 공격자들이 지속적으로 공격했음을 알 수 있다.

SSHD 서비스에 대한 공격을 위해 공격자가 가장 많이 사용한 공격 규칙은 5706과 5703이었다. 공격 규칙 5706은 SSH 스캔 공격이고 5703는 Break-in 의심

표 6. 고유 공격자 통계 (SSHD)
Table 6. Statistics of unique attackers (SSHD)

	mean	std	25%
rule ID	1.09	0.36	1
duration	9.2e+05	2.5e+06	0
blocks	13.29	164.20	1
	50%	75%	max
rule ID	1	1	4
duration	0	4.2e+03	1.5e+07
blocks	1	2	6359

표 5. 공격 요약
Table 5. Summary of attacks

service	sshd(%)	http(%)		mail(%)	
		cross	non-cross	cross	non-cross
total blocks	73,809	4,471		619	
unique attackers	6,085	1,101		322	
single block	6,084(8.2)	1,292(28.9)	2,436(54.4)	325(52.5)	327(53.8)
multiple blocks	67,724(91.8)	3,179(71.1)	2,035(45.6)	294(47.5)	292(47.2)

공격이다. SSHD 연결설정 과정에서 클라이언트로부터 식별 문자열을 받지 못하면 SSH 스캔 공격으로 분류한다. Break-in 의심 공격은 역방향 DNS(Domain Name Service)가 설정되어 있지 않거나 잘못 설정되어 발생하고 사용자 크리덴셜(Credential)에 대한 무작위 대입 공격과 병행해 발생한다.

공격자의 C^r과 D^b 및 C^b 사이에는 낮은 상관관계를 보였다. 상관관계가 상대적으로 높은 항목은 D^b와 C^b로써 상관 계수가 약 0.34인 것으로 확인되었다. 상관관계가 낮은 것은 SSHD 서비스에 대한 공격의 휴지기가 길기 때문으로 보인다.

표 7은 HTTP 서비스에 대해 고유 공격자의 C^r과 D^b 및 C^b를 보여준다. C^r은 최대 8개로써 고유 공격자별 최대 4개의 공격 규칙이 사용된 SSHD 서비스보다 많았다. HTTP에 대한 공격의 형태가 SSHD보다 다양하기 때문이다. 하나의 고유 공격자가 공격한 피공격자의 수는 $Q3$를 기준으로 2개로써 일반적으로 HTTP 서비스를 공격하는 공격자는 적은 수의 피공격자만 공격하는 것으로 볼 수 있다. 하나의 고유 공격자는 평균 1.06개의 공격 규칙을 사용해 피공격자를 공격했고 $Q3$를 기준으로도 1개의 공격 규칙을 사용했다. 공격의 지속 시간은 $Q3$를 기준으로 36,025 초(10.0 시간)로 나타났다.

HTTP 서비스를 공격하는 공격자는 웹 스캔 공격인 공격 규칙 31151, 은닉정보 탈취시도 공격인 31516과 디렉토리 횡단 공격인 31104, SQL 주입 공격인 31164를 많이 사용했다. 은닉정보 탈취시도 공격은 .htaccess나 .history 등 숨겨진 파일에 접근을 시도하는 공격이다.

HTTP 서비스에 대한 C^r와 D^b 및 피공격자의 총 수(C^v) 사이에서 낮은 상관관계를 보였다. D^b와 C^v

표 7. 고유 공격자 통계(HTTP)
Table 7. Statistics of unique attackers (HTTP)

	mean	std	25%
rule ID	1.06	0.45	1
victims	1.81	1.53	1
duration	1.0e+06	3.4e+06	0
blocks	3.56	14.20	1
	50%	75%	max
rule ID	1	1	8
victims	1	2	10
duration	0	3.6e+04	2.9e+07
blocks	1	2	302

표 8. 고유 공격자 통계(Mail)
Table 8. Statistics of unique attackers (Mail)

	mean	std	25%
rule ID	1.00	0.07	1
victims	1.00	0.07	1
duration	4.2e+05	2.6e+06	0
blocks	1.91	9.37	1
	50%	75%	max
rule ID	1	1	2
victims	1	1	2
duration	0	0	3.1e+07
blocks	1	1	166

사이의 상관 계수는 0.35, 공격의 총 수와 C^v 간의 상관계수는 0.36이었지만 C^b와 D^b 간에는 낮은 상관 계수(0.2)를 확인할 수 있었다. C^b와 C^r 사이에는 상대적으로 높은 상관 계수(0.42)를 보였다. 많이 공격할수록 많은 공격 규칙을 사용하는 것을 알 수 있다.

Mail 서비스에 대한 공격은 표본 수가 적어 분석 결과를 일반화하기 어렵다. 1년간 수집한 고유 공격자의 총 수는 322개였으며 총 619회 공격이 차단되었다. 하나의 고유 공격자에 대해 약 1.92회 공격이 차단된 것으로 보아 공격의 강도와 지속성이 낮은 것을 알 수 있다. 공격 규칙 3301 즉, postfix 서비스를 구동 중인 서버를 메일 메시지의 중개서버로 이용하려는 시도가 다수를 차지했다.

Mail 서비스에 대한 공격은 일반적으로 하나의 공격 규칙으로 하나의 피공격자만 공격하며 공격의 지속 시간도 매우 짧은 것(표 8 참조, $Q3$가 0)으로 판단된다. 하지만 일부 공격자는 오랫동안 공격을 가하는 것으로 확인된다. Mail 서비스에 대한 공격은 D^b와 C^b 사이에 0.66의 상관 계수를 보였다. 즉, 공격의 지속 시간과 공격 차단의 수가 비례함을 알 수 있다. C^r와 C^v 및 D^b 사이에는 0.19 이하의 매우 낮은 상관관계를 보였다.

5.2 공격 규칙별 공격 분석

본 절에서는 공격의 차단 간격(I^b)을 MALP 알고리즘의 주요 특성으로 활용하기 위해 공격 규칙별 I^b를 상세히 분석했다. 짧은 차단 간격은 많은 공격량을 의미한다.

그림 3은 공격자가 SSHD 서비스를 공격하기 위해 사용한 공격 규칙별 I^b를 보여준다. 녹색 삼각형은 평균을 의미한다. 공격량이 많은 상위 10개의 공격 규칙만 표시했다. 표 9는 그림 3의 공격 규칙 중에 공격 차단

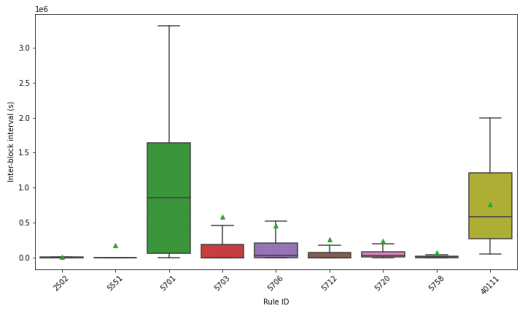

그림 3. 공격 규칙별 차단 간격(SSHD).
Fig. 3. Inter-block intervals per rule ID (SSHD).

표 9. SSHD에 대한 상위 4개 공격규칙별 차단 간격(초)
Table 9. Inter-block intervals of top 4 rule IDs (SSHD)

rule ID	blocks	mean (s)	min (s)
2502	52,012	8.2e+03	611
5758	5,679	6.7e+04	656
5706	5,096	4.6e+05	629
5703	2,799	5.8e05	605
rule ID	25% (s)	50% (s)	max (s)
2502	841.0	2,443	12,242,553.0
5758	4152.5	9736.0	10,211,641.0
5706	3835.25	31975.5	14,345,406.0
5703	856	1,123	11,306,408

횟수가 많은 상위 4개 공격 규칙의 I^b를 보여준다.

공격 규칙 2502와 5703은 5758과 5706에 비해 I^b가 낮으므로 공격의 강도가 높은 것으로 판단된다. 표 9를 보면 공격자들은 공격 규칙 2502를 가장 많이 사용했음을 알 수 있다. 공격 규칙 2502는 무차별 비밀번호 대입 공격, 5706은 SSHD 스캔 공격, 5703은 Break-in 의심 공격이다. 표 9에는 명시하지 않았지만, 공격 규칙 5551과 5712는 I^b의 $Q2$가 각각 689초와 743초로써 공격의 강도가 매우 높았다. 본 연구에서는 공격이 탐지되면 600초 동안 공격자를 차단하므로 $Q2$ 값 689초는 차단이 해제된 후 89초 만에 다시 차단되었음을 의미한다.

그림 4는 HTTP 서비스를 공격하기 위해 사용한 일부 공격 규칙별 I^b를 보여준다. 그림의 좌측과 우측은 각각 교차 공격과 비교차 공격을 의미한다. 교차 공격의 차단 간격이 비교차 공격의 차단 간격보다 낮은 것으로 보아 하나의 공격자가 다수의 피공격자를 공격하는 것을 알 수 있다. 특히, 공격 규칙 31104과 31151을 사용하는 공격은 교차 공격이 많을 것으로 추측된다.

표 10을 보면 공격 규칙 31104와 31151의 교차 공격이 많음을 수치로 확인할 수 있다. 예를 들어, 공격 규칙 31104는 교차 공격의 차단 횟수가 1,201회로써 비교차 공격의 차단 횟수인 878회 보다 많은 것을 알 수 있다. 특히, 공격 규칙 31151과 31103 및 31164의 교차 공격은 제1사분위값($Q1$)이 600초 미만으로 측정되어 공격의 강도가 매우 높았다. 공격 규칙 31104는 일반적인 웹 공격이고 31151은 웹 스캔(다수의 HTTP 404 오류 유발) 공격이며 31103은 SQL 주입 공격이다.

공격 규칙 31103을 사용하는 교차 공격은 공격 강도는 매우 높지만 지속 시간은 다른 공격 규칙에 비해 상대적으로 짧은 특징을 보였다. 공격이 탐지되면 600초 동안 공격을 차단하므로 하나의 공격자가 다수의 피공격자를 공격하면 차단 간격이 600초 미만으로 표시될 수 있다. 표 7의 C^v 값을 비교해 종합하면 HTTP 서비스에 대한 공격은 일반적으로 비교차 공격이 많지만, 특정 공격 규칙을 사용하는 일부 공격자들에 의해 다수의 교차 공격이 발생한다고 볼 수 있다.

표 11은 Mail 서비스에 대한 공격 규칙별 교차 및 비교차 공격의 통계를 보여준다. Mail 서비스의 경우, 공격 규칙 3302와 3357에서 교차 공격과 비교차 공격의 차단 횟수가 동일했다. 즉, 하나의 공격자가 여러 개의 피공격자를 공격하는 교차 공격이 해당 규칙을 사

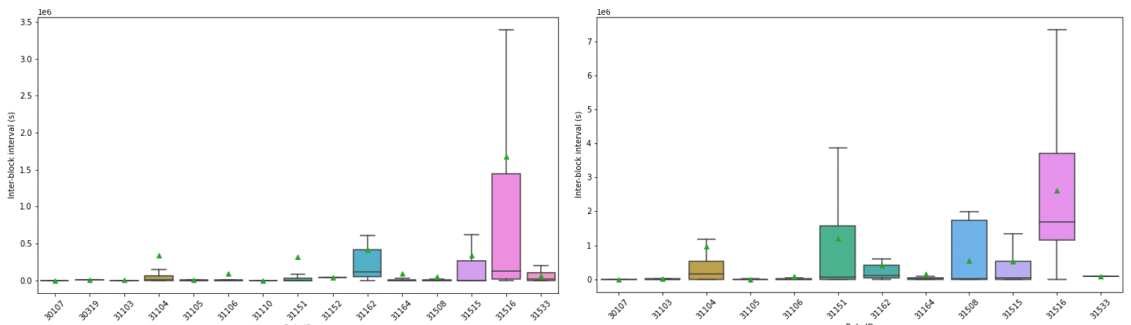

그림 4. HTTP에 대한 교차 및 비교차 공격의 공격 차단 간격(좌: 교차 공격, 우: 비교차 공격).
Fig. 4. Inter-block intervals of cross and non-cross attacks (left: cross attacks) for HTTP.

표 10. HTTP에 대한 공격 규칙별 공격 차단 간격(초)
Table 10. Inter-block intervals (seconds) per rule ID (HTTP)

rule ID	type	blocks	mean (s)	min (s)	25% (s)	50% (s)	75% (s)	max (s)
31104	cross	1201	337,410	0	662	7,255	60,689	14,604,050
	non-cross	878	968,812.6	609	4,587	152,368	517,945.25	18,116,492
31151	cross	1073	316,712.7	0	324	677	32,401	15,429,430
	non-cross	545	1,197,407	617	636	55,841	1,567,415	14,118,170
31516	cross	247	1,682,943	0	19,102	122,681	1,441,129	24,716,554
	non-cross	109	2,624,715	4,086	1,144,602	1,688,190	3,708,058	12,056,000
31515	cross	162	344,437	0	640	769	262,050	6,467,853
	non-cross	126	538,504	637	745.25	52,171.0	539,052.50	7,229,190
31103	cross	120	10,566.99	0	1	47.5	869	522,323
	non-cross	105	17,410	651	2,274	5,890	11,326	522,323

표 11. Mail에 대한 공격 규칙별 공격 차단 간격(초)
Table 11. Inter-block intervals (seconds) per rule ID (Mail)

rule ID	type	blocks	mean (s)	min (s)
3301	cross	77	8.2e+5	12
	non-cross	75	8.5e+5	2,335
	type	25% (s)	50% (s)	max (s)
3301	cross	6,266	9,810	2.7e+7
	non-cross	9,705	11,347	2.7e+7

용하는 공격에서는 발생하지 않았다. 공격 규칙 3301에서는 교차 공격이 발견되었지만 I^b가 100분 이상인 것으로 보아 공격의 강도가 낮은 것으로 판단된다.

VI. 제안 및 평가

본 장에서는 앞 장에서 분석한 공격 특성을 바탕으로 MALP 알고리즘을 제안하고 컴퓨터 시뮬레이션을 통해 알고리즘의 성능을 평가한다.

6.1 MALP 알고리즘

MALP 알고리즘은 그림 5와 같이 동작한다. 공격에 대한 방어 또는 공격자에 대한 차단 시간을 설정하기 위한 기본 전략은 더 공격할수록 더 긴 시간 동안 공격자를 차단하는 것이다. 지속적으로 공격하는 공격자에게 불이익을 주기 위해 차단이 해제되는 순간부터 일정 시간 동안 벌칙 시간(T^p)을 갖게 하고 벌칙 시간에 추가적인 공격이 탐지되면 공격 차단 시간을 직전의 차단 시간보다 길게 가져간다.

그림 5의 차단 시간 T_2는 이전 라운드의 벌칙 시간 T_1^p 동안 공격을 받지 않았으므로 T_1과 동일한 차단

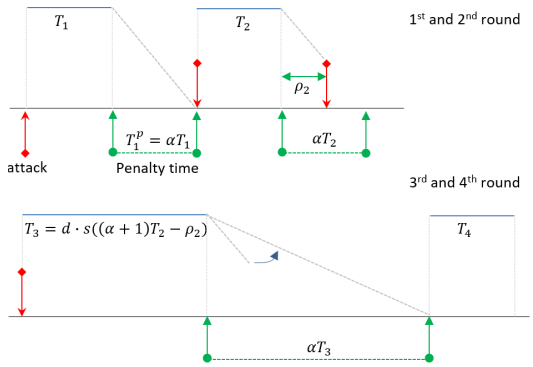

그림 5. 공격 차단 시간의 설정.
Fig. 5. Configuration of attack blocking time.

시간을 갖는다. 하지만, 이전 라운드의 벌칙 시간에 공격을 받은 T_3는 이전 라운드에서 남은 벌칙 시간 $\alpha T_2 - \rho_2$를 이전 라운드의 차단 시간 T_2에 추가하고 공격의 심각도 d와 공격의 강도 s를 가중해 차단 시간을 설정한다.

$$T_n = \begin{cases} \text{if } \kappa_{n-1} = 0, \\ \quad \max\limits_{s \in S_{n-1},\, d \in D_{n-1}} [s \cdot d \cdot T_0] \\ \text{if } \kappa_{n-1} = 1, \\ \quad \max\limits_{s \in S_{n-1},\, d \in D_{n-1}} [s \cdot d((\alpha+1)T_{n-1} - \rho_{n-1})] \end{cases} \quad (1)$$

n번째 라운드에서 공격 차단 시간 T_n을 설정하는 방법은 식(1)과 같다. κ_{n-1}은 T_{n-1}^p 시간 동안 공격이 탐지되었는지를 나타낸다. S_{n-1}과 D_{n-1}은 T_{n-1}^p까지 공격자가 가한 공격의 d와 s의 집합이다. 공격의 강도

s는 공격 규칙별 I^b의 $Q1$을 기준으로 $3T_0$ 미만(1,800초/0.5시간)은 3, $3T_0$에서 $6T_0$(3,600초/1시간) 미만은 2, $6T_0$ 이상은 1로 설정했다. 차단 시간의 기본값인 T_0는 OSSEC의 차단 시간과 동일하게 600초로 설정했다. $Q2$ 이상의 I^b는 공격의 휴지기가 통계에 포함되어 있을 가능성이 있으므로 $Q1$ 값을 이용했다.

지속성을 갖는 공격 규칙의 경우, 심각도는 OSSEC의 심각도를 기준으로 6, 8, 10 등 세 가지 경우만 확인되었으므로, 심각도 d는 6부터 7은 1, 8에서 9는 2, 10 이상은 3으로 구분했다. 본 연구에서는 d와 s의 수준을 각각 세 개의 등급으로 구분하고 d와 s에 동일한 가중치를 적용했지만, 성능의 최적화를 위해 등급을 다르게 설정할 수도 있다. 알고리즘을 단순화하기 위해 등급 구분을 기준으로 d와 s를 설정했다. 마지막으로 계수 α는 1로 설정했다.

표 12는 일부 공격 규칙의 d와 s를 보여준다. MALP 알고리즘의 방화벽 적용 즉, 공격자가 탐지되면 피공격자를 구분하지 않고 ACL 규칙을 설정하는 것이 목표이므로 s는 교차 공격의 통계를 활용했다. 고유 공격자가 교차 공격을 가할 경우, 피공격자를 구분해 ACL 규칙을 설정하면 피공격자의 수만큼 필터 규칙이 필요하다.

HTTP나 Mail 서비스에 대한 공격은 단수 차단 즉, 교차 공격과 비교차 공격에서 일회성 공격은 각각 전체의 28.9%와 54.4%를 차지하므로 긴 차단 시간은 방화벽 자원의 낭비를 의미한다.

표 12. 공격의 심각도(d)와 강도(s)
Table 12. Attack severity (d) and strength (s)

service	rule ID	d	s (non-cross)	s (cross)
http	31151	3	3	3
	31515	1	3	3
	31104	1	2	3
	31508	1	1	3
ssh	5551	3	3	3
	5701	2	2	2
	5720	3	1	1
	5758	2	1	1
mail	3357	3	1	1
	3353	3	1	1
	3355	3	1	1

6.2 성능 평가

MALP 알고리즘의 성능을 평가하기 위해 Python 시뮬레이터를 개발하고 제4장에서 설명한 CSV 파일을 이용해 이벤트 기반(Event-driven)의 모의실험을 수행했다. OSSEC 서버가 공격자를 탐지하면 모든 목적 서버에게 차단 명령을 내리는 방식과 MALP의 성능을 비교했다.

표 13. 성능 비교
Table 13. Performance comparison

	Block actions	Skipped block actions
OSSEC	78,899	2,322
MALP	14,815	64,290

표 13은 결과를 보여준다. 차단 명령을 생략한 횟수는 피공격자의 목적 서버에만 선택적으로 차단 명령을 내리는 방식과 비교했다. 차단 명령을 내린 횟수나 생략한 횟수와 무관하게 MALP은 OSSEC에서 차단한 모든 공격자들을 동일하게 차단한다.

OSSEC의 경우에는 총 78,899회 공격 차단 명령이 내려졌다. MALP은 14,815회의 차단 명령만으로도 OSSEC과 동일한 수의 공격을 차단하므로 약 81.22% 효과적으로 동작했음을 알 수 있다. 공격 차단 시간이 길어지면 차단 명령을 내린 횟수가 줄어들게 되는데, 이는 하나의 차단 명령으로 더 많은 공격을 방어할 수 있다는 의미이다. OSSEC의 경우에 생략할 수 있는 공격 차단 명령의 수는 2,322개였지만 MALP은 64,290회로 약 3.3 크기 정도(Order of magnitude)로 증가했다.

그림 6은 공격 규칙별 MALP 알고리즘의 최대 차단 횟수와 차단 명령의 최대 생략 횟수 및 최장 차단 시간을 보여준다. 차단 명령을 내리는 횟수가 적고 차단 명령을 생략한 횟수가 많을수록 알고리즘이 효과적으로 동작한 것으로 볼 수 있다. MALP가 HTTP 서비스보다는 SSHD 서비스에 대한 공격 차단에 효과적으로 동작하고 있다. 이는 SSHD 서비스에 대한 공격의 강도와

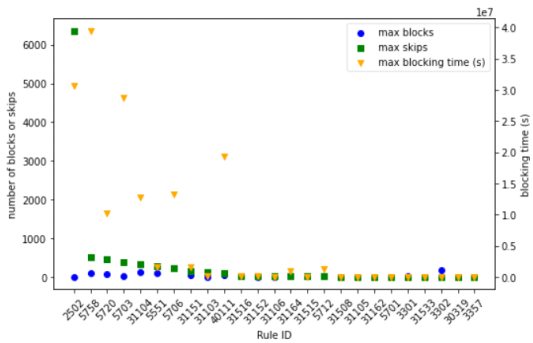

그림 6. 공격 규칙별 MALP 성능.
Fig. 6. MALP performance per rule ID.

지속성이 HTTP에 비해 높고 총 공격의 양도 SSHD가 많기 때문이다.

가장 오랫동안 차단된 공격자는 공격 규칙 5758을 사용했으며 약 455.39일간 차단되었다. OSSEC에서 정의한 공격 규칙 5758의 심각도는 8이고 허용되는 인증 실패 횟수를 초과했을 때 발생한다. HTTP 서비스의 경우, 공격 규칙 31104와 31103 등에서 효과적으로 대응했다. 두 공격 규칙은 은닉 정보를 탈취하기 위한 공격으로 분류할 수 있다.

공격 차단의 최대 횟수와 공격 차단 명령의 최대 생략 횟수의 비율(ψ)이 1 이하로 측정된 공격 규칙은 3301, 3302, 5706 등이었다. 해당 공격 규칙을 사용하는 공격은 데이터 특성으로 I^b를 활용하는 것이 비효과적일 수 있다. 공격 규칙 3301과 3302는 Mail 서비스에 대한 공격이고 5706은 SSH 스캔 공격으로써 고유 공격자로부터의 공격이 산발적으로 일어나거나 공격 시간이 짧을 것으로 예상된다. 이에 비해 공격 규칙 5758, 5551, 5703, 2502, 31103, 31152 등은 ψ가 3 이상으로 측정되어 MALP 알고리즘이 효과적으로 동작했다.

그림 7은 시뮬레이션 환경에서 제안한 알고리즘이 공격 차단을 위해 사용한 ACL 규칙의 수를 보여준다. MALP 알고리즘을 사용하면, 최대 66개의 ACL 규칙으로 공격자를 방어할 수 있었다.

SSHD 서버를 개방한 7월 이후부터는 사용한 필터 규칙의 수가 증가하는 것을 알 수 있다. SSHD 서비스에 대한 공격의 지속성이 높아 일부 공격자들에 대해서 차단 시간이 길어졌기 때문이다. 본 연구에서는 하나의 SSHD 서버에서 수집한 로그만 활용했으므로 여러 SSHD 서버의 개방이 필터 규칙의 크기 변화에 주는 영향은 알 수 없다. 만약 개방되는 SSHD 서버의 수와 필요한 필터 규칙의 수가 비례한다면, 천 개 미만의 필터 규칙으로 총 13개 서버에 대한 공격을 차단할 수 있을 것으로 예상된다.

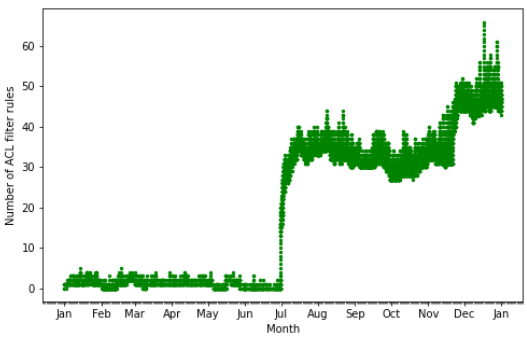

그림 7. 설치된 ACL 규칙의 수.
Fig. 7. Number of installed ACL rules.

Ⅶ. 결 론

본 논문은 소규모 방화벽에서 발생할 수 있는 ACL 규칙의 저장 공간 부족 문제와 ACL 규칙의 증가로 인한 성능 저하 문제를 완화하기 위해 공격 차단 시간을 동적으로 설정할 수 있는 MALP 알고리즘을 제안했다. 모의 환경에서 수행된 시뮬레이션 결과, 제안된 알고리즘이 공격 차단 명령을 내린 횟수 측면에서 OSSEC HIDS보다 81.22% 더 효과적으로 동작함을 확인했다. 또한 최대 66개의 ACL 규칙만으로도 공격자, 특히 지속성이 높은 공격을 차단할 수 있어, 소규모 방화벽에서 발생할 수 있는 저장 공간의 부족 문제와 성능 저하 문제를 완화할 수 있을 것으로 기대된다.

References

[1] Verizon Enterprise Solution, "*2020 Data Breach Investigations Report*," Retrieved Apr. 29, 2024, from https://www.verizon.com/business/resources/reports/2020-data-breach-investigations-report.pdf

[2] A. Voronkov, L. A. Martucci, and S. LindSkog, "Measuring the usability of firewall rule sets," *IEEE Access*, vol. 8, pp. 27106-27121, Feb. 2020. (https://doi.org/10.1109/ACCESS.2020.2971093)

[3] D. Melkov, A. Šaltis, and Š. Paulikas, "Performance testing of Linux firewalls," in *Proc. 2020 IEEE eStream*, pp. 1-4, Apr. 2020. (https://doi.org/10.1109/eStream50540.2020.9108868)

[4] J. K. Lee, T. Hong, and G. Li, "Traffic and overhead analysis of applied pre-filtering ACL firewall on HPC service network," *J. Commun. and Netw.*, vol. 23, no. 3, pp. 192-200, 2021. (https://doi.org/10.23919/JCN.2021.000011)

[5] L. Wusteney, M. Menth, R. Hummen, and T. Heer, "Impact of packet filtering on time-sensitive networking traffic," in *Proc. 2021 17th IEEE Int. Conf. Factory Commun. Syst. (WFCS)*, pp. 59-66, Jun. 2021.

(https://doi.org/10.1109/WFCS46889.2021.9483611)

[6] S. Applebaum, T. Gaber, and A. Ahmed, "Signature-based and machine-learning-based web application firewalls: A short survey," *Procedia Comput. Sci.*, no. 189, pp. 359-367, 2021. (https://doi.org/10.1016/j.procs.2021.05.105)

[7] O. Chakir, Y. Sadqi, and Y. Maleh, "Evaluation of open-source web application firewalls for cyber threat intelligence," in *Big Data Analytics and Intell. Syst. for Cyber Threat Intell.*, pp. 35-48, River Publishers, 2023.

[8] D. Cid, A. Hay, and R. Bray, "OSSEC host-based intrusion detection guide," *Syngress*, 2008.

[9] D. B. Cid, "*Log Analysis using OSSEC*," Retrieved Apr. 29, 2024, from http://www.academia.edu/8343225/Log_Analysis_using_OSSEC

[10] Z. Chen, J. Liu, W. Gu, Y. Su, and M. Lyu, "Experience report: Deep learning-based system log analysis for anomaly detection," *arXiv preprint, arXiv:2107.05908*, 2021. (https://doi.org/10.48550/arXiv.2107.05908)

[11] A. Farzad and T. A. Gulliver, "Unsupervised log message anomaly detection," *ICT Express*, vol. 6, no. 3, pp. 229-237, 2020. (https://doi.org/10.1016/j.icte.2020.06.003)

[12] L. Ceragioli, P. Degano, and L. Galletta, "Can my firewal system enforce this policy?," *Comput. & Security*, vol. 117, 2022. (https://doi.org/10.1016/j.cose.2022.102683)

[13] A. Sahu, P. Wlazlo, N. Gaudet, A. Goulart, E. Rogers, and K. Davis, "Generation of firewall configurations for a large scale synthetic power system," in *Proc. 2022 IEEE TPEC*, pp. 1-6, Feb. 2022. (https://doi.org/10.1109/TPEC54980.2022.9750776)

[14] V. H. Le and H. Zhang, "Log-based anomaly detection without log parsing," in *Proc. 36th IEEE/ACM Int. Conf. Autom. Software Eng.*, pp. 492-504, Nov. 2021. (https://doi.org/10.1109/ASE51524.2021.9678773)

[15] J. Zhu, S. He, J. Liu, P. He, Q. Xie, Z. Zheng, and M. R. Lyu, "Tools and benchmarks for automated log parsing," in *Proc. 2019 IEEE/ACM 41st ICSE-SEIP,* pp. 121-130, May 2019. (https://doi.org/10.48550/arXiv.1811.03509)

[16] F. A. Vadhil, M. F. Nanne, and M. L. Salihi, "Importance of machine learning techniques to improve the open source intrusion detection systems," *Indonesian J. Electr. Eng. and Inf. (IJEEI)*, vol. 9, no. 3, pp. 774-783, 2021. (https://doi.org/10.52549/ijeei.v9i3.3219)

[17] J. Park, J. Kim, B. B. Gupta, and N. Park, "Network log-based SSH brute-force attack detection model," *Comput., Materials & Continua*, vol. 68, no. 1. pp. 887-901, Mar. 2021. (https://doi.org/10.32604/cmc.2021.015172)

[18] J. S. Goodgion, "Active response using host-based intrusion detection system and software-defined networking," *Theses and Dissertations*, 1575, 2017.

[19] MaxMind, Retrieved Apr. 29, 2024 from https://dev.maxmind.com/geoip/geolite2-free-geolocation-data.

[20] A. Navlani, A. Fandango, and I. Idris, "*Python Data Analysis: Perform data collection, data processing, wrangling, visualization, and model building using Python*," Packt Publishing Ltd., 2021.

조 진 용 (Jinyong Jo)

2013년 : 광주과학기술원 정보통신공학과 박사
2003년~현재 : 한국과학기술정보연구원 책임연구원
2016년~현재 : 국제인증연합(eduGAIN) 운영그룹 위원

<관심분야> Trust and Identity, Networked applications and services
[ORCID:/0000-0001-6830-3604]

김 승 해 (Seung-Hae Kim)

2008년 : 전북대학교 정보보호공학 박사
1996년~현재 : 한국과학기술정보연구원 책임연구원
2021년~현재 : 한국과학기술정보연구원 연구망서비스팀 팀장

<관심분야> 라우팅 프로토콜 보안, 인증, 망관리, 정보보호
[ORCID:0000-0002-8403-7577]

박 주 원 (Ju-Won Park)

2010년 8월 : 광주과학기술원 정보기전공학부 박사
2010년 9월~2013년 7월 : KT 유무선네트워크 연구소
2013년 8월~현재 : 한국과학기술정보연구원 책임연구원

<관심분야> 고성능 컴퓨팅, 클라우드 컴퓨팅
[ORCID:0000-0003-1388-1583]

조 부 승 (Buseung Cho)

2017년 : 성균관대학교 컴퓨터공학 박사
2005년~현재 : 한국과학기술정보연구원
2018년~현재 : 과학기술연합대학원대학교 데이터 및 HPC 과학 부교수

<관심분야> 소프트웨어 정의 네트워크, 네트워크 관리
[ORCID:0000-0002-4661-5700]