The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 23, No. 3, pp.119-125, Jun. 30, 2023. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2023.23.3.119
JIIBC 2023-3-17

이기종 네트워크 장치를 사용하는 시스템의 효율적인 관리를 위한 로그 수집 방법

Log Collection Method for Efficient Management of Systems using Heterogeneous Network Devices

양재호*, 김영곤**

Jea-Ho Yang*, Younggon Kim**

요 약 IT 인프라 운영이 고도화하면서 시스템을 관리하는 방식이 널리 보급되어 있으며, 최근에는 Syslog를 활용한 개선방법들이 연구되고 있다. 그러나 이러한 방법으로 수집한 로그 데이터를 활용하여 시스템 관제를 할 경우 다양한 형식으로 추출되는 로그를 전문 인력이 분석해야 하는 어려움이 있다. 본 논문은 엣지 컴퓨팅을 활용하여 Syslog 데이터를 분산 수집하고 중복 데이터를 전처리하여 중앙 데이터베이스에 적재하는 시스템을 구축 방법을 제시하고자 한다. 또한, 데이터사전을 구성하여 실시간으로 데이터를 분류하고 카운팅하는 기능을 제공하며, 데이터사전에 등록된 데이터에 대해서는 중앙 데이터베이스로의 전송을 제한하는 시스템을 구현한다. 이를 통해 데이터 사전의 정의어 패턴을 유지하며, 중복 데이터와 시간 중복을 제어하여 중앙 데이터베이스에 정제된 데이터를 적재함으로써 빅데이터 분석을 위한 기초 자료를 확보할 수 있다. 시뮬레이션결과 제안된 알고리즘과 프로시저를 구체적인 예시와 함께 설명하고, syslog 데이터를 활용하여 그 성능을 검증하였다. syslog 데이터는 실제 로그 데이터에서 추출한 예시를 포함하고 있으며 이를 통해 로그 데이터로부터 필요한 정보를 정확하게 추출하였고, 분류 및 적재 과정에서 정상적인 처리가 이루어지는지를 확인하였다. 이러한 시스템은 엣지 환경에서 로그 데이터를 효율적으로 수집하고 관리하기 위한 솔루션으로 활용하여 기술의 확산 측면에서도 효과를 기대할 수 있다.

Abstract IT infrastructure operation has advanced, and the methods for managing systems have become widely adopted. Recently, research has focused on improving system management using Syslog. However, utilizing log data collected through these methods presents challenges, as logs are extracted in various formats that require expert analysis. This paper proposes a system that utilizes edge computing to distribute the collection of Syslog data and preprocesses duplicate data before storing it in a central database. Additionally, the system constructs a data dictionary to classify and count data in real-time, with restrictions on transmitting registered data to the central database. This approach ensures the maintenance of predefined patterns in the data dictionary, controls duplicate data and temporal duplicates, and enables the storage of refined data in the central database, thereby securing fundamental data for big data analysis. The proposed algorithms and procedures are demonstrated through simulations and examples. Real syslog data, including extracted examples, is used to accurately extract necessary information from log data and verify the successful execution of the classification and storage processes. This system can serve as an efficient solution for collecting and managing log data in edge environments, offering potential benefits in terms of technology diffusion.

Key Words : Infrastructure operations, Syslog, Data dictionary, Duplicate data, Big data analysis

*정회원, 한국공학대학교 컴퓨터공학과
**정회원, 한국공학대학교 컴퓨터공학과
접수일자 2023년 5월 29일, 수정완료 2023년 6월 5일
게재확정일자 2023년 6월 9일

Received: 29 May, 2023 / Revised: 5 June, 2023 /
Accepted: 9 June, 2023
**Corresponding Author: ykkim@tukorea.ac.kr
Dept of Computer Engineering, Tech University of Korea, Korea

Ⅰ. 서 론

최근 정보 시스템의 복잡성과 다양성이 증가함에 따라 로그 데이터의 중요성이 크게 부각되고 있다. 로그 데이터는 시스템의 동작과 상태에 대한 흔적을 담고 있으며, 이를 효과적으로 분석하고 활용함으로써 시스템의 안정성, 유지보수 용이성 문제 해결 등에 기여할 수 있다. 그러나 대량의 로그 데이터를 다루는 것은 복잡하고 번거로운 과정을 요구하며, 이로 인해 로그 데이터의 분석 및 활용에 어려움을 겪는 경우가 많다[1][5].

이에 따라 로그 데이터의 전처리 및 적재 과정은 매우 중요한 작업으로 간주되고 있으며 전처리 과정에서는 로그 데이터로부터 필요한 정보를 추출하고, 데이터의 일관성을 확보하기 위해 변환 작업을 수행한다. 적재 과정에서는 전처리된 로그 데이터를 목적지 테이블에 적절하게 저장함으로써 데이터의 구조화와 분류를 실시한다. 이러한 전처리 및 적재 과정은 로그 데이터의 분석, 모니터링, 보고 등 다양한 활용을 위한 기반을 마련하는 핵심 작업이다[2].

기존의 로그 데이터 전처리 및 적재 방식은 수동적이고 비효율적인 특징을 가지고 있으며, 로그 데이터의 복잡성과 다양성에 따른 처리 과정에서 오류가 발생할 수 있는 문제점이 있다. 이에 따라 본 연구에서는 로그 데이터의 효과적인 전처리 및 적재를 위한 알고리즘과 프로시저를 제안한다. 제안된 알고리즘은 로그 데이터의 구조화와 의미 해석을 정확하게 수행하기 위해 다양한 처리 단계를 포함하고 있다. 프로시저는 제안된 알고리즘을 실제로 구현하고 로그 데이터의 자동 처리를 위한 실행 단계를 제시한다.

본 연구에서는 제안된 알고리즘과 프로시저를 구체적인 예시와 함께 설명하고, syslog 데이터를 활용하여 그 성능을 검증한다. syslog 데이터는 실제 로그 데이터에서 추출한 예시를 대표적으로 포함하고 있으며 이를 통해 로그 데이터로부터 필요한 정보를 정확하게 추출하고, 분류 및 적재 과정에서 정상적인 처리가 이루어지는지를 확인한다.

결과적으로, 제안된 알고리즘과 프로시저는 로그 데이터의 전처리 및 적재 과정을 효율적이고 정확하게 수행할 수 있는 방법을 제시한다. 이를 통해 로그 데이터의 관리와 활용을 간소화하고, 시스템의 안정성과 적시성을 향상시킬 수 있다. 또한, 본 연구 결과는 로그 분석 및 빅데이터 분야에 기여하며, 다양한 산업 분야에서의 로그 데이터 관리에 대한 지침과 기법을 제시하는데 활용될 것으로 기대한다.

Ⅱ. 관련 연구

1. SYSLOG

Syslog은 네트워크 장비에서 발생하는 로그 메시지를 수집, 기록, 분석하는 데 사용되는 프로토콜 및 서비스입니다. 네트워크 장비는 라우터, 스위치, 방화벽, 서버 등 다양한 장치를 포함한다.

시스템 및 네트워크 이벤트에 대한 정보를 로그 형태로 기록한다. 예를 들어, 장비의 동작 상태, 경고, 오류, 인증 문제 등 다양한 이벤트에 대한 로그를 생성한다. 이 로그는 보안, 문제 해결, 성능 모니터링 및 장애 조치와 같은 다양한 용도로 활용된다[2].

일반적으로 장비에서 생성된 로그를 중앙 집중화된 서버 또는 저장소로 전송한다. 이를 통해 여러 장비에서 발생하는 로그를 통합하여 중앙에서 관리하고 분석할 수 있지만 대규모 환경에는 적합하지 않다. 또한, Syslog은 로그 메시지에 기본적인 메타데이터를 포함하며, 이는 로그의 발생 시간, 이벤트의 중요도, 메시지의 소스 등을 나타낸다[1].

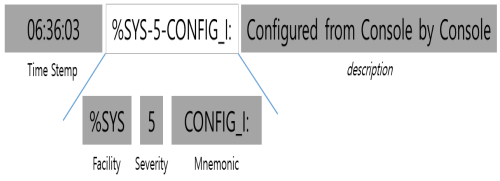

그림 1. Syslog의 유형
Fig. 1. Type of Syslog

Syslog은 IT 인프라 관리 및 보안 감사를 위한 핵심 도구로 사용되며, 문제 발견, 이벤트 추적, 오류 해결 등에 큰 도움을 준다. 네트워크 장비에서 발생하는 로그를 효율적으로 수집하고 분석함으로써 시스템의 안정성과 유지보수를 용이하게 할 수 있다[2].

하지만 메시지의 분류가 운영체제의 종류에 따라 다르고 펌웨어를 제공하는 하드웨어 제조사 마다 정해진 규격 없어 장애 분석 시 효율성이 떨어진다[1].

2. 기존의 로그수집시스템

기존의 로그 수집 시스템은 네트워크 장비 및 서버에

서 발생하는 다양한 형식의 로그 이벤트를 실시간으로 수집하고 중앙 집중화된 서버로 전송하여 관리하는 시스템이다. 이를 통해 시스템 상태 모니터링, 보안 감시, 문제 해결 등에 필요한 정보를 추출하고 활용할 수 있었다. 그러나 기존의 로그 수집 시스템은 로그 데이터 처리의 복잡성과 다양한 로그 형식에 대한 분석 어려움 등의 한계가 있다[4].

첫째, 각각의 로그 형식은 고유한 필드와 포맷을 가지고 있으며, 이를 효과적으로 추출하고 해석하기 위해서는 복잡한 데이터 처리 및 변환 작업이 필요하다. 이는 로그 분석 및 관리 과정에서 추가적인 작업과 리소스를 요구하며, 시스템 관리자에게 부담을 줄 수 있다.

둘째, 각 장비나 시스템은 고유한 로그 형식을 사용하며, 이로 인해 로그 데이터의 통합 분석이 어려워지고 형식이 서로 다르기 때문에 통합된 분석과 보고서 작성이 어렵고, 이로 인해 안정성 및 문제 해결 등의 작업에 어려움을 겪을 수 있다[2].

셋째, 기존의 로그 수집 시스템은 대규모 환경에서 발생하는 대용량 로그 데이터를 처리하는 데 한계가 있다. 로그 데이터의 양이 증가함에 따라 수집, 저장, 분석, 검색 등의 작업에 필요한 처리 성능과 자원이 부족할 수 있습니다. 이로 인해 실시간 모니터링이 지연되거나 데이터 분석이 제한되는 경우가 발생할 수 있다[3][6].

이러한 한계들은 기존의 로그 수집 시스템이 신뢰성과 효율성 면에서 개선의 필요성을 보여준다.

본 논문은 엣지 컴퓨팅을 활용하여 Syslog 데이터를 분산 수집하고 중복 데이터를 전처리하여 중앙 데이터베이스에 적재하는 시스템을 구축 방법을 제시하고자 한다.

Ⅲ. 본 문

1. 이기종 네트워크장치 로그 수집 아키텍처

이기종 네트워크장치 로그 수집 아키텍처는 다양한 구성 요소와 기능을 포함하고 있다, 구성 요소로는 분산 수집 서버, 중앙 데이터베이스, 데이터사전, 그리고 관리 및 모니터링 인터페이스 등이 있다.

분산 수집 서버는 수집된 로그 데이터를 중앙 데이터베이스에 적재하기 전에 전처리하는 역할을 수행하며 중복 데이터를 제거하거나 데이터의 분류와 카운팅을 위한 데이터사전을 구성하고 실시간으로 데이터를 처리한다. 이 과정에서 데이터사전에 등록된 데이터에 대해서는 중앙 데이터베이스로의 전송을 제한할 수 있는 중복방지 기능을 제공한다.

중앙 데이터베이스는 전체 로그 데이터를 저장하고 관리하는 중심적인 역할을 수행한다. 전처리된 로그 데이터는 중앙 데이터베이스에 적재되어 빅데이터 분석을 위한 기초 자료로 활용된다. 이를 통해 데이터 사전의 정의어 패턴을 유지하고 중복 데이터와 시간 중복을 제어하여 정제된 데이터를 확보할 수 있다.

또한, 이기종 네트워크장치 로그 수집 아키텍처는 관리 및 모니터링 인터페이스를 제공한다. 이를 통해 관리자는 로그 수집 상태를 실시간으로 모니터링하고 로그 데이터에 대한 검색 및 조회를 수행할 수 있다. 또한, 보안 이벤트 발생 시 즉각적인 대응이 가능하도록 경고 및 알림 기능을 제공한다.

이와 같은 이기종 네트워크장치 로그 수집 아키텍처는 다양한 종류와 규모의 네트워크 환경에서 로그 데이터를 체계적으로 수집하고 관리하기 위한 효율적인 방법을 제공한다. 이를 통해 네트워크 성능, 보안 및 문제 해결에 대한 신속하고 정확한 대응이 가능해지며, 전체적인 IT 인프라의 안정성과 신뢰성을 향상시킬 수 있다.

제안된 로그의 수집 시스템의 프로세스는 아래 그림 1과 같다.

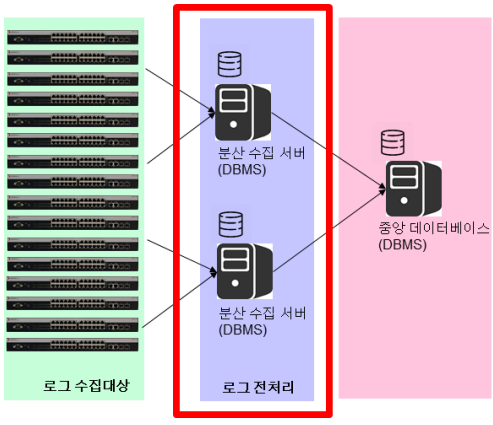

그림 2. 시스템 구성도
Fig. 2. system diagram

2. 전처리 및 적제 과정

분산 수집 서버가 운영되는 엣지 환경에서 Syslog 데이터의 효율적인 전처리와 저장은 매우 중요하다.

프로시저 "preprocessLogsScheduler()"는 로그 데이터의 처리와 전처리를 자동화하는 데 사용되는 코드이다. 이 프로시저는 원시 로그 데이터(raw_logs)를 읽어와 필요한 정보를 추출하고, 이를 기반으로 로그를 그룹

화하고 카운트한다. 그런 다음, 전처리된 로그 데이터를 새로운 테이블(preprocessed_logs)에 삽입하고, 1분 이전의 로그 데이터는 삭제하는 알고리즘을 고안하였고 그 동작하는 프로세스는 5단계로 구성된다.

가. 프로시저 시작 시, raw_logs 테이블과 preprocessed_logs 테이블을 초기화한다. 이 두 테이블은 로그 데이터의 원시 버전과 전처리된 버전을 저장하기 위해 사용된다.

나. log_counts라는 빈 딕셔너리를 생성한다. 이 딕셔너리는 로그의 종류별로 카운트 값을 저장하는 용도로 사용된다.

다. raw_logs 테이블에서 로그 항목을 하나씩 반복하면서 다음 작업을 수행한다.

(1) 각 로그 항목에서 필요한 정보를 추출한다. 이 정보에는 로그의 timestamp, facility, severity, mnemo nic, description 등이 포함된다.

(2) 추출한 정보를 기반으로 고유한 로그 키(log_key)를 생성한다. 로그 키는 로그 항목을 고유하게 식별하기 위해 사용된다.

(3) 생성한 로그 키가 log_counts 딕셔너리에 이미 존재하는지 확인한 후, 없을 경우 0으로 초기화한다.

(4) 해당 로그 키의 카운트 값을 1 증가시킨다.

라. 로그 키(log_key)를 기반으로 log_counts 딕셔너리를 반복하면서 다음 작업을 수행한다.

(1) 각 로그 키에서 필요한 정보를 추출한다.

(2) 추출한 정보와 해당 로그 키의 카운트 값을 preprocessed_logs 테이블에 삽입한다.

마. 현재 시간으로부터 1분 이전의 로그 데이터를 raw_logs 테이블에서 삭제한다.

이 프로시저를 통해 로그 데이터의 처리와 전처리 작업을 자동화할 수 있으며, 이를 통해 로그 데이터의 분석과 이해를 용이하게 할 수 있다.

스케줄러와 통합된 전처리 프로시저문 방식의 슈도코드는 아래 표 1과 같다.

표 1. 전처리 및 적제 과정 알고리즘

Table 1. Algorithm for preprocessing and loading process

```
PROCEDURE preprocessLogsScheduler()
BEGIN
    log_counts = EMPTY DICTIONARY;
    FOR EACH log_entry IN 'raw_logs' LOOP
        log_key
        = createLogKey(extractLogDetails(log_entry));
        IF log_key NOT IN log_counts THEN
        log_counts[log_key] = 0; END IF;
        log_counts[log_key] += 1;
    END LOOP;

    FOR EACH log_key IN log_counts.keys LOOP
        INSERT INTO 'preprocessed_logs'
        (timestamp, facility, severity, mnemonic,
        description, count)
        VALUES
        (extractLogDetailsFromKey
        (log_key).timestamp,
        extractLogDetailsFromKey
        (log_key).facility,
        extractLogDetailsFromKey
        (log_key).severity,
        extractLogDetailsFromKey
        (log_key).mnemonic,
        extractLogDetailsFromKey
        (log_key).description,
        log_counts[log_key]);
    END LOOP;

    DELETE FROM 'raw_logs' WHERE timestamp
    < CURRENT_TIMESTAMP - INTERVAL 1 MINUTE;
END;
```

3. 중복 전송을 제한하는 중복방지 과정

프로시저 mainprocessLogs()는 로그 데이터를 처리하고 관련 정보를 추출하여 메인 테이블에 적재하는 일련의 작업을 수행하는 프로세스이다. 이 프로세스는 데이터 사전을 구성하고 중복 제거를 자동화하며 주기적으로 실행되어 시스템의 로그 데이터를 효율적으로 관리하는 알고리즘을 고안하였고 그 동작하는 프로세스는 5단계로 구성된다.

가. preprocessed_table_name에서 전처리된 로그 데이터를 가져옵니다. 이 데이터에는 timestamp, facility, severity, mnemonic, description을 포함하고 있다.

나. 가져온 로그 데이터를 순차적으로 처리합니다. 각 로그에 대해 중복을 확인하고 중복 여부에 따라 처리를 분기한다. 만약 중복된 로그라면 해당 로그를 main_table_name에 삽입한다. 중복되지 않은 로그라면 해당 로그의 facility, severity, mnemonic, descrip tion을 데이터 사전역할을 하는 information_table_name에 삽입한 뒤, 해당 로그를 main_table_name에 삽입하고 이를 반복하여 전체 로그 데이터를 입력한다.

다. 처리가 완료된 로그 중에서 timestamp가 one_minute_ago보다 이전인 로그를 preprocessed_table_name에서 삭제한다. 이를 통해 1분 이전의 로그 데이터를 정리한다.

라. preprocessed_table_name의 내용을 초기화하여 다음 주기의 로그 처리를 위해 준비한다.

마. 마지막으로, 스케줄러를 사용하여 1분마다 processLogs() 프로시저를 주기적으로 실행한다. 이를 통해 로그 처리와 데이터 정리 작업을 자동화하고 지속적으로 수행할 수 있다.

이 프로시저를 사용하면 로그 데이터를 효율적으로 처리하고 중복을 제거하여 정확하고 일관된 데이터를 유지할 수 있다. 또한 스케줄러를 사용하여 주기적으로 실행되므로 데이터 정리 작업을 손쉽게 관리할 수 있다. 이러한 장점을 통해 시스템의 로그 데이터 관리 및 분석 작업을 향상시킬 수 있다.

중복 전송을 제한하는 알고리즘은 아래 표 2와 같다.

표 2. 중복 전송을 제한하는 중복방지 알고리즘

Table 2. An anti-duplication algorithm to limit duplicate transmissions

```
CREATE PROCEDURE mainprocessLogs() AS
BEGIN
  FOR log IN preprocessed_table_name LOOP
    IF (중복 확인) THEN INSERT INTO
    main_table_name (log);
    ELSE INSERT INTO information_table_name
    (facility, mnemonic, description); INSERT INTO
    main_table_name (log);
    END IF;
  END LOOP;
  DELETE FROM preprocessed_table_name
          WHERE timestamp < one_minute_ago;
  CLEAR preprocessed_table_name;
  SCHEDULE EVERY 1 MINUTE;
END;
```

Ⅳ. 실험 및 결과

1. 전처리 및 적재 과정 검증

raw_logs 테이블의 로그 항목인 timestamp, facility, severity, mnemonic, description를 이용해 각각의 값을 변환하고 정상적으로 분류하는 및 preprocessed _logs 테이블에 적재 과정 검증절차를 확인하였다. 각각의 log 데이터에 5개 변수인 Timestamp, Facility, seve rity, Mnemonic, description 값을 변경하여 전송시뮬레이션을 수행하여서 5개에 대해 3개의 샘플에 대하여 변이를 시켜서, Count 결과 각각 독립적인 개체로 인식하여 count 값이 1이고, 동일하게 인식했다면 count 가 3이되어야 한다. 우리는 각각이 독립적이서 count 값이 1로 인식하였기에 검증이 성공적이라고 판단된다. 시뮬레이션 결과는 아래 표 3과 같다.

표 3. 전처리 및 적재 과정 검증

Table 3. Verification of pre-treatment and loading process

① timestamp 검증
- raw_logs Table
Mar 1 18:46:11: %SYS-5-CONFIG_I: Configured from console by vty2
Mar 1 18:46:13: %SYS-5-CONFIG_I: Configured from console by vty2
Mar 1 18:46:14: %SYS-5-CONFIG_I: Configured from console by vty2
- preprocessed_logs Table
timestamp: Mar 1 18:46:11, facility: %SYS, Severity: 5, mnemonic: CONFIG_I, description: Configured from console by vty2, count: 1
timestamp: Mar 1 18:46:13, facility: %SYS, Severity: 5, mnemonic: CONFIG_I, description: Configured from console by vty2, count: 1
timestamp: Mar 1 18:46:14, facility: %SYS, Severity: 5, mnemonic: CONFIG_I, description: Configured from console by vty2, count: 1
② facility 검증
- raw_logs Table
Mar 1 18:46:11: %SYS-5-CONFIG_I: Configured from console by vty2
Mar 1 18:46:11: %SYR-5-CONFIG_I: Configured from console by vty2
Mar 1 18:46:11: %SYV-5-CONFIG_I: Configured from console by vty2
- preprocessed_logs Table
facility: Mar 1 18:46:11, facility: %SYS, Severity: 5, mnemonic: CONFIG_I, description: Configured from console by vty2, count: 1
facility: Mar 1 18:46:11, facility: %SYR, Severity: 5, mnemonic: CONFIG_I, description: Configured from console by vty2, count: 1
facility: Mar 1 18:46:11, facility: %SYV, Severity: 5, mnemonic: CONFIG_I, description: Configured from console by vty2, count: 1
③ severity 검증
- raw_logs Table
Mar 1 18:46:11: %SYS-5-CONFIG_I: Configured from console by vty2
Mar 1 18:46:11: %SYR-2-CONFIG_I: Configured from console by vty2
Mar 1 18:46:11: %SYV-3-CONFIG_I: Configured from console by vty2
- preprocessed_logs Table
severity: Mar 1 18:46:11, facility: %SYS, Severity: 5, mnemonic: CONFIG_I, description: Configured from console by vty2, count: 1
severity: Mar 1 18:46:13, facility: %SYS, Severity: 2, mnemonic: CONFIG_I, description: Configured from console by vty2, count: 1
severity: Mar 1 18:46:14, facility: %SYS, Severity: 3, mnemonic: CONFIG_I, description: Configured from console by vty2, count: 1
④ mnemonic 검증
- raw_logs Table
Mar 1 18:46:11: %SYS-5-CONFIG_I: Configured from console by vty2
Mar 1 18:46:11: %SYR-5-CONFID_I: Configured from console by vty2
Mar 1 18:46:11: %SYV-5-CONFIV_I: Configured from console by vty2
- preprocessed_logs Table
mnemonic: Mar 1 18:46:11, facility: %SYS, Severity: 5, mnemonic: CONFIG_I, description: Configured from console by vty2, count: 1

mnemonic: Mar 1 18:46:11, facility: %SYR, Severity: 5, mnemonic: CONFIG_D, description: Configured from console by vty2, count: 1
mnemonic: Mar 1 18:46:11, facility: %SYV, Severity: 5, mnemonic: CONFIG_V, description: Configured from console by vty2, count: 1
⑤ description 검증
- raw_logs Table
Mar 1 18:46:11: %SYS-5-CONFIG_I: Configured from console by vty2
Mar 1 18:46:11: %SYR-5-CONFIG_I: Configured from console by vty3
Mar 1 18:46:11: %SYV-5-CONFIG_I: Configured from console by vty4
- preprocessed_logs Table
description: Mar 1 18:46:11, facility: %SYS, Severity: 5, mnemonic: CONFIG_I, description: Configured from console by vty2, count: 1
description: Mar 1 18:46:11, facility: %SYS, Severity: 5, mnemonic: CONFIG_I, description: Configured from console by vty3, count: 1
description: Mar 1 18:46:11, facility: %SYS, Severity: 5, mnemonic: CONFIG_I, description: Configured from console by vty4, count: 1

2. 중복 전송을 제한하는 중복방지 검증

preprocessed_logs 테이블의 로그 항목 중 timestamp를 제외한 중복항목을 필터링하고 최종적으로 main_logs에 적재 과정 검증절차를 확인하였고 127개 line의 log data 로 시뮬레이션으로 수행한 결과 정상적인 중복 count로 인식하여 표 4와 같이 count 값이 각각 다른 값인 9,11,12,8,50,4,16,17로 인식되어 정상적으로 count를 성공하였다. 로그데이터의 시뮬레시션 결과는 아래 표 4과 같다.

표 4. 중복 전송을 제한하는 중복방지 검증 결과
Table 4. Result of anti-duplication verification to limit redundant transmission

- main_logs Table
Feb 8 04:00:48: %SEC-6-IPACCESSLOGRP: list 177 denied igmp 198.51.100.197 -> 224.0.0.22, 1 packet (Count: 9)
Feb 9 04:00:48: %SEC-6-IPACCESSLOGSP: list INBOUND-ON-F11 denied igmp 198.51.100.2 -> 224.0.0.2 (20), 1 packet (Count: 11)
Feb 10 04:00:48: %SEC-6-IPACCESSLOGNP: list 171 denied 0 198.51.100.1 -> 255.255.255.255, 1 packet (Count: 12)
May 3 19:11:33: %IPV6-6-ACCESSLOGP: list ACL-IPv6-E0/0-IN/10 permitted tcp 2001:DB8::3(1027) -> 2001:DB8:1000::1(22), 9 packets (Count: 8)
Jun 20 02:41:40: %SEC-6-IPACCESSLOGP: list 177 denied udp 198.51.100.195(55250) -> 198.51.100.255(15600), 1 packet (Count: 50)
Jun 20 02:41:45: %SEC-6-IPACCESSLOGDP: list 151 denied icmp 198.51.100.1 -> 198.51.100.2 (3/4), 1 packet (Count: 4)
Jun 20 02:41:56: %SEC-6-IPACCESSLOGP: list 150 denied tcp 198.51.100.12(59825) -> 172.217.10.46(80), 1 packet (Count: 16)
Jun 20 02:42:28: %SEC-6-IPACCESSLOGRL: access-list logging rate-limited or missed 18 packets (Count: 17)

V. 결 론

본 연구에서는 로그 전처리 및 적재 과정을 위한 알고리즘과 프로시저를 개발하였으며, 이를 syslog 데이터를 활용하여 검증하였고. 이를 통해 다음과 같은 결론을 도출할 수 있었다.

첫째, 개발한 알고리즘은 로그 데이터의 필드들을 정확하게 추출하고 변환하는 과정에서 효과적으로 작동함을 확인하였다. syslog의 timestamp, facility, severity, mnemonic, description 값을 정확하게 분리하고, 필요한 변환 작업을 수행하였다. 이를 통해 로그 데이터의 구조화와 의미 해석을 정확하게 수행할 수 있었다.

둘째, 프로시저를 통해 구현된 로그 전처리 및 적재 과정은 신뢰성 있게 동작함을 확인하였다. 알고리즘에 따라 변환된 로그 데이터는 정상적으로 분류되어 목적지 테이블에 적절하게 저장되었다. 또한, 분류된 로그 데이터의 일관성과 정확성을 확인하여 알고리즘의 성능을 평가하였고, 이를 통해 시스템의 로그 데이터 처리 효율성을 향상시킬 수 있음을 확인하였다.

마지막으로, 본 연구의 결과는 로그 관리와 빅데이터 분야에 유용한 지침과 기반을 제공함으로써 다양한 응용 분야에 활용될 수 있음을 시사한다. 로그 데이터의 효과적인 처리와 분석은 시스템의 안정성과 빅데이터에 적합한 기초데이터 강화에 기여할 수 있으며, 실제 환경에서의 적용 가능성과 확장성을 고려하여 알고리즘의 성능 향상과 최적화를 추구할 필요가 있다.

본 연구의 한계점은 syslog 데이터에 대한 한정성이 있으며, 실제 대규모 환경에서의 적용성과 성능 평가를 고려해야 한다. 또한, 더욱 다양한 로그 유형과 변환 규칙을 포함하는 확장된 알고리즘의 개발과 병렬 처리, 최적화 방법의 고려가 필요하다.

앞으로의 연구 방향으로는 실제 환경에서의 적용 가능성과 성능 향상을 위한 실험 및 평가를 수행할 예정이며 더욱 효율적이고 확장 가능한 알고리즘을 개발하기 위해 머신러닝 및 딥러닝 기법을 활용하고, 실시간 로그 분석 및 이상 탐지를 위한 연구를 진행할 것이다.

이 연구 결과는 로그 관리 및 빅데이터 분야에 기여하며, 로그 데이터의 효과적인 활용과 시스템 보호에 관련된 다양한 연구 및 산업 응용 분야에 유용한 지침과 기반을 제공할 것으로 기대한다.

References

[1] SHENGLIN ZHANG, YING LIU, WEIBIN MENG, JIAHAO BU, SEN YANG, YONGQIAN SUN, DAN PEI, JUN XU, YUZHI ZHANG, LEI SONG, MING ZHAN, 'Efficient and Robust Syslog Parsing for Network Devices in Datacenter Networks', Journal of IEEE, Vol. 8, No. 6, pp.30245-30261, February 2020.
DOI: 10.1109/ACCESS.2020.2972691

[2] Sizhe Rao, Minghui Wang, Cuixia Tian, Xin'an Yang, Xiangqiao Ao, 'A Hierarchical Tree-Based Syslog Clustering Scheme for Network Diagnosis', 17th International Conference on Network and Service Management (CNSM), pp.27-34, 2021.
DOI: 10.23919/CNSM52442.2021.9615506

[3] Satoru Kobayashi, Kazuki Otomo, Kensuke Fukuda, 'Causal analysis of network logs with layered protocols and topology knowledge', Journal of IFIP, Vol. 22, No. 5, pp.46-52, December 2022.
DOI: https://doi.org/10.1145/3565475.3569084

[4] Muhammad Ibrar, Lei Wang, Gabriel-Miro Muntean, Aamir Akbar, Nadir Shah, Kaleem Razzaq Malik, 'PrePass-Flow: A Machine Learning based technique to minimize ACL policy violation due to links failure in hybrid SDN', Journal of IFIP, Vol. 184, pp.1-15, January 2021.
DOI: https://doi.org/10.1016/j.comnet.2020.107706

[5] Xuan Zhang, Xin Qiu, Junjie Liu, Rui Guo, Shu Shi, Lincheng Li, Jiawei Zeng, 'A Novel Network Asset Security Protection System', International Conference on Artificial Intelligence in Everything (AIE), pp.442-445, 2022.
DOI: 10.1109/AIE57029.2022.00090

[6] Park In-Kyu, Granule-based Association Rule Mining for Big Data Recommendation System', The Journal of The Institute of Internet, Broadcasting and Communication(IIBC), Vol.21 No.3, pp.67-72, 2021.
DOI: 10.7236/JIIBC.2021.21.3.67

[7] Seung-Yeon Hwang, Yoon-Bin An, Dong-Jin Shin, Oh Jae-Kon, Jin- Yong Moon, Jeong-Joon Kim, "A Study on the Document Topic Extraction System Based on Big Data.", The Journal of the Institute of Internet, Broadcasting and Communication(IIBC), Vol.20, No.5, pp.207-214, Oct.2020.
DOI: http://doi.org/10.7236/JIIBC.2020.20.5.207

저 자 소 개

양 재 호(정회원)

- 2013년 2월 : 한양대학교 분자생명과학부(이학사)
- 2017년 2월 : 세종사이버대학교 정보보호전공(공학석사)
- 2017년 3월 ~ 현재 : 한국공학대학교 컴퓨터공학과 박사과정
- 관심분야 : 소프트웨어공학, 정보통신시스템, 장애예측기술

김 영 곤(정회원)

- 1983년 2월 : 경북대학교 전자공학과(공학사)
- 1985년 2월 : 연세대학교 본대학원 전자공학과(공학석사)
- 2000년 2월 : 한국과학기술원 전산학과(공학박사)
- 1985년 ~ 2007년 : KT 수석연구원
- 2007년 ~ 현재 : 한국공학대학교 컴퓨터공학과 교수
- 관심분야 : 소프트웨어공학, 정보통신시스템, 객체지향 분석 및 설계