클라우드 컴퓨팅 환경에서 NoSQL기반 대용량 보안로그 통합

An Integration of Large-scale Security Log based on NoSQL in Cloud Computing Environment

2014년 2월 25일

조선대학교 대학원

컴퓨터공학과

정 희 진

클라우드 컴퓨팅 환경에서 NoSQL기반 대용량 보안로그 통합

지도교수 김 판 구

이 논문을 공학석사학위신청 논문으로 제출함.

2013년 10월

조선대학교 대학원

컴퓨터공학과

정 희 진

정희진의 석사학위논문을 인준함

위원장 조선대학교 교수 李 俊 (인)

위 원 조선대학교 교수 丁 日 鎭 (인)

위 원 조선대학교 교수 金 判 九 (인)

2013년 11월

조선대학교 대학원

목 차

표 목 차

그 림 목 차

ABSTRACT

An Integration of Large-scale Security Log based on NoSQL in Cloud Computing Environment

HuiJin Jeong
Advisor : Prof. Pankoo Kim, Ph.D.
Department of Computer Engineering
Graduate School of Chosun University

As cloud computing technologies are rapidly advancing, cloud environment is significantly expanding. Although cloud environment provides users with convenience, prevention and detection of possible security invasion accidents is still unsolved problems. The most intrinsic method to prevent security invasion accident is to collect security logs for each system and then analyze them.

This paper proposes NoSQL-based large capacity security log integration method for cloud security platform. Since cloud computing provides various services to users, a new security log that is different from existing one is likely to occur. Therefore, this paper proposes large scaled security log management to collect, store and integrate logs considering characteristics between heterogeneous machine.

Amount of large scaled security logs which should be collected and stored under cloud computing environment is rapidly increasing. However, conventional collection method based on RDBMS storage can't afford the amount of security logs. In order to solve this problem, this paper proposes NoSQL-based method to collect and integrate security logs. NoSQL is more effective and rapid data

storage than conventional RDBMS. As a result, this paper confirmed service availability of NoSQL and then integrated more than 87% of all logs in a way that sets key for common area in security log field and integrate them and it also proved that speed is improved when number of processing nodes is increasing.

Ⅰ. 서론

A. 연구 배경

클라우드 컴퓨팅 기술의 발전으로 서비스 제공자는 서비스를 구현하는 데 필요한 기술들에 대하여 익힐 필요가 없어졌으며 다양한 서비스를 클라우드 컴퓨팅 환경에서 손쉽게 구성하고 최종 사용자에게 제공해 줄 수 있게 되었다[2]. 신규 서비스 진입 장벽의 낮아짐으로 인하여 손쉽게 다양한 콘텐츠를 제공할 수 있게 되었고 이에 따라 다양한 로그가 발생하였으며, 최근에는 스마트 폰 보급률의 증가와 네트워크 억세스 기술의 발달에 따른 보안로그는 꾸준히 증가하는 시점에 도달해 있다[19].

네트워크 환경개선으로 인하여 이용자들은 시간과 공간에 제약에 구애받지 않고 손쉽게 클라우드 서비스의 접근할 수 있게 되었다. Amazon, Google, Microsoft, 등 세계적 기업들은 Public Cloud 서비스를 클라우드 컴퓨팅 시스템 도입에 더욱 박차를 가하고 있다[12].

클라우드 컴퓨팅은 사용자 측면에서는 혁신적인 시스템이 틀림없지만 관리자 측면에서는 보안적인 측면에서는 새로운 문제를 야기한다. 자원을 소유하지 않고 일부 또는 전체에 대하여 아웃소싱 하는 구조적인 특징으로 인하여 새로운 보안 위협에 노출된다[23]. 클라우드 컴퓨팅 환경에서 해커에 의한 침투가 일어나게 되면 일반적인 상황보다 큰 문제가 발생할 수 있다[20]. 그렇기 때문에 보안관리자는 일반적인 시스템 운영 시 이기종 시스템에서 쏟아져 나오는 대용량의 보안로그를 한정된 시간 안에 효과적인 분석하여 원하는 내용을 찾아서 잠재적인 보안 위협을 찾아내고 침해사고를 사전에 방지하여 효율적인 서비스 환경을 유지해야 한다[33, 34]. 이러한 대용량 데이터의 가치는 현재 국가 및 기업들이 주시하고 있으며 보유하고 있는 데이터 속에서 새로운 지식의 가치를 창출해 내기 위해 노력하고 있다. 이를 위하여 해결책으로 제시된 방법 중 가장 현재 이슈가 되고 있는 기술은 Hadoop이다. 현재 Hadoop은 빅데이터 처리에 대한 대안으로 평가되고 있다.

본 논문에서는 클라우드 환경에서 발생하는 보안로그를 통합하는 NoSQL 기반

대용량 보안로그 통합 방법을 제안하고 평가합니다. 보안로그는 다양한 서비스를 제공하고 이용자 수가 증가 할수록 기존과는 다른 새로운 보안로그가 추가 발생할 가능성이 높습니다. 그러므로 이종의 보안로그의 특성을 고려하여 로그수집, 저장, 통합하는 대용량 보안로그 관리 방법이 필요하다.

B. 연구 동기 및 목적

DDoS(Distributed Denial of Service, 분산 서비스 거부 공격), XSS(Cross Site Scripting), CSRF(Cross Site Request Forgery) 공격, SQL Injection 등 다양한 방법의 해킹공격 증가로 인해 Application Server, Web Server, IDS(Intrusion Detection System, 침입탐지시스템), IPS(Intrusion Prevention System, 침입방지시스템), Firewall(방화벽) 등의 시스템에서는 대용량의 보안로그들을 지속해서 발생시키고 있다[21, 22].

보안로그(Security Log)란 해커나 사용자가 시스템에 접근 후 사용한 명령어에 대한 기록 그리고 시스템에 명령한 행위에 대하여 시스템에서 처리한 결과와 에러 메시지 등 시스템에서 발생한 운영정보를 모두 기록한 파일이라고 할 수 있다[31]. 관리자의 입장에서는 시스템의 보안 사고를 예방 및 유지하기 위해서는 보안로그 분석 및 모니터링은 매우 중요한 과제라고 볼 수 있다. 보안로그를 분석 및 모니터링 하는 것은 보안 침해 상황 발생 시 해커의 이용한 시스템의 처리 내용이나 이용 상황을 시간의 흐름에 따라 기록된 것으로 침해 상황 발생 시 생성된 보안로그는 해커 및 공격의 근원지를 추적하는 기본 정보가 되기 때문에 매우 중요하다고 볼 수 있다. 기존 시스템에 내재된 보안 위협과 더불어 새로운 형태의 보안 위협에도 노출됨에 따라 이에 대한 대응책이 시급히 요구되고 있다.

본 논문에서는 시스템에서 발생한 대용량 보안로그를 한정된 시간 안에 효과적인 분석을 하기 위하여 NoSQL기반 데이터 저장소와 클라우드 기반으로 한 데이터 처리 플랫폼인 Hadoop을 보안로그 통합에 사용한다. 또한 효과적인 저장을 위해 Hadoop을 기반으로 동작하는 HBase 플랫폼을 프레임워크 구성에 사용하여 클라우드 기반 대용량 보안로그 통합 방법을 제안한다.

C. 논문의 구성

클라우드 컴퓨팅 환경에서 NoSQL 기반 대용량 보안로그 통합 방법론을 제시하기 위해 본 논문은 다음과 같은 구성으로 작성 되었다.

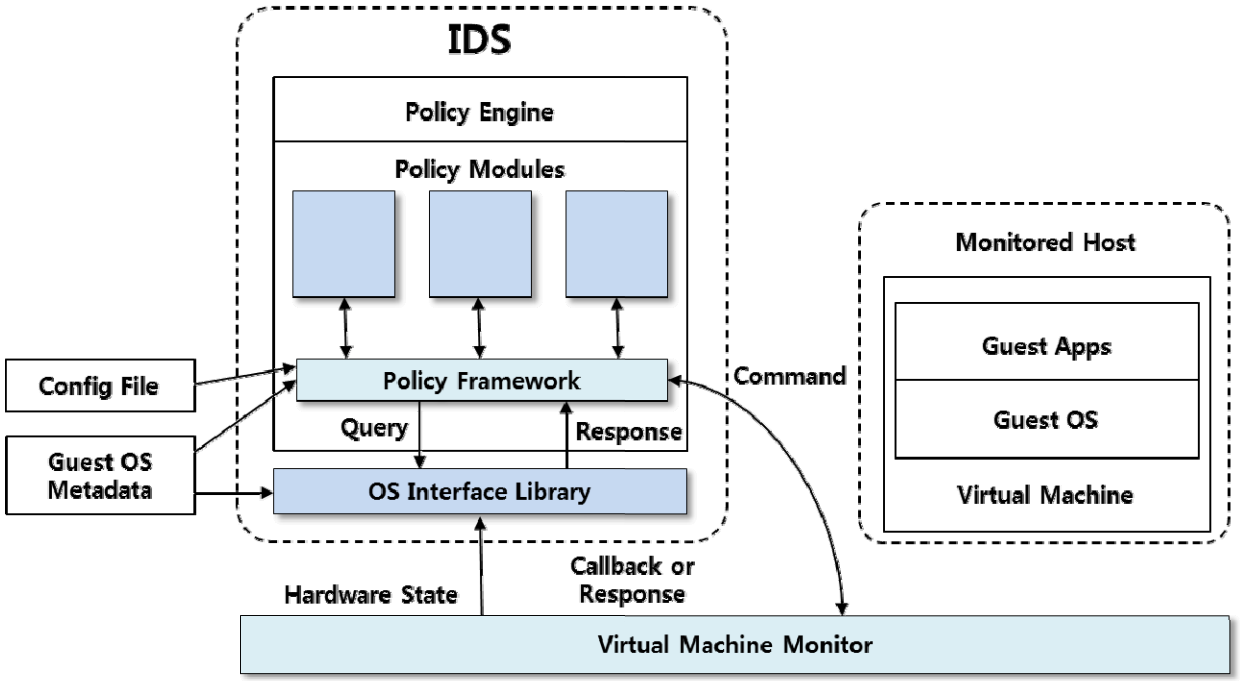
1장은 현재 대용량 보안로그가 발생 되는 IT 기술의 배경과 이에 따라 발생하는 상황을 극복하기 위하여 발표된 기술의 적용분야 및 적용 범위에 대하여 기술하며, 2장에서는 관련 연구, 3장에서는 효과적인 대용량 보안로그 통합을 위해 제안된 프레임워크를 기술하며, 4장에서는 기존에 보안로그 저장소로 사용된 RDBMS와 제안한 NoSQL을 비교 평가하고 MapReduce 기반의 보안로그 통합 방법과 분산처리의 우수성, 그리고 제안된 통합 방법에 대하여 설명한다. 5장에서는 결론을 맺고 제안된 기술 대한 향후 연구 방향을 제시한다.

Ⅱ. 관련 연구

A 클라우드 컴퓨팅 보안

본 절에서는 클라우드 컴퓨팅 보안의 동향과 관련 연구에 대해 살펴본다. 클라우드 컴퓨팅은 언제 어디서나 컴퓨팅 자원을 필요에 따라 차용하여 네트워크를 통해 다양한 방식으로 접근하는 모델이다. 즉, 소프트웨어, 스토리지, 네트워크 등 사용 가능한 대부분의 컴퓨팅 자원들을 필요한 만큼 받아 사용하고 이에 따라 일정 비용을 지급하는 방식으로서 IT 운영비용을 절감하고 개선된 협업 환경과 규모 확장성을 제공하는 장점이 있다[3]. 클라우드 컴퓨팅은 자원 관리의 효율성, 사용자 편의성 등 다양한 이점을 내세우며 새로운 컴퓨팅 환경으로 주목받고 있지만, 보안성 측면에서는 많은 취약성을 가지고 있다. 기존 컴퓨팅 시스템에서 발생 가능한 보안 위협과 더불어 클라우드 컴퓨팅이 갖는 시스템 구조적 특징으로 인해 새로운 형태의 보안 위협에 직면하고 있다. 클라우드 컴퓨팅에서는 기존 컴퓨팅 환경과 달리 가상화 엔진에 의한 보안위협, 관리자에 의한 보안위협, 네트워크 전송과정에서의 보안 위협이 추가된다. 따라서 클라우드 컴퓨팅 서비스를 위해 인증, 접근제어, 암호화 등을 수행하는 인터페이스들은 악의적인 공격 및 사고에 대응할 수 있도록 설계되어야 하며, 또한 피싱이나 소프트웨어 취약성을 이용한 공격을 통해 사용자 계정이나 서비스에 대한 탈취가 발생할 수 있기 때문에 이에 대한 대응책이 필요한 실정이다[4].

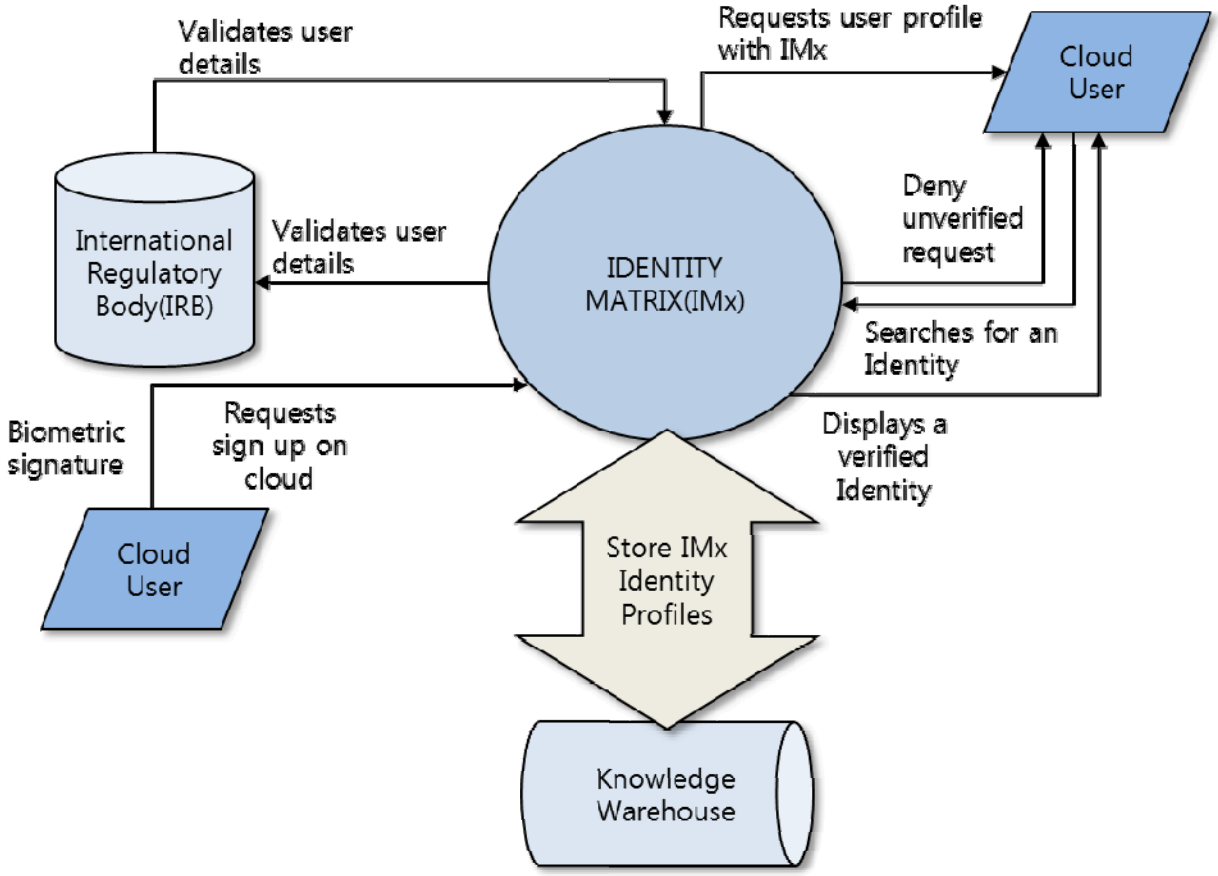
스탠포드 대학에서는 클라우드 시스템의 구조적 특성을 고려하여 가상 머신 상에서 모니터링 하는 호스트를 동작시키고, 가상 머신 모니터를 통해 가상 머신 내부 상태를 분석하는 기법을 통해 호스트 외부에서 호스트에 대한 공격 탐지를 하는 기법을 연구하였다[5].



[그림 2-1] 가상머신 내부 상태 분석(VMI) 기반 IDS 구조[5]

Mulezzeni[6]는 대표적 클라우드 스토리지 서비스인 Dropbox의 인터페이스 보안 취약성을 분석하고, 인터페이스 보안성 향상의 필요성을 제기하였다.

Sarah[7]는 수많은 서비스가 각각 다른 사용 권한을 가지고 있는 클라우드 시스템에서의 사용자 인증과 자원에 대한 접근을 제어하는 기술에 대한 연구로 [그림 2-2]을 보면 클라우드 서비스 사용자와 직접적으로 접하게 되는 Identity Matrix(IMx)를 중심으로 사용자들이 저장한 중요한 정보들이 저장된 knowledge warehouse와 각 정보들의 권한과 인증 등이 저장된 International Regulatory Body(IRB)가 연결되어 있다. 사용자가 정보를 저장하게 되면 검증을 통해 자동적으로 사용자의 정보는 knowledge warehouse에 저장되는데 이 때 IMx에서 접속중인 모든 사용자의 행위 감시를 통해 데이터에 대한 접근을 제어할 수 있는 기술을 연구하였다.



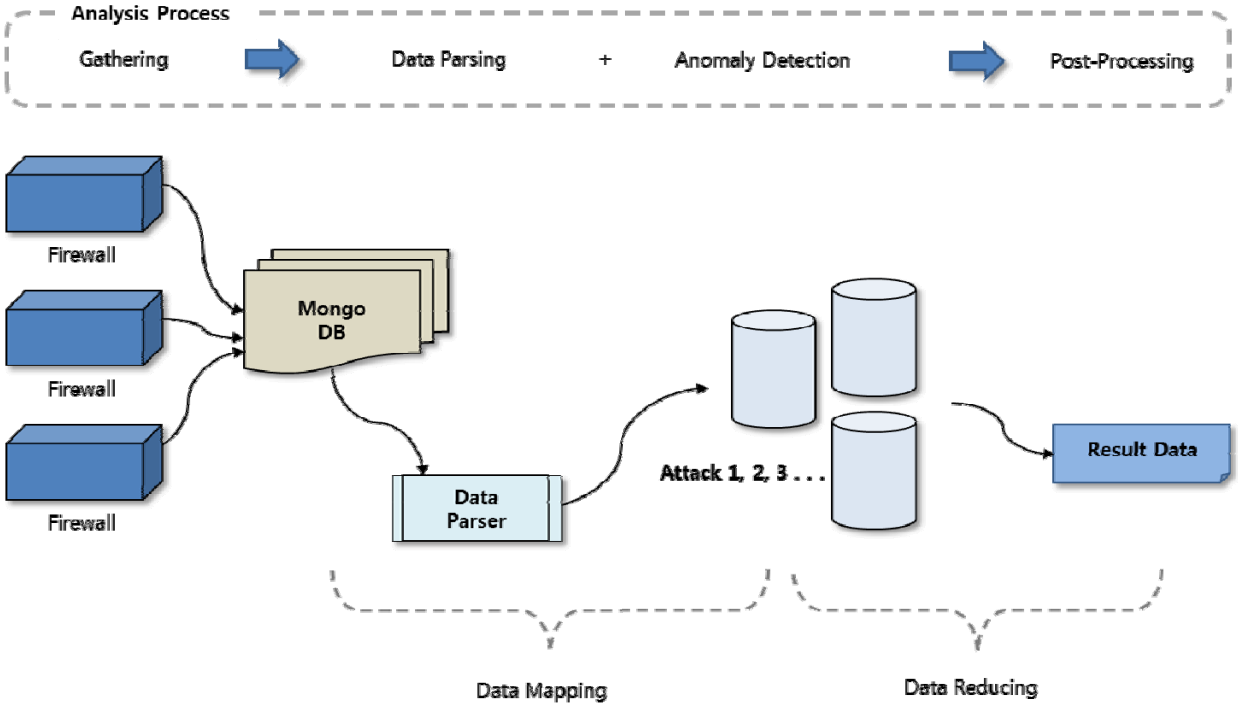
[그림 2-2] IMx를 이용한 사용자 행위 감시 시스템[7]

현재 대부분의 클라우드 서비스가 제공하고 있는 보안기능은 각 서비스에 제한적으로 구현되어 타 클라우드 서비스의 보안기능과의 연동이 어려운 실정이다. 이에 클라우드 컴퓨팅 기술의 특징으로 인해 발생될 수 있는 보안적 문제에 대한 해결책을 제시하기 위한 연구가 활발히 수행되고 있다.

B. 이기종 보안 시스템의 대용량 보안로그 데이터 통합

보안로그 데이터는 시스템 동작 중에 발생한 이벤트들에 대한 순차적인 기록, 시스템이나 네트워크상에 동작되고 있는 내용에 대한 기록을 말한다. 네트워크 및 보안장비, 서버시스템, DBMS, 서비스 등에서 사용자의 행위를 기록하여 보관하며 이를 통해 시스템의 안정적인 운영을 지원하거나, 해킹 등의 불법 침해를 당하였을 때 침입경로 추적과 취약점을 파악하여 현재 발생한 문제와 앞으로 발생할 수 있는 문제를 파악할 수 있다. 따라서 보안로그 데이터 수집 및 분석은 보안관리 시스템에서 매우 중요하다. 하지만 다양한 이기종 보안 장비에서 발생되는 보안로그의 양은 점점 증가하고 있으나 현재 통합 보안 관리시스템은 DBMS를 이용하여 보안로그를 저장하고 분석하는 방식으로 대용량 처리 방법으로는 부적합한 실정이다.

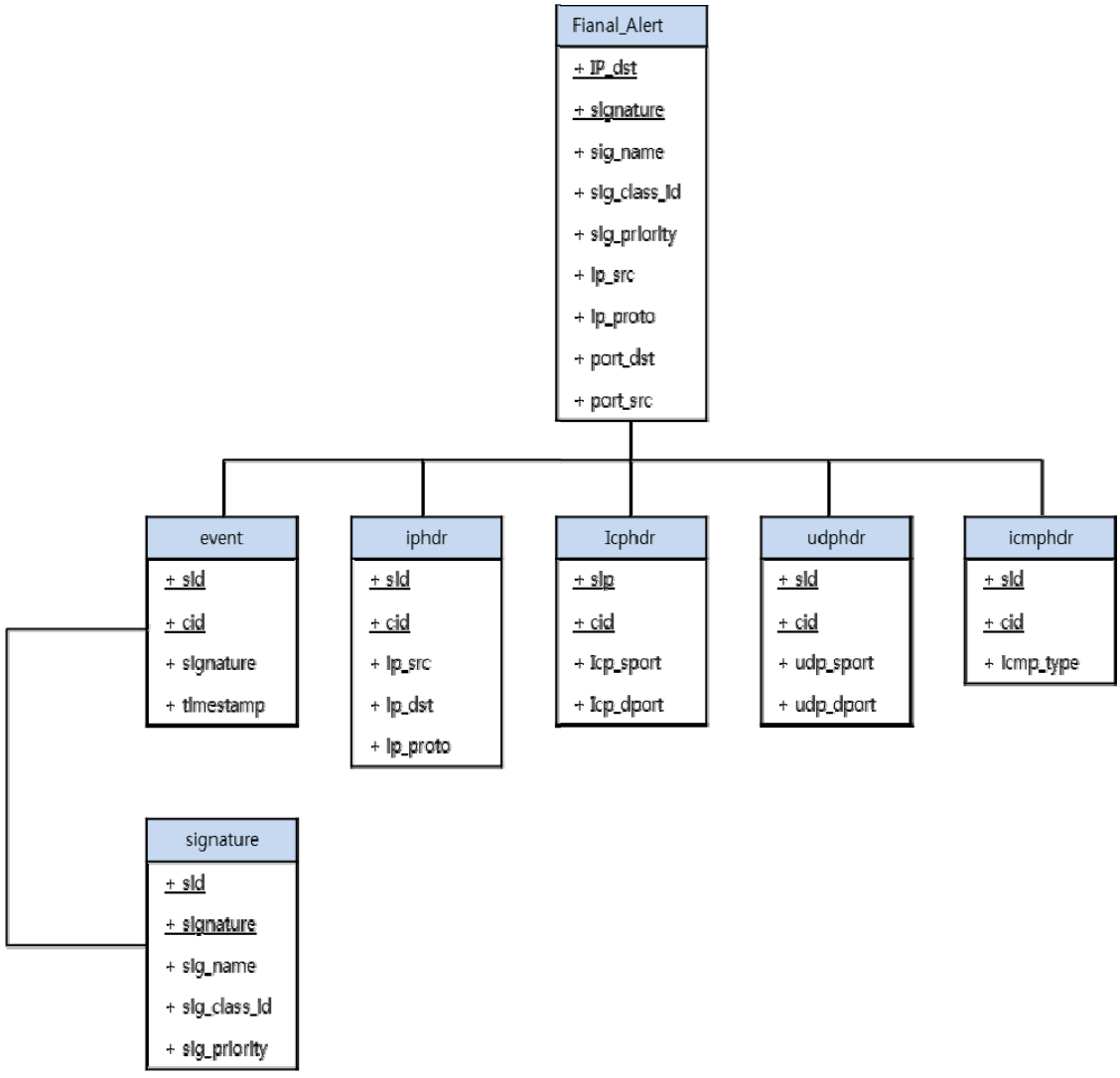
B. Choi et al.[32]는 대용량화되어가고 있는 이기종의 방화벽 로그 데이터를 통합적으로 수집 및 분석할 수 있는 NoSQL 기반의 Mapreduce 설계를 이용한 보안로그 분석 시스템을 제안하였으며, 이를 기존의 RDBMS방식과 데이터 처리 성능을 비교하였고, 평가를 위해 3가지 공격 패턴을 선정하고 분석을 수행하였다.



[그림 2-3] NoSQL기반의 MapReduce를 이용한 로그 분석 기법[32]

[그림2-3]은 NoSQL기반의 MapReduce를 이용한 보안로그 분석 기법 모델의 전체적 프로세스를 나타낸다.

Wei-Yu et al.[8]는 Hadoop과 MapReduce 프로그래밍 모델을 활용하여 이기종 보안 장비에서 발생한 다양한 보안로그를 각각의 특정필드(Signature, Destination_IP)를 기준으로 병합한 모델을 제시하였다. 현재까지 이루어진 대부분의 MapReduce 설계 기반의 로그 분석 기법들의 전신이 되는 모델이다.



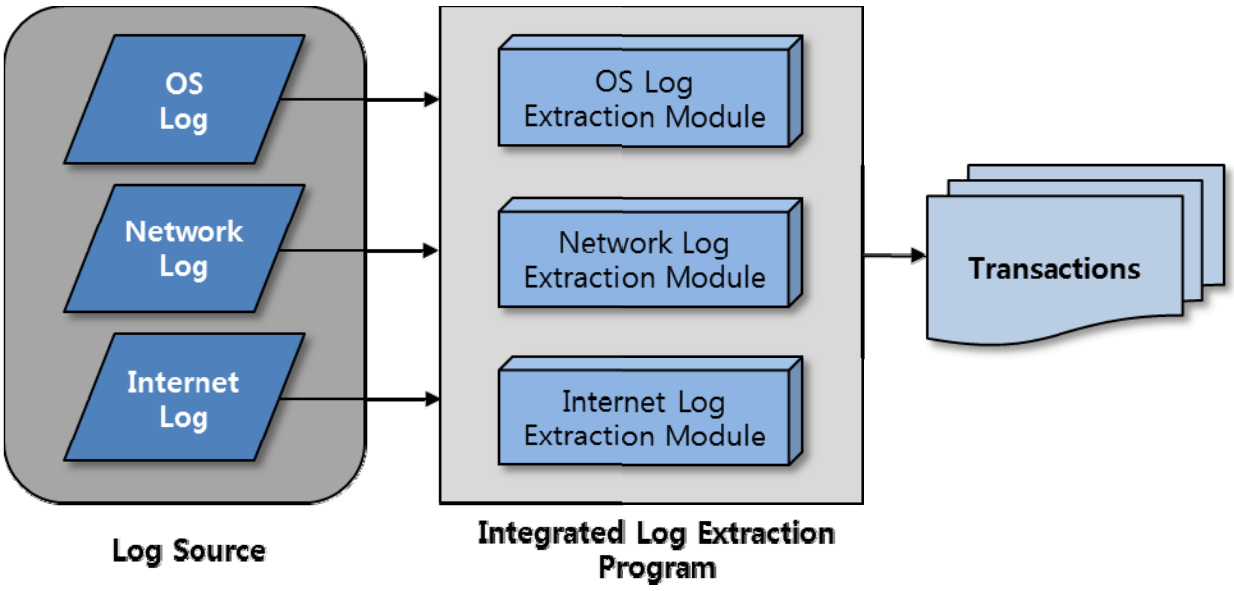
[그림 2-4] 방화벽 보안로그통합 테이블 구조[8]

Y. Lee et al.[9]는 MapReduce 프로그래밍 모델을 활용하여 Flow 데이터의 Destination Port와 Octets 두 가지 요소에 대한 병합을 통해 Traffic Data 측정과 분석에 적용하였다.

W. Kim et al.[33]는 대다수의 통신장비와 보안장비에서의 로그표준인 syslog와 운영체제의 활동기록 및 접속기록에 대한 유닉스/리눅스 시스템 그리고 윈도우즈서버의 로그에 대한 특성을 고찰하였으며, 여러 종류의 시스템에서 발생되는 로그를 통합하여 관리하기 위한 이기종 로그 데이터의 생성에서부터 사용과 소멸에 이르는 생명주기 방법론을 제시하여 안정적인 시스템을 유지하고 개선하는 방법을 연구하였다.

H. Lee et al.[34]는 웹 로그, 웹 IDS 로그, 웹 방화벽 로그 등 서로 다른 형식의 로그들에서 통합 로그에 필드(Time, Source_IP&Port, Destination_IP&Port)에 해당하는 정보를 추출하고 중복된 필드 정보의 연관성을 통해 통합 로그를 생성하고 이를 이용하여 웹 공격을 탐지할 수 있는 시스템을 설계하고 구현하였다. 제안한 시스템은 다중 웹 세션에 대한 분석 과정을 수행하고 웹 시스템 공격과 관련된 연관성을 분석하여 공격 이벤트를 추출하는 기법을 제시하였다.

J. Shin. et al.[35]는 네트워크 패킷 로그, 운영체제 로그 외에도 사용자가 접근하는 인터넷 사이트에 관한 정보인 인터넷 접속기록 로그를 추가로 수집하여 다양한 환경의 로그들을 추출하여 처리에 적합한 형태로 변환과정을 거친 후 비정상행위 탐지를 위한 통합로그 추출 프로그램을 제안하였다.



[그림 2-5] 통합 로그 추출 프로그램 구조 [35]

C. 클라우드 컴퓨팅 데이터 처리기술

1. Hadoop

하둡(Hadoop)은 대용량 데이터의 분산 처리를 위해 거대한 컴퓨터 클러스터에서 동작하는 오픈 소스 프레임워크이다. 원래 오픈소스 웹 검색엔진 너치(Nutch)의 분산처리를 지원하기 위해 개발된 것으로, 아파치 루씬(Lucene)의 하부 프로젝트로 시작되었다. 하둡의 필수 프레임워크는 구글 파일 시스템(GFS)을 벤치마킹하여 하둡 분산 파일시스템(HDFS:Hadoop Distributed File system)과 맵 리듀스(Mapreduce)로 구성된다[24, 10].

하둡 분산 파일 시스템(HDFS)은 대규모 데이터를 저장할 수 있는 분산 파일 시스템으로 대규모 데이터에 대한 고성능 접근을 제공하며, 파일을 기본 64Mbyte로 나누어 분산 저장하여, 안정적이고 빠른 저장소의 역할을 한다[29]. 하둡은 저비용의 범용성 서버에서 동작하도록 고려되었으며 소프트웨어 수준에서 작업 수행 과정에서 발생할 수 있는 장애를 탐지하고 극복할 수 있도록 하였다[28].

그리고 맵리듀스(Mapreduce)는 대용량의 데이터를 빠르고 안전하게 병렬처리하기 위해서 보통의 상용하드웨어를 이용한 분산프로그래밍이라고 할 수 있다[1]. 기존의 IT 아키텍쳐가 애플리케이션이 있는 서버로 데이터를 불러와 처리하는 방식이었다면 하둡의 맵리듀스는 클러스터로 묶인 개별 데이터 노드로 프로그램을 보내 데이터가 존재하는 서버에서 직접 처리하는 사상으로 설계되어있다. 이 때문에 맵리듀스를 통한 병렬처리는 실시간에는 어울리지 않고 일괄 배치(Batch)작업에 최적화되어있다[11].

또한, 빅데이터를 위한 플랫폼은 대용량 데이터의 분산저장과 다수의 서버 클러스터에서 일어나는 병렬처리를 꼽을 수 있는데 하둡은 이에 최적화 되어있다. 이로써 하둡 기반의 다양한 연관 솔루션과 도구들이 등장하며 사용자들은 이를 활용하여 서비스의 목적에 따라 하둡 하부 프레임워크를 선택하여 구성한다.

2. Hbase

HBase는 구글 빅테이블(BigTable)의 오픈소스 클론 프로젝트로써 HDFS에 구현한 컬럼 기반 분산 데이터베이스로써 동일한 형태를 가진 데이터를 컬럼으로 그룹화하여 저장한다[36]. 기존의 관계형 데이터베이스(RDBMS)는 데이터를 로우 단위로 읽기·쓰기를 수행하며, 로우 전체를 사용하는 경우에 더욱 유리하다. 기존의 방식과 다른 관점의 HBase와 같은 컬럼 기반 데이터베이스는 동일한 데이터 형태를 가지고 있는 데이터를 컬럼 단위로 저장하기 때문에 저장 공간 효율성이 좋고, 하나의 컬럼 단위로 I/O를 하는 경우에 기존 RDBMS보다 훨씬 뛰어난 성능을 보인다. 또한, HDFS는 대용량의 시퀀스 파일 포맷에 강점을 보이지만, 삽입된 데이터의 갱신을 하고자 할 때 기존 데이터를 삭제하고 다시 삽입하는 방식을 취하므로 대규모 확장성과 분산을 고려하지 않았다. 하지만 HBase는 확장성 문제에 접근하여 단지 노드만 추가하면 선형적으로 확장될 수 있도록 개발되었다. HBase는 RDBMS가 아니며 SQL도 지원하지 않지만, 범용하드웨어로 이루어진 클러스터는 엄청나게 흩어진 테이블을 수용할 수 있으며, 메모리에서 데이터를 관리하고 메모리가 가득 찼을 시에는 정렬된 파일에 플러시 하는 쓰기 지연 방식을 취함으로써 디스크 I/O를 최대한 줄일 수 있으며 랜덤 읽기와 작은 데이터의 실시간 읽기/쓰기에 적합하다[25, 26].

HDFS가 네임노드와 데이터노드로 구성되어 있고, 맵리듀스가 잡트래커와 태스크트래커로 구분된 것처럼 HBase에서는 HBase 마스터노드가 하나 이상의 리전서브 슬레이브를 조율 합니다. HDFS와 맵리듀스가 대량의 데이터 집합에 대한 배치 작업을 처리하기에 강력한 도구임은 틀림없지만, 개별 레코드를 효과적으로 읽거나 쓰는 방법을 제공하지는 않는 단점이 있다. 이런 기능적인 단점을 보강하여 HBase는 구성 되어 있다[27].

HBase는 마스터서버(HMaster)와 다수의 리전서버(HRegionServer)로 구성되며, SPoF(SinglePointofFailure)를 없애기 위해 마스터-마스터 형태의 복제를 지원한다.

HBase가 분산 환경에서 구축될 시에는 하둡의 하부 프로젝트 중 하나인 Zookeeper가 항상 필요하며, Zookeeper는 HBase의 동기화 등의 매니지먼트를 담당한다[1

8]. 클라이언트가 데이터 삽입 및 검색을 요청하려면 우선 Zookeeper에 해당 데이터를 요청한다. Zookeeper는 마스터서버인 HMaster가 가지고 있는 테이블의 목록을 가지고 있으며, 이 목록에 의해 HMaster에게 해당 데이터를 요청한다. HMaster는 리전서버인 HRegionServer에 저장된 데이타의 시작 로우를 가지고 있으며, 이를 기반으로 HRegionServer가 클라이언트에게 데이터를 제공하게 한다. HRegionServer는 클라이언트의 실제 입출력 요청을 수행하는데 HMaster의 입출력 요청에 대해 최근 입력된 데이터를 가지고 있는 메모리인 Memstore를 검색하고 Memstore에 해당 데이터가 없으면 HDFS에 저장된 데이터를 메모리로 적재하여 실제 입출력을 처리한다.

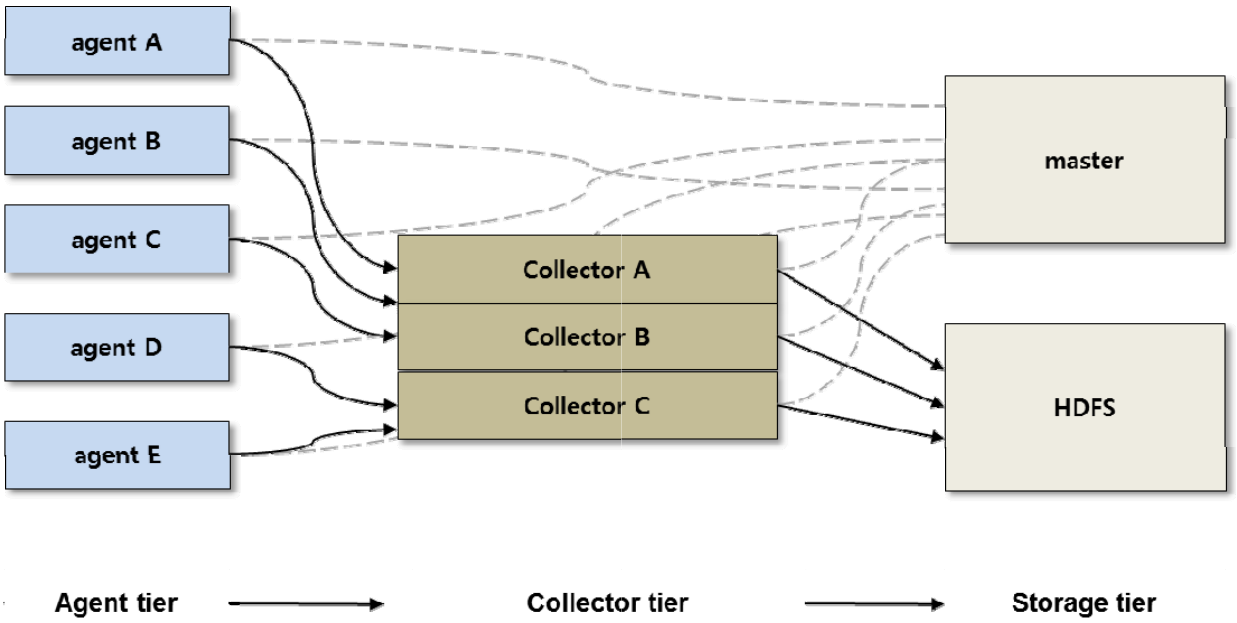
3. 보안로그 수집기

로그 수집 프레임워크(Log Aggregator Framework)는 단위 보안장비 및 네트워크 자원, 네트워크 장비의 보안로그와 트래픽 분석정보 등 흩어져 있는 각종 보안로그데이터를 일관된 방법으로 수집하여 지능적 분석을 위한 정규화 된 데이터를 생산할 수 있도록 설계된 프레임워크로써 보안로그수집 분야에서 반드시 필요한 부분이다.

하둡에 로그 파일을 저장하고자 할 때, 직접 하둡에 파일을 옮기거나 프로그램을 통해 파일 단위로 저장할 수 있지만, 수십 대의 서버에 쌓인 로그 파일을 일일이 옮기기는 쉽지 않다. 이를 아주 효과적으로 처리할 수 있는 수집용 오픈 소스로 Apache Flume이 있다.

Flume은 Cloudera에서 공개한 대량 로그 데이터를 효율적으로 수집, 집계, 이동하는 것을 목적으로 개발된 높은 안정성과 가용성을 갖춘 로그 관리용 오픈 소스이다. 수많은 서버에 분산된 많은 양의 로그 데이터를 flume을 통해서 HDFS와 같은 기본적인 저장소에 수집해 준다[37].

Flume은 에이전트 계층, 컬렉터 계층, 스토리지 계층으로 구성되어 있다.



[그림 2-5] Flume 데이터 저장 흐름[37]

노드는 물리적 노드와 논리적 노드로 구분할 수 있다. 물리적 노드는 머신의 한 JVM위에서 동작하는 하나의 자바 프로세스이다. 물리적 노드도 논리적 노드와 동일하게 동작하지만 물리적 노드위에는 다수의 논리적 노드를 생성할 수 있다. 그래서 필요한 용도에 따라 여러 가지 논리적 노드를 생성해서 데이터 흐름을 구성할 수 있다. 모든 논리적 노드는 이벤트를 생산하는 source와 이벤트를 소비하는 sink 2가지 컴포넌트를 가지고 있으며, source는 어디서 데이터를 수집하는 지를 지정하고 sink는 어디로 데이터를 보내야 하는지를 지정한다.

다시 말해서 Flume은 수평적 확장을 지원한다. 이는 시스템에 추가적인 머신을 추가함으로써 전체 처리율의 향상을 의미하며, 계층별로 부하량에 따라 노드를 추가해서 전체적인 성능을 향상할 수 있다.

4. NoSQL

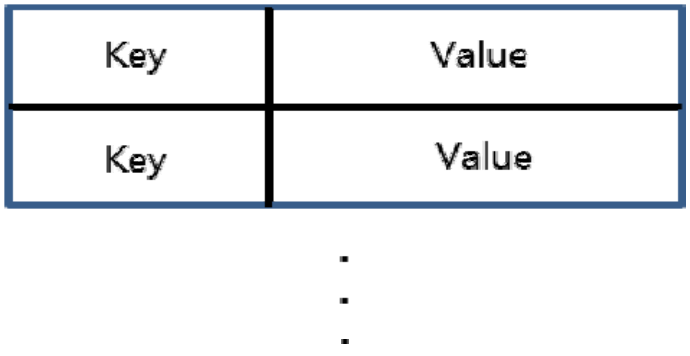
기존 소셜 네트워크 서비스로 대표되는 소셜 미디어의 성장과 최근 스마트 폰으로 대변되는 모바일 장치의 확산이 결합하여 우리 주변에는 규모를 가늠할 수 없을 정도로 많은 정보와 데이터가 생산되는 '빅데이터(Big Data)' 환경이 도래하고 있다. 빅데이터란 과거 아날로그 환경에서 생성되던 데이터에 비하면 그 규모가 방대하고, 생성 주기도 짧고, 형태도 수치 데이터뿐 아니라 문자와 영상 데이터를 포함하는 대규모 데이터를 말한다[39]. 이는 기존의 데이터 저장 시스템으로는 지원할 수 없는 한계를 도달 하였고 결국에는 새로운 형태의 데이터 저장 기술을 요구하게 되었다.

기존의 관계형 데이터베이스(RDBMS)는 데이터의 무결성 및 정합성을 보장을 기준으로 설계된 데이터베이스이다. RDBMS는 정규화 된 데이터베이스 스키마를 사용하고 정보와 정보간의 Join기능을 이용하여 데이터 표현이 자유롭다. 하지만 이러한 정형적인 스키마 구조가 가지는 특성 때문에 RDBMS는 클라우드 분산 환경에는 부적합하며 데이터베이스 확장에 한계가 많다. 이에 데이터베이스의 확장성을 고려하여 개발된 데이터베이스를 NoSQL(Not Only SQL)이라고 한다[38]. NoSQL은 데이터의 무결성 및 정합성을 완벽하게 보장하지는 못하지만, 데이터 추가 및 삭제, 정형적인 데이터 구조에서 자유로우므로 데이터를 저장시 분산 저장에 유연하다고 볼 수 있다. 데이터를 분산 저장하기 때문에 몇몇 데이터를 저장하고 있는 데이터베이스가 응답 하지 않더라도 동작에 이상이 없는 시스템을 제공하는 데 중점을 두고 있다[30].

NoSQL은 데이터 저장 구조에 따라 다음과 같이 구분할 수 있다.

■ Key-Value Model

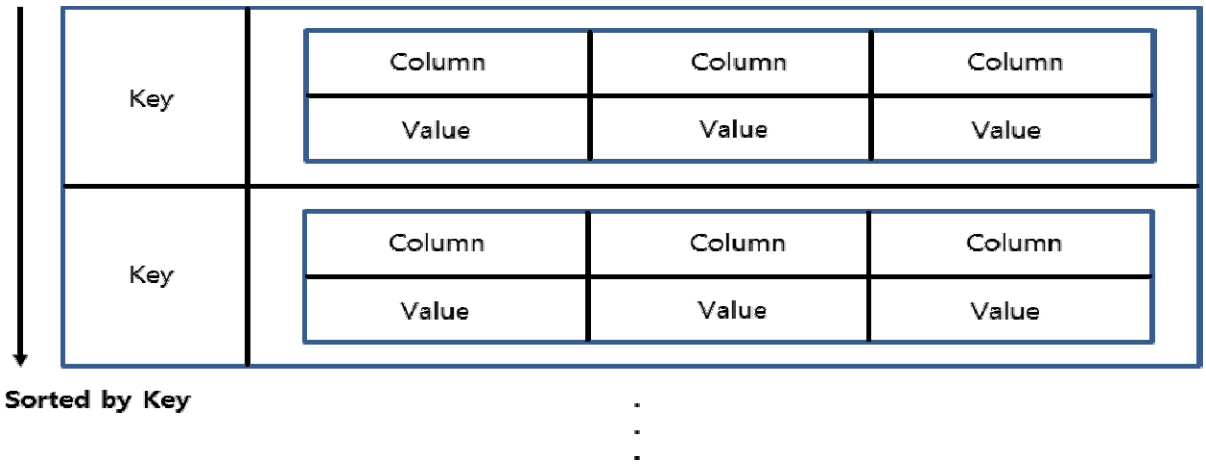
Key-Value Model은 단순하고 빠른 메소드 Get, Put, Delete 기능을 제공하는 모델로 모든 NoSQL 기본적인 모델이다. Key-Value 모델은 단순한 표현이기 때문에 역으로 매우 강력한 데이터 모델링을 지원한다.



[그림 2-7] Key-Value Model 데이터구조

■ Ordered Key-Value Model

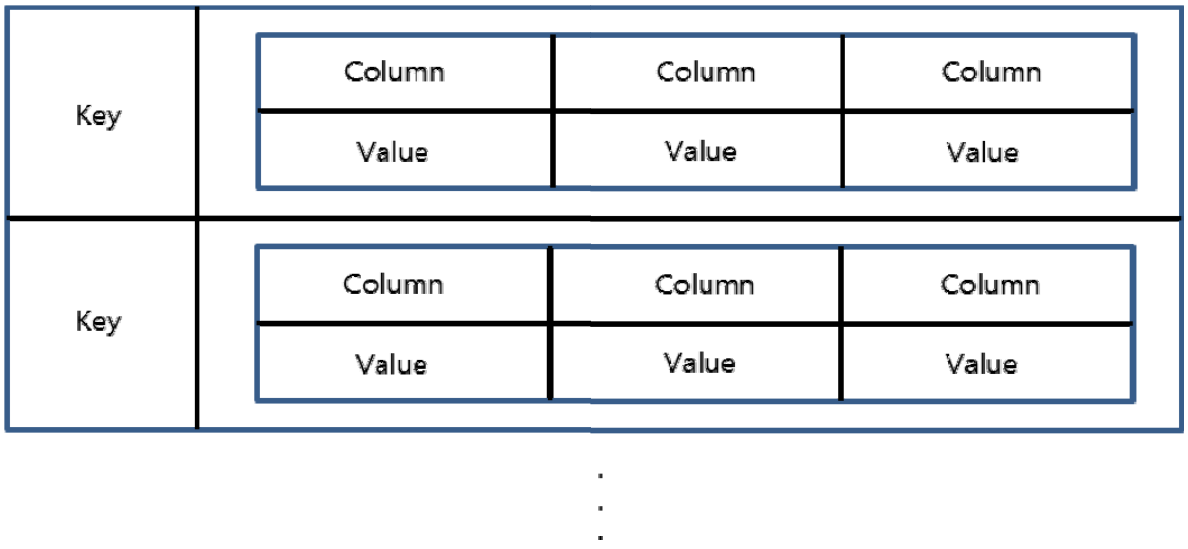
Ordered Key-Value Model은 Key 간의 순서에 따른 범위 검색은 기본적으로 지원하고 Key값의 순차 읽기를 통하여 데이터 접근 시 보다 Key-Value Model 보다 범위 적으로 접근이 가능하다. 다만 Key-Value Model과 마찬가지로 프로그램적으로 Value에 대한 관리가 필요하다.



[그림 2-8] Ordered Key-Value Model 데이터구조

■ Column-Style Model

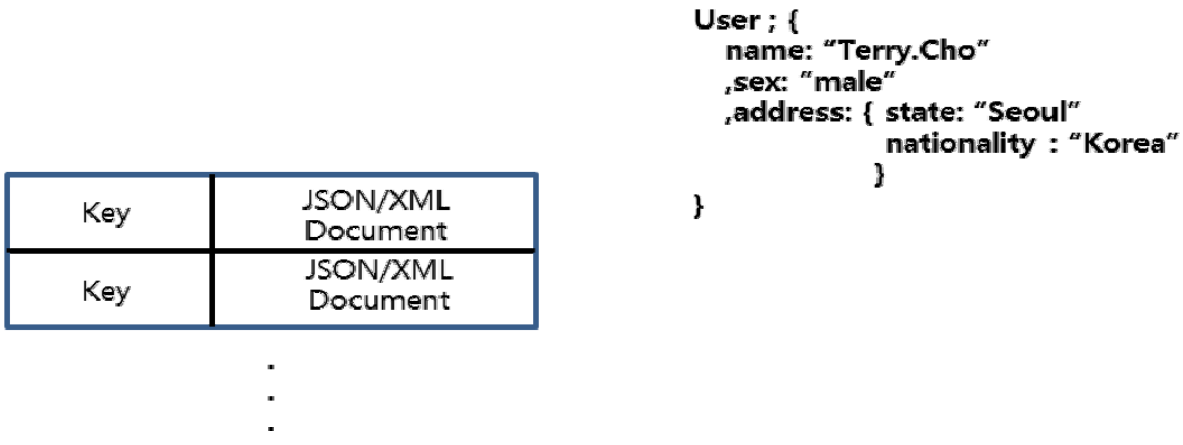
Column-Style Model은 Column을 기본 구성으로 한다. Column을 기반으로 다차원적인 데이터구성을 지원하기 때문에 비정형 데이터 모델링에 유용하다.



[그림 2-9] Column-Style Model 데이터구조

■ Document Database Model

Document Database Model은 SON, XML 형태의 데이터 모델링을 지원하기 때문에 SON, XML의 유연성 있는 기능을 그대로 사용할 수 있다. 기본적으로 필드 이름을 Index로 지원한다.



[그림 2-10] Document Database Model 데이터구조

■ Graph Data Model

Graph Data Model은 RDBMS에서 스키마 구조 간 속성을 노드로 사용하고 관계를 노드 간 엣지 사용하여 스키마간의 확장성 분석을 통하여 데이터를 그래프 형태로 표현한다.

다음 [표 2-1]는 NoSQL 데이터 저장형식에 따른 NoSQL 분류표이다[42].

[표 2-1] NoSQL 종류

구분	종류
Key Value	키 기반 데이터 저장, Redis, Azure Table Storage, 등
Column	컬럼 기반 데이터 저장 Cassandra, HBase, Amazon, SimpleDB
Document	문서 기반 데이터 저장 MongoDB, CouchDB등
Graph	그래프 형식으로 데이터를 저장 Neo4j, GraphBase, VertexDB

Ⅲ. NoSQL 기반 대용량 보안 로그 통합

클라우드 컴퓨팅의 발전으로 시스템 장비의 집중화, 거대화, 고속화가 진행되고 있다. 그로 인해 사용자에게 제공되는 서비스 질 및 처리속도는 크게 향상 된 만큼 발생하는 이종의 보안로그 데이터의 양 또한 기하급수적으로 증가하고 있다. 이에 기존에 사용되던 보안로그 데이터 저장소인 관계형 데이터베이스 기반 보안로그 분석 시스템들은 보안로그 저장 속도 및 데이터 저장 공간이 부족하여 정상적인 시스템 운영에 문제를 일으킬 수 있다. 이러한 기존 방식의 문제점은 클라우드 컴퓨팅 환경에서 발생되는 대용량 보안로그 데이터에 대한 효율적인 보안로그 분석을 처리하는데 시간적 부담감과 처리 속도에 문제점을 발생시키고 있다[12]. 그리고 점차 지능화 되어가고 있는 공격도구로 인하여 보안로그를 분석하여 침해탐지 상황을 밝혀내는 것도 많은 어려움이 있다. 보안로그 분석은 침해 탐지에서 가장 기본적이면서도 매우 중요한 작업이다. 그러나 기존 관계형 데이터베이스의 특성으로 인하여 새로운 방식의 공격 유형 및 이종의 보안로그를 추가 저장시에는 테이블 및 이들 간의 관계를 재 생성해야 하기 때문에 새로운 공격 유형을 대응하는 절차가 복잡해 질 수밖에 없다.

본 논문에서는 이러한 관계형 데이터베이스에 가지고 있는 약점을 보완하여, 클라우드 컴퓨팅 환경을 이용하여 대용량화 되어가고 있는 이종의 보안로그 데이터의 분석방법에 대하여 제시한다. NoSQL 기반에 분산 및 병렬 처리에 최적화 되어 있는 새로운 저장소를 적용하여 기존에 관계형 데이터베이스 방식의 단점인 속도 및 성능 저하 현상을 방지하는데 효과적인 것은 확인 하였다. 또한, 다수의 보안 로그데이터에 대하여 MapReduce 처리방법을 적용하여 중복된 보안로그를 통합시키고 이를 통해 분석 프로세스를 효과적으로 줄일 수 있는 방안에 대하여 제시한다.

A. 대용량 보안로그 데이터 통합을 위한 프레임워크 설계

본 장에서는 대용량 보안로그 통합을 위한 전체 프레임워크 설계에 관해 기술한다. 대용량 보안로그 통합은 대규모의 보안로그의 수집 시점과 생성된 대용량 보안로그에 대한 수집 및 전처리 과정, 통합 과정, 그리고 보안 시스템에 다시 적용될 방안 등을 고려하여 설계해야 한다.

1. 프레임워크 요구사항

보안로그의 통합은 대용량 보안로그가 발생하는 시점에 추가 및 수정, 삭제되는 일들이 빈번하더라도 이를 사용하는 서비스에 문제가 없어야 하고, 보안 침해 상황 발생 시 이를 탐지하는 과정에서 대용량 보안로그의 활용하는 시스템의 성능 저하가 없어야 한다. 또한, 기존 보안로그의 저장 및 관리가 클라우드 컴퓨팅 규모에 영향을 주지 말아야 하며, 새로운 보안 시스템의 적용에 서 데이터 확장이 용이해야 한다.

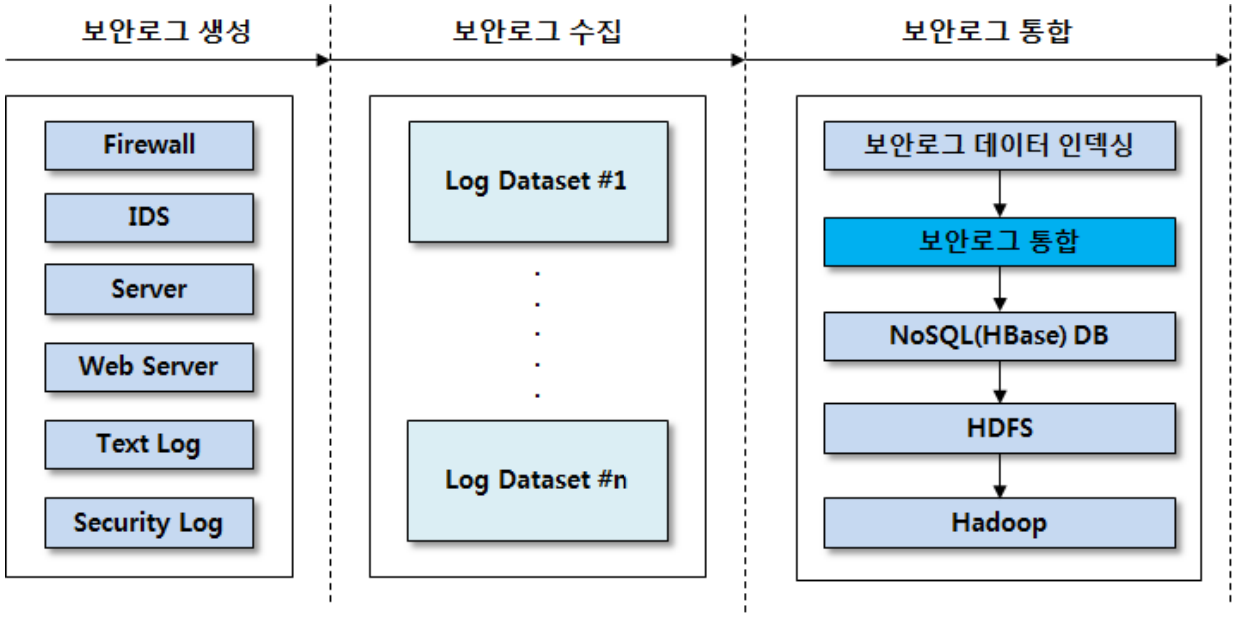
신뢰성 측면에서는 보안 시스템 적용 시 대용량 보안로그를 처리하는 클라우드 컴퓨팅 환경의 다수 시스템에 대한 네트워크 대역폭 부족, 메모리 부족 등에 의한 서비스 실패에 따른 보안로그의 손실을 방지할 수 있는 내결함성을 갖춰야 한다.

2. 보안로그 통합을 위한 프레임워크 구성

대용량 보안로그 통합의 첫 번째 과정은 클라우드 컴퓨팅 환경의 다수의 이종의 시스템에 저장된 보안로그의 수집이다. 보안로그의 구조의 분석을 통하여 로그의 패턴을 찾아내고 데이터를 수집 후 원하는 형태로 데이터를 가공 후 저장해야 한다. 이러한 작업은 보안로그 데이터의 통합을 위해 필수적인 과정이다. 데이터 통합 시에도 수많은 데이터 중에서 어떠한 데이터를 추출할지 선택하는 과정은 보안로그 데이터 통합의 기본적인 과정이라고 볼 수 있다.

본 논문에서는 클라우드 컴퓨팅 환경에서 이종의 보안 시스템에서 생성되는 보안로그를 보안로그 수집기를 통해 수집하고, 수집된 보안로그를 전처리 과정과 인덱싱 과정을 통해 NoSQL 기반의 HBase 데이터베이스로 저장된다. 이는 이종의 시스템에서 발생하는 보안로그를 효과적으로 수집하고 저장하여 로그를 통합 관리

하기 위한 과정이다. 다음 [그림 3-1]은 본 논문에서 제시한 전체 프레임워크를 도식화한 것이다.



[그림 3-1] 대용량 보안로그 통합을 위한 전체 프레임워크

보안 로그 분석시스템의 구성의 전제 조건은 다음과 같다.

(1) 시스템 운영 및 공격 상황이 발생한다고 가정함.

(2) 시스템에서 보안 로그 생성됨.

(3) 생성된 보안로그는 로그 수집기로 전송됨.

(4) 전송이 완료된 로그는 전처리 과정을 진행함.

(5) 전처리가 완료 된 로그데이터는 NoSQL 데이터베이스에 저장됨.

(6) 저장된 보안로그들은 분석하여 패턴 및 정보 추출함.

(7) 추출 된 데이터는 관리자에게 전달된다.

3. 통합 보안로그 저장을 위한 NoSQL

클라우드 컴퓨팅 환경에서 NoSQL 솔루션은 관계형 데이터베이스 모델링보다 효율적인 데이터 모델을 생성하기 위해서 데이터 접근 알고리즘 및 스키마 구조에 대한 이해가 필요하다. NoSQL은 크게 4가지 기능으로 분류 되어 저장할 데이터의 특성을 고려하여 선정해야 한다. NoSQL 솔루션 별 특징을 비교 분석한 표이다.

NoSQL 데이터베이스는 앞서 언급한 바와 같이 크게 4가지 기능으로 분류되어 있고 저장할 데이터의 특성을 고려하여 선정해야 한다.

보안로그 분석 시스템을 구성하기 위해서는 Column Style Model구조를 가지고 있는 Hbase 솔루션이 적합하다고 생각하였다. Column 기반 데이터 저장구조는 Column을 기준으로 새로운 데이터 테이블 구조를 만들 수 있고 이러한 구조는 데이터 추적 및 시간에 흐름에 따라 발생 되는 보안로그 데이터 저장 및 분석에 유용할 것으로 보인다.

[표 3-1] NoSQL 솔루션 기능 비교[40]

구분	MongoDB	CouchDB	HBase	Cassandra
개발 언어	C++	erlang	Java	Java
구분	Document	Document	Wide Column	Wide Column
프로토콜	BSON	HTTP/REST	HTTP/REST Thrift	TCP/Thrift
Map-Reduce	Yes	No	Yes	Yes
Index	Yes	Yes	Yes	Yes
검색지원	Yes	No	Yes	Yes
노드구성	Master-Slave	Multi-Master	Multi-Slave	Multi-Master
분산 저장	Yes	YES	YES	YES

또한 Hbase는 대용량 데이터 처리의 모범답안이라고 불리는 클라우드 컴퓨팅 기술인 Hadoop을 기반으로 운영되기 때문에 OpenSource 라는 단점을 보완하고 차후 탄탄한 솔루션으로 발전해 갈 가능성이 매우 높다. 의례적으로 미국의 클라우데라(Cloudera)사의 임팔라(Impala), 호튼웍스(Hortonworks)사의 스팅거(Stinger), 맵알(MapR)의 드릴(Drill), 그루터 타조(Tajo) 등 업체에서도 Hadoop 기반 대용량 데이터 분석 솔루션을 개발에 박차를 가하고 있는 만큼 가까운 시일 내에 Hadoop 기반으로 동작하는 각종 플랫폼이 대용량 데이터 처리에 많은 영향을 끼칠 것으로 여겨진다. 본 논문은 Hadoop기반으로 대용량 데이터 저장 기능을 지원하는 HBase 보안로그 저장 플랫폼으로 채택하였다[41].

B. 대용량 보안로그 수집 및 전처리

본 장에서는 클라우드 컴퓨팅 환경에서 이종의 시스템에서 발생하는 보안로그 통합을 위한 보안로그 데이터의 범위와 통합을 위한 구조 설계에 대해 기술한다. 클라우드 컴퓨팅 환경은 다양한 형태의 이종의 시스템과 보안 시스템이 존재하므로 발생하는 보안로그의 구조적 형태와 클라우드 컴퓨팅에서의 데이터 처리 알고리즘을 고려하여 설계할 필요가 있다. 특히, 대용량 데이터를 처리하고 운영하는데 필요한 분산 처리 플랫폼인 HBase 시스템의 특성과 MapReduce 프로그래밍 모델을 참조하여 설계한다.

1. 보안로그 수집 범위

a. 시스템기반 보안로그

시스템기반 보안로그는 리눅스, 유닉스, Window 등 에서 이종의 시스템에서 생성되는 메시지이다. 이를 이용하여 시스템이나 응용 프로그램에서 발생하는 각종 메시지를 관리자가 확인하여 사용 할 수 있다. 그중에서도 SYSLOG는 운영체제에 관계없이 동일하게 사용할 수 있다는 장점도 갖고 있으며 기록되는 내용은 시스템에서 동작하는 프로세스의 정상 운영 메시지부터 시스템 오류 메시지까지 다양한 이벤트를 포함하고 있다. 침해 사고 발생 시 로그를 분석함으로써 행위추적에 중요한 근거 자료로 활용이 가능하다[13].

[표 3-2] 리눅스 SYSLOG 로그 설명

로그 종류	설명
syslog	unix 시스템에서 로그 메시지를 처리하기 위한 표준화된 인터페이스이며, 운영체제 종류에 관계없이 동일 사용
messages	시스템의 부팅시와 부팅 이후의 실행된 데몬들의 메시지에 대한 로그
lastlog	사용자의 마지막 로그인 정보 저장
xferlog	ftp 접근에 대한 사용 로그
sercure	시스템 접속 보안 인증(ssh)에 관한 로그 파일

[표 3-2]는 리눅스 및 유닉스 시스템에서 공통적으로 사용되는 보안 로그양식에 대한 설명 표이다. 아래의 표를 살펴보면 다양한 데몬에서 사용자의 행위에서 발생된 정보를 각 파일 별로 구별하여 저장하는 것을 알 수 있다. 본 논문에서는 시스템에서 일어나는 행위에 대한 확인 및 추적하기 위해서는 모든 이종의 시스템에서 사용이 가능한 syslog 보안로그 파일의 구조를 분석하여 로그 통합 시 고려하였다.

b. 웹 서비스 기반 보안로그

웹 로그(web log)란 웹 사이트 방문자들이 제품이나 서비스를 구매하는 과정을 통해 발생하는 데이터이다. 정보획득이나 구매를 목적으로 인터넷 사이트를 방문하는 방문자들은 로그의 형태로 사이트 내에 흔적을 남기는데 이러한 데이터를 기반으로 해서 다양한 정보를 추출해 내는 것이 웹 로그 분석이다[14].

소프트웨어인 아파치 웹서버와 윈도우즈서버에서 기본으로 사용되는 IIS에서 사용하는 기본적인 로그 방식은 방문자가 페이지를 조회하거나 특정 행위에 대한 정보를 수집하여 기록하는 방식을 제공한다[16].

[표 3-3] Web Server Log 로그 설명

Access_log	접속 요청 및 시도에 대한 로그
Error_log	접속 요청 시 에러에 대한 로그 기록
Referer_log	접속 페이지의 방문 전 위치 정보 로그 기록
Agent_log	웹브라우저 타입 및 버전에 대한 로그 기록

먼저[표 3-3] 살펴보면 아파치 웹서버는 크게 가지 오류로그(Error log), 접속로그(Access Log), 리퍼러 로그(Referer_log), 에이전트 로그(Agent_log)등 으로 구성되어 있으며 관리자에 의하여 위치 및 기록 방법을 지정할 수 있다.

본 논문에서는 웹서버에서 일어나는 행위에 대해 모든 기록을 하고 있는 웹기반 보안로그 파일의 구조를 분석하여 로그 통합 시 고려하였다.

c. IDS 기반 보안로그

IDS(Intrusion Detection System)는 침입탐지시스템이라 불리며 탐지위치에 사용목적 및 탐지 위치에 따라서 다양한 사용자의 행위를 감시 및 탐지하도록 설계되

었고 IDS의 수행 과정 특징은 데이터 위치, 침임 탐지 방법, 침입 대응 방법에 따라 분류 된다[15].

[표 3-4] IDS 동작 위치에 따라 구분

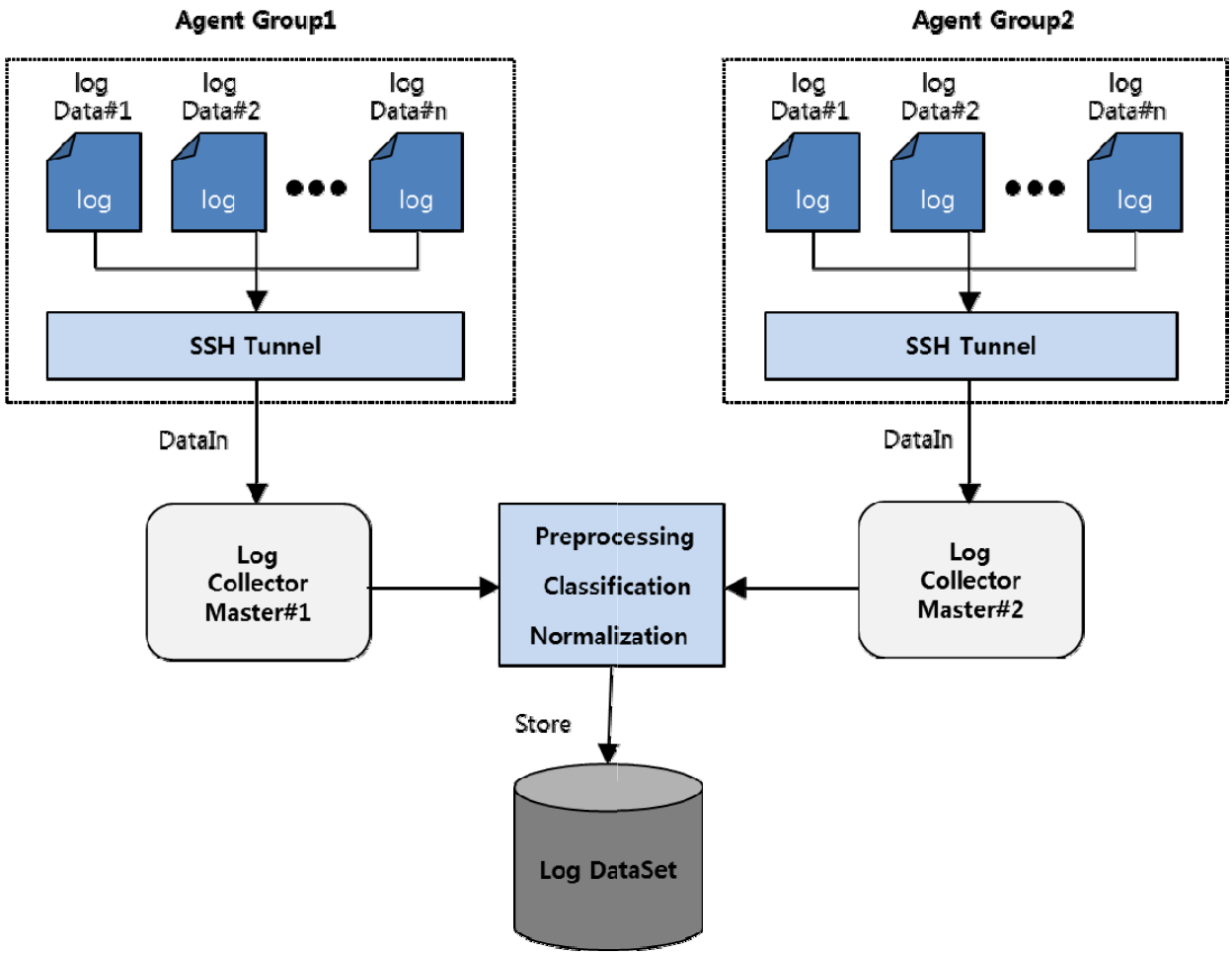
종류	내용
Network-based IDS	Network 패킷을 수집하여 내/외부 탐지
Single Host-based IDS	Host의 로그파일로부터 정보를 수집하여 탐지
Multiple Host-based IDS	다중의 Host로부터 정보를 수집하여 탐지
Hybrid type IDS	Host와 Network 조합하여 탐지

[표 3-4]는 다양한 환경에서 동작하는 IDS 시스템의 구동 위치를 나타낸다. IDS 시스템은 구동위치에 따라서 다양한 로그가 발생하지만 기본적인 보안로그의 필드 구조는 필드의 위치가 다르지만 실제로는 같은 내용을 기록하고 있다[17].

본 논문에서는 Host-Based 기반 IDS로그 파일을 분석하여 보안로그 통합 구조를 설계 하였다.

2. 대용량 보안로그 수집

대용량 보안로그 분석 시스템을 구성할 시 가장 중요한 것은 보안로그 수집 및 수집 대상 선정 부분이다. 보안로그 데이터의 구조의 분석을 통하여 보안로그의 패턴을 찾아내고 원하는 형태로 데이터를 가공 후 저장해야 한다. 이러한 작업은 보안로그 데이터의 통합을 위해 빠트릴 수 없는 필수적인 과정이다. 데이터 통합 시에도 수많은 데이터 중에서 어떤 데이터를 추출 할 것 인지 데이터 범위 선정은 보안로그 통합의 기본적인 과정이라고 볼 수 있다. 본 절에서는 클라이언트 자체적으로 프로그램을 내장하여 로그를 수집하고 분석하는 구조가 아닌 원격 수집서버를 이용하여 클라이언트에서 생성된 보안로그를 수집하여 저장하는 모듈을 제안한다. 이를 이용할 경우에는 클라이언트의 성능 저하 문제를 발생시키지 않을 수 있으며 별도의 보안로그 수집 및 통합 절차를 적용 할수 있는 장점이 있다. 다음 [그림 3-2]는 이종의 시스템에서 발생할 수 있는 보안로그 데이터를 수집하는 프레임워크를 표현한 것이다.



[그림 3-2] 보안로그 수집 프레임워크

보안로그 수집 프레임워크의 단계별 프로세스는 다음과 같다.

(1) 시스템 별로 설정된 위치에 보안로그 새로운 보안로그 생성 체크함.

(2) 시스템에서 새로운 보안로그가 생성됨.

(3) Log Collector Master 서버와 SSH Tunnel 생성 유무 확인함.

(4) SSH Tunnel 생성 되지 않았다면 SSH Tunnel 생성함.

(5) SSH Tunnel을 이용하여 보안로그 데이터 전송함.

(6) 전송된 보안로그 데이터는 전처리 과정을 진행 완료 후 저장함.

(7) 1 ~ 6 까지의 과정을 반복함.

보안로그 수집 과정에서 원격지에서 발생하는 보안로그 데이터도 SSH 통신을 이용하기 전송하기 때문에 보안상의 문제를 줄일수 있다.

3. 대용량 보안로그 전처리

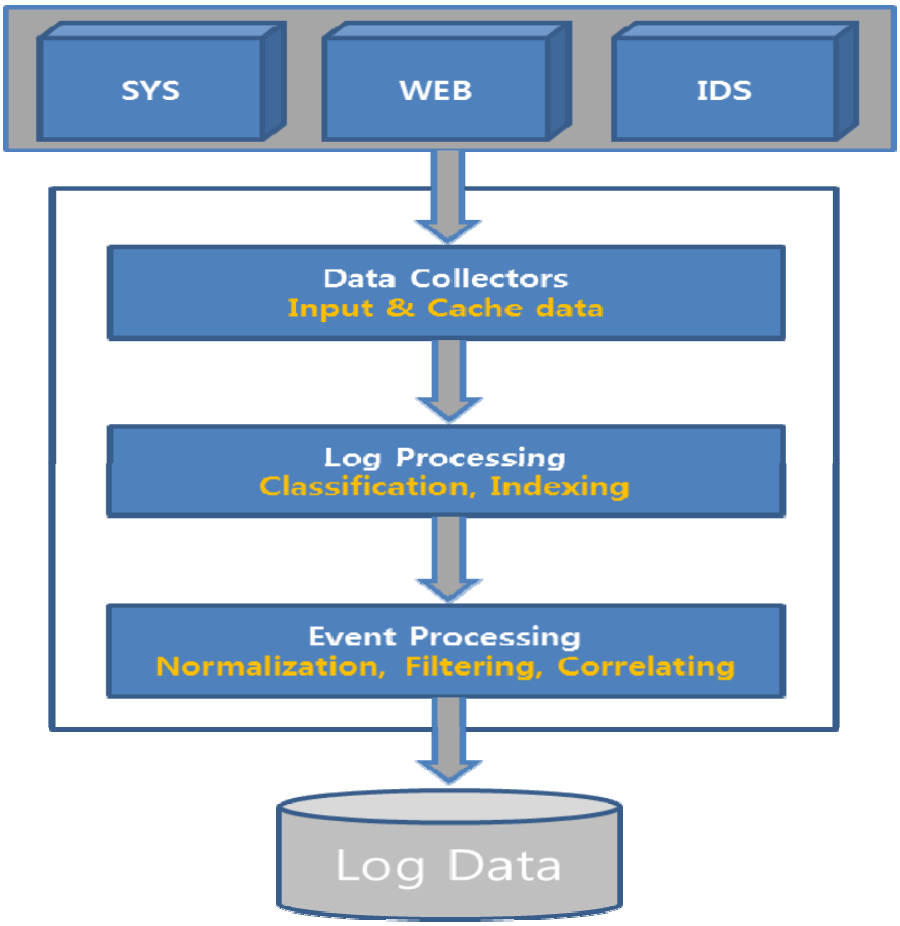
선별된 보안로그는 데이터 저장 전에 분석 가능한 형태로 데이터를 변환해야 하며 데이터 수집에서 가장 중요한 부분은 수집된 데이터의 질이다. 전처리 과정은 방대한 보안로그 데이터에서 불필요한 데이터를 삭제하며 유용한 데이터를 추출하기 위한 과정이라고 볼 수 있다. 본 논문에서는 수집 대상 보안로그 데이터의 질을 높이기 위해서 보안로그 데이터별로 전처리 과정을 진행하였다. 전처리 과정에서는 불필요한 문자열을 제거 및 데이터 공통 영역을 체크하게 된다. 또한, 보안로그 데이터 필드 구조 및 필드내용을 검증하여 저장 전에 생성된 보안로그의 정상 유무를 판단한다.

[표 3-5] 고려한 대상 보안로그 필드 구조

1. SysLog
DateTime:Hostname:Daemon Name[Daemon Process ID]: Log Info
2. WebLog
ClientIP : DateTime : Rquest Info : Return Code : Byte Size
3. IDS Log
DateTime : Hostname : SIP : DIP : Log Info

[표 3-5] 본 논문에서 수집대상으로 지정한 보안로그의 필드를 나열한 것이다. 생성되는 보안로그에 대한 자세한 설명은 앞 3.A절에서 확인 할수 있다.

보안로그 데이터 수집과정에서 가장 먼저 해야 할 작업은 분석 할 대상의 구조를 확인하고 명확히 구분되도록 한다. 또한, 각각의 보안로그는 항목별로 필요한 데이터 필드만 수집되어야 하며 필요에 따라 관리자가 정의한 형식에 맞추어 분류 및 정규화 작업을 진행할 수 있다. 다음 [그림 3-3]은 보안로그 데이터 전처리 과정에 대한 프로세스는 보안로그 데이터의 질을 향상 시키는데 중점을 두고 설계하였다.



[그림 3-3] 데이터 전처리 과정 구조

보안로그 데이터 전처리 과정에 대한 구조는 [그림 3-3] 나타내며 데이터 전처리 과정에 대한 프로세스 진행 구조는 다음과 같다.

(1) SYSTEM, WEB SERVER, IDS에서 보안로그 데이터가 발생함.

(2) 입력된 보안로그 데이터를 검증함.

(3) 보안로그 데이터를 내용을 분석하여 분류 작업을 진행함.

(4) 분류된 보안로그는 특성에 따라 관리자가 정의한 형식으로 정제함.

(5) 정제된 데이터를 저장소에 저장함.

(6) 1 ~ 6번의 과정을 반복한다.

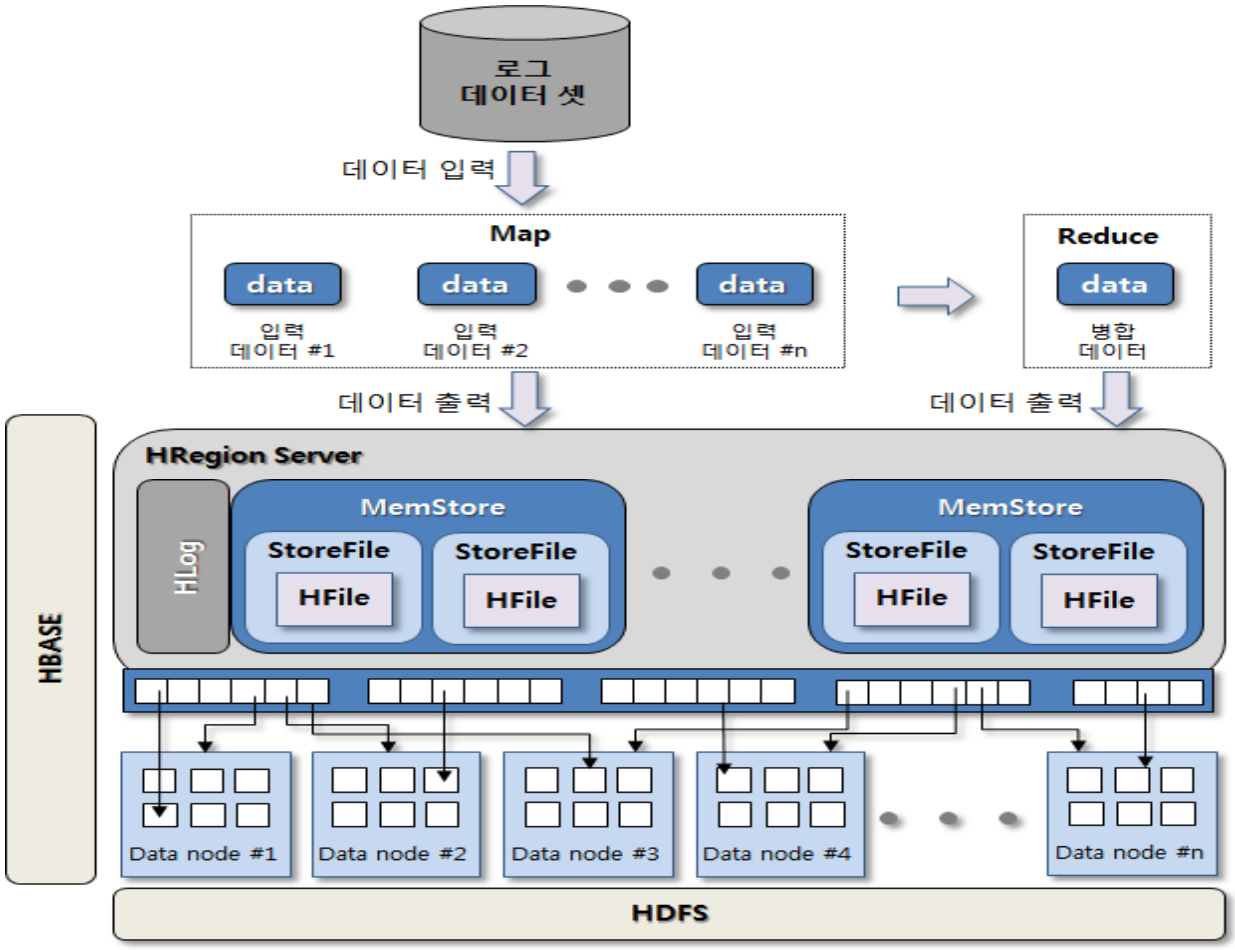
데이터 전 처리 과정을 진행함으로써 보안로그 데이터에서 데이터 노이즈를 줄임으로, 불필요한 데이터 저장 공간의 낭비 및 보안로그 통합 작업 시 오류를 줄일 수 있다.

C. NoSQL을 이용한 대용량 보안로그 통합

본 절에서는 제안하는 NoSQL 기반 대용량 보안로그 통합을 위한 데이터 처리 방법에 대하여 기술한다.

1. NoSQL을 이용한 대용량 보안로그 통합 구성도

다음 [그림3-10]은 본 논문에서 제안하는 NoSQL 기반 대용량 보안로그 통합 구성도이다. 보안로그 데이터는 시간의 흐름이나 사용자 접근에 따라서 새롭게 생성이 되면 그 후에는 수정이나 삭제가 많이 이루어지지 않으며 대부분 추후 분석을 위해 데이터 읽기가 발생한다. 이를 고려하여 NoSQL 기반 보안로그 데이터 저장소를 구성하였다.



[그림 3-4] 대용량 보안로그 통합 구조

보안로그 데이터 통합 과정에 대한 프로세스 진행 구조는 다음과 같다.

(1) LogData Set에 전처리 완료된 데이터가 저장됨.

(2) LogData를 읽어들임.

(3) 데이터 특징에 따라서 Key를 설정하고 Key의 횟수를 측정함.

(4) 측정된 Key 별로 횟수를 통합함.

(5) 통합된 LogData를 저장함.

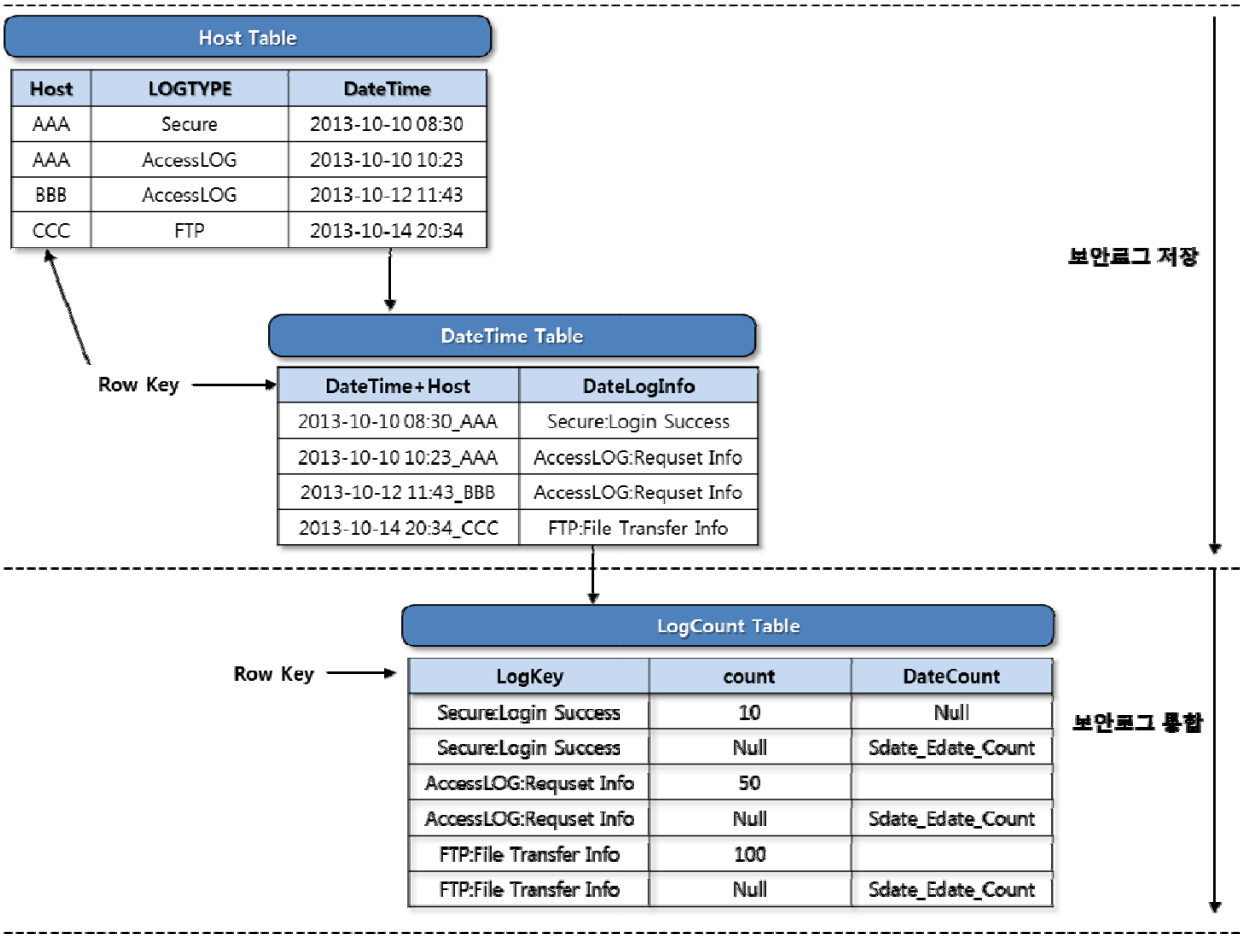
(6) 1 ~ 6번의 과정을 반복함.

보안로그 데이터는 시간의 흐름에 따라 발생함으로 보안로그 통합 기준을 시간으로 고려하여 통합하는 것이 사용자와 보안로그의 행위 추적에 유리하다. 그리고 다양한 서비스를 제공하는 새로운 이종의 보안로그가 발생할 확률이 높은 클라우드 컴퓨팅 환경에서 기존에 사용되던 관계형 데이터베이스와는 다르게 데이터 저장구조를 변경하지 않아도 되는 장점이 있다.

2. 대용량 보안로그 통합 모델링

본 절에서는 대용량 보안로그 통합을 위한 NoSQL 기반의 데이터 모델링을 기술한다. NoSQL 데이터 모델링은 기존 데이터베이스 시스템 특성과 형식이 다르므로 모델링 접근방법이 다르다. NoSQL을 이용하여 데이터를 모델링하기 위해서는 우선, 기존 데이터베이스 시스템에서는 저장하고자 하는 도메인 모델을 분석하고, 개체간의 관계(Relationship)를 식별하여, 테이블을 정의한 후, 쿼리를 구현하였지만, NoSQL의 경우에는 기존 접근 방법을 역순으로 진행해야 한다.

NoSQL 데이터모델링은 도메인 모델을 선정한 후, 쿼리 결과를 정의하고, 이에 맞게 테이블을 디자인해야 한다. 기존 데이터베이스 시스템의 경우, 테이블을 기반으로 자유롭게 쿼리를 수행할 수 있지만, NoSQL의 경우 복잡한 쿼리 기능이 없기 때문에, 반대로 도메인 모델에서 어떤 쿼리 결과가 필요한지를 정의한 후, 이 쿼리 결과를 얻기 위한 데이터 저장 모델을 역순으로 디자인해야 한다.



[그림 3-5] 대용량 보안로그 통합 모델링 예

보안로그 데이터 통합 과정에 대한 프로세스 진행 구조는 다음과 같다.

(1) LogData Set에 공통된 필드 구분을 확인함.

(2) 공통된 필드 두가지를 키로 조인하여 Row키로 선정함.

(3) 키 이외에 보안로그 데이터에 대한 정보를 입력함.

(4) 입력 된 보안로그내용을 기준으로 통합 작업을 진행함.

(5) 통합 작업시 관리자가 지정한 시작데이터 시간과 끝데이터 시간을 기준값을 함께 저장함.

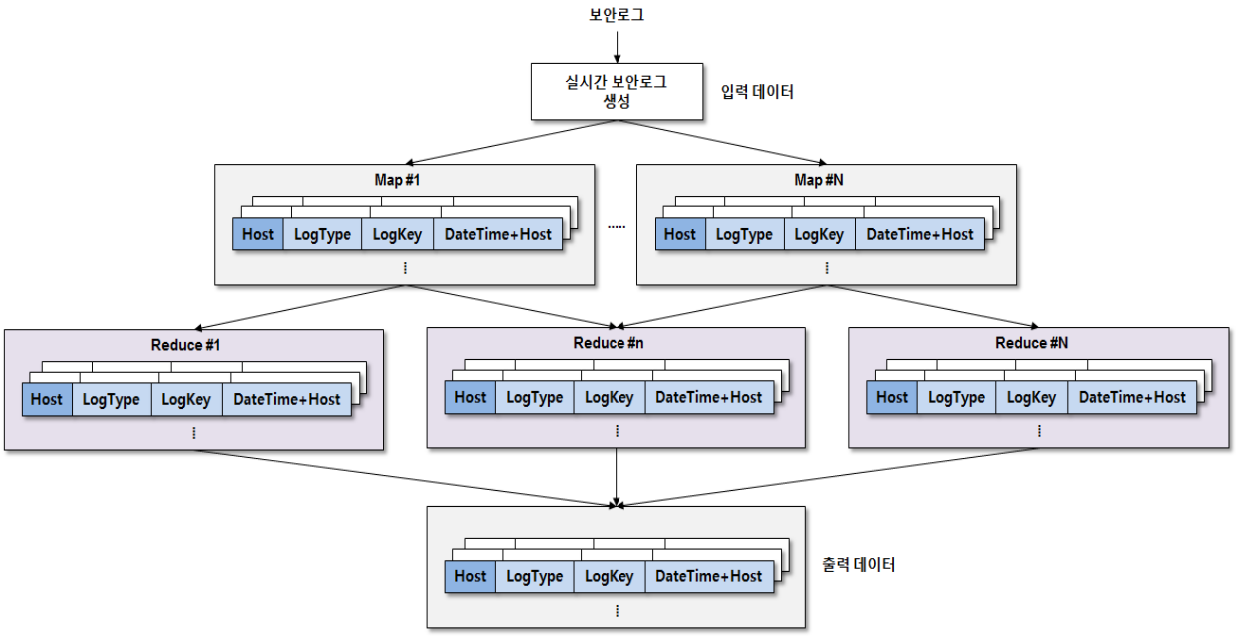
(6) 1 ~ 5번의 과정을 반복함.

NoSQL 데이터 저장시는 Row Key선택을 신중히 해야한다. 이는, Row 키를 기반으로 데이터 정렬 및 읽기를 하기 때문이다. 본 논문에서는 2가지의 값을 병합하여 Row Key를 사용하는 방법을 제안한다. Row Key의 값을 살펴보면 데이터가 1

개의 데이터가 아닌 “-“와 ”:“를 이용하여 2가지의 값을 구분하여 입력하였다. 값을 병합하여 입력하는 이유는 Ordered Key 기반 NoSQL의 경우에는 Key를 기반으로 데이터 추출이 가능하기 때문에 이와 같은 조인된 KEY를 사용하면 TimdeDate와 Host 두가지 영역을 동시에 추출이 가능하기 때문이다. 이는 같은 TimeDate와 같은 Host가 가진 데이터를 따로 구분하지 않아도 다른 TimeDate와 다른 Host 명을 가진 데이터가 시작되면 출력이 되지 않기 때문에 데이터 추출시 불필요한 처리를 줄일수 있다.

3. NoSQL 기반 대용량 보안로그 인덱스

본 논문의 NoSQL 기반 대용량 보안로그 통합은 보안 시스템의 특성상 많은 보안로그 데이터가 짧은 시간에 대량으로 발생하므로 이러한 데이터들을 실시간으로 색인 처리를 하기 위해서 Map/Reduce 기반의 분산처리 방법을 이용한다. Map/Reduce 기반의 분산처리를 이용한 대용량 보안로그 통합은 전처리 부분과 보안로그 통합, 대용량 보안로그 인덱싱 프로세스로 구성된다.



[그림 3-6] 보안로그 인덱스 프로세스

[표 3-6]에서 각 보안 시스템 종류별로 보안로그가 병합을 위해 설계된

MapReduce로 프로그래밍한 의사 코드이다. Mapper 함수에서는 각 보안로그 종류별로 Key를 생성 하고 Count를 생성 한다. [표 3-7]의 Reducer 함수에서는 각 보안로그 종류별로 동일한 Key 값을 갖는 선택하여 병합한다. 이 과정을 반복하여 보안로그 병합이 이루어진다. 이 때 시간 범위는 통합 보안로그 분석 시 관리자가 통합하고 임의로 원하는 시간을 지정할 수 있다. 각 시스템에서 수집된 보안로그는 정규화하고 전처리되어 공통 형식의 텍스트로 변환된다.

다음 [표 3-6] 보안로그 통합에 사용된 Map Reduce 코드중 Map 코드를 나타낸 것이다.

[표 3-6] 보안로그 통합을 위한 Map 의사코드

```
Security Log Integration Pseudo Code
Map (Log)
While ( LogList.Log != NULL)  {
   if (SYSLOG){
   Map_SYSLOG_key (hostname, TimeDate, Demon Info, Log Info)
 }
 else if (WEBLOG){
   Map_WEBLOG_key (ClientIP, TimeDate, Request URL, Log Info)
 }
 else if (IDSLOG){
   Map_IDSLOG_key (DateTime : Hostname : SIP : DIP : Log Info)
 }
}
```

보안로그데이터 통합 과정에 대한 Map 함수의 진행 구조는 다음과 같다.

(1) 보안로그가 처리요청이 들어 왔는지 판단한다.

(2) 보안로그 데이터별로 공통 영역을 선택한다.

(3) 공통 선택한 영역이 또 다른 보안로그 영역과도 일치하는지 확인한다.

(4) 보안로그 영역과 동일하면 해당 영역을 기준으로 데이터 Key를 생성한다.

(5) 생선 된 Key와 값을 임시 저장소에 저장한다.

(6) 보안로그 데이터 1 ~ 5 영역을 반복 한다.

(7) 새롭게 생성할 Key 가 없다면 Reduce 함수를 호출한다.

Map 함수에서는 보안로그 필드에서 공통된 영역을 기준으로 Key를 생성한다. 본 논문에서는 LogInfo를 기준으로 보안로그 데이터 병합을 실시 하였다.

다음 [표3-6] 보안로그 통합에 사용된 Map Reduce 코드중 Reduce 코드를 나타낸 것이다.

[표 3-7] 보안로그 통합을 위한 Reduce 의사코드

```
Security Log Integration Pseudo Code
Reduce (Log)
{
While (LogKeyList != NULL)  {
   for (Start_Time_Value <= End_Time_Value)  {
      if(current SYSLOG_key == next SYSLOG_key){
         Reduce_Key (current SYSLOG, next SYSLOG)
         }
      else if (current WEBLOG_key == next WEBLOG_key){
         merge (current WEBLOG, next WEBLOG)
         }
      else if (current IDSLOG_key == next IDSLOG_key){
   merge (current IDSLOG, next IDSLOG)
  }
 }
}
```

보안로그데이터 통합 과정에 대한 Reduce 함수의 진행 구조는 다음과 같다.

(1) Map 함수에서 생성된 Key 와 값을 임시 저장소에서 읽어온다.

(2) 읽어온 보안로그 Key를 선택한다.

(3) 선택된 Key 별로 같은 Key값을 가진 데이터가 존재 하면 Key 최종 Count를 증가 시킨다.

(4) 더 이상 같은 Key 가 존재하지 않는 다면 다음 Key로 넘어 간다.

(5) 1 ~ 4 번 영역을 반복한다.

Reduce 함수는 발생된 보안로그 마지막 병합 작업을 수행 한다고 볼 수 있다. 보안로그 통합을 통하여 동일한 내용이 발생이 빈번한 보안로그 Data를 효율적으로 통합 하면 데이터 저장공간 및 분석할 전체 데이터 크기를 획기적으로 줄일 수 있다.

Ⅳ. 실험 및 평가

A. 실험 환경 및 실험 시나리오

본 실험은 [표 4-2]와 같은 환경을 구성하여 진행 되었다. 실험에 사용한 데이터베이스는 NoSQL 기반의 HBase-0.94.12 버전을 사용하고 관계형 데이터베이스는 가장 많이 사용되는 MySQL-5.1.69 버전을 선택하여 데이터 입력 질의 성능을 비교 테스트하였다. 보안로그 통합 테스트에 사용된 데이터는 Wikibench에서 제공하는 Wikipedia AccessLog 보안로그 데이터 1억 건을 이용하여 실험 하였다[43]. 전 세계 Wikipedia 사이트에서는 하루 약 10억 건의 AccessLog 데이터가 생성되고 있다. 사용한 데이터셋은 그 중 약 2007년 09월 07에서 08일에 발생한 10%의 데이터를 제공 한 것으로 대용량 보안로그 통합 실험 데이터 셋으로 적합한 크기의 보안로그 데이터 셋 이라고 볼 수 있다.

[표 4-1] AccessLog 데이터 셋 예

3325795705 1191194118.688 http://en.wikipedia.org/wiki/Smallville_Season_7 -
3325795682 1191194118.939 http://en.wikipedia.org/wiki/Yamato_period -
3328738540 1191195450.546 http://en.wikipedia.org/wiki/Digitalism_(band) -
3328738531 1191195450.582 http://en.wikipedia.org/wiki/Kadazan-dusun -
3328738577 1191195450.593 http://en.wikipedia.org/wiki/Betty_Berzon -

[표 4-2] 실험 환경

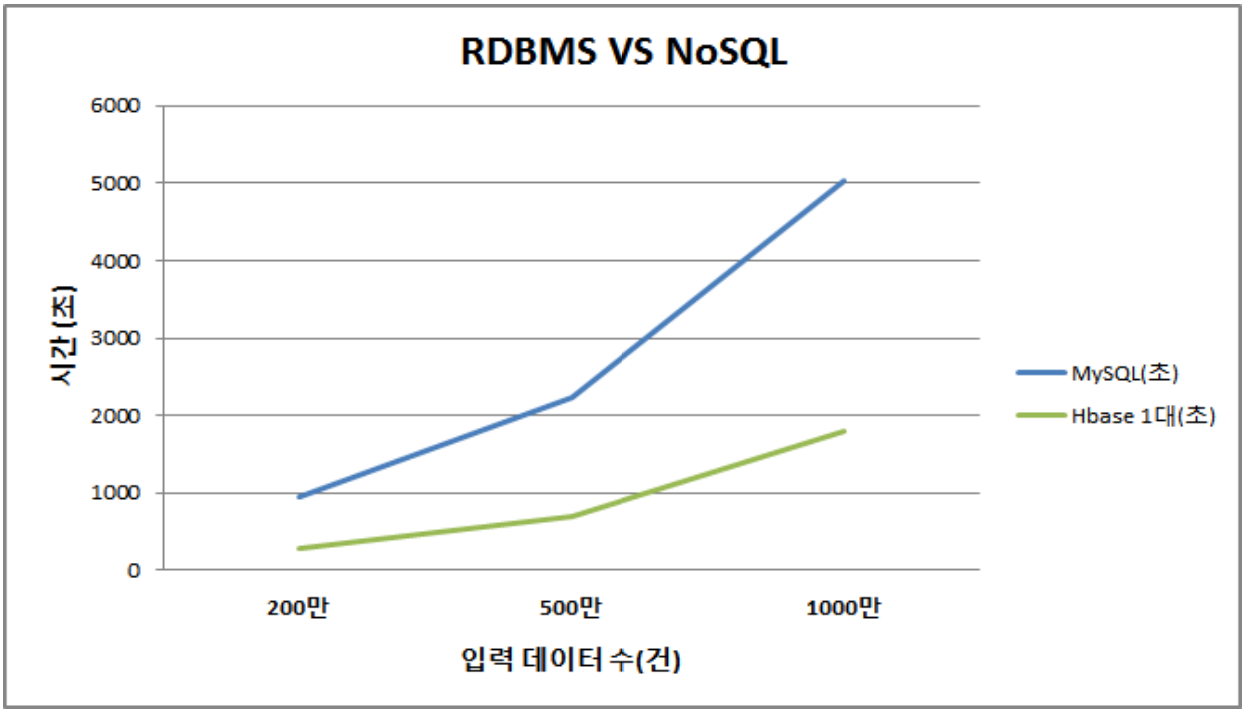
구분	설명
테스트 환경	CPU: Intel Core i7 2.8Ghz , RAM: 12G (3대 할당) CPU: Inter Core i7 3.4Ghz , RAM: 4G (1대 할당)
RDBMS	MySQL-5.1.69
HBase	Hbase-0.94.12, 데이터 처리 쓰레드: 50 머신 수 : 가상머신 VMWare 총4대 사용

B. 성능 테스트

성능 테스트는 크게 두 가지의 실험으로 구성하였다. 첫 번째는 NoSQL 방식이 기존의 RDBMS방식에 비해 얼마나 빠른 속도로 대용량의 보안로그를 수집하여 통합 할 수 있는지를 판단하기 위한 실험으로 각각의 RDBMS 데이터베이스와 HBase 솔루션 1개Node 에서 100만, 500만, 1,000만 건의 보안로그 데이터를 솔루션 및 환경 별로 입력하여 성능을 비교하였다.

두 번째 성능 테스트는 본 논문에서 제안한 보안로그 통합 방법이 클라우드 컴퓨팅 기술로 구성된 MapReduce기법을 이용하여 보안로그 통합에 대한 테스트를 진행한다. 제안된 방법으로 테스트한 결과 대용량 보안로그의 양이 획기적으로 줄어든 것을 알 수 있었다.

[그림 4-1] 는 데이터 삽입 속도 테스트에 대한 결과이다. 삽입 테스트는 보안로그 데이터를 시간 안에 얼마만큼 처리 할 수 있는지, 얼마나 가져올 수 있는지를 성능 비교 테스트이다. 결과 그래프를 확인해 보면 200만 건의 데이터 삽입 시에는 성능의 차이가 크게 나지 않지만 500만 건과 1,000만 건 데이터의 양이 늘어나면 늘어날수록 데이터 삽입 속도의 차이는 큰 것을 확인할 수 있다.



[그림 4-1] RDBMS VS NoSQL 데이터 입력 속도 비교

테스트로 결과로 유추해 볼 때 보안로그 데이터가 2,000만 건, 3,000만 건으로 늘어나면 그 격차는 더욱 커질 것으로 판단된다.

[표 4-3] 는 대한 결과를 정리한 것이다. MySQL 과 Hbase 1노드 두 솔루션의 속도 차이를 살펴보면 200 만 건의 데이터를 삽입하는 속도 차이는 663초이고 500 만 건 데이터 삽입 경우에는 1,542초 약 2.4배정도가 증가 하였고 1,000 만 건의 데이터를 삽입 할 경우에는 3,236초로 속도 차가 약3배 정도가 증가 하였다. 이로 인해 데이터가 늘어날수록 MySQL의 데이터 입력 성능은 점점 저하되는 것으로 확인 할 수 있다.

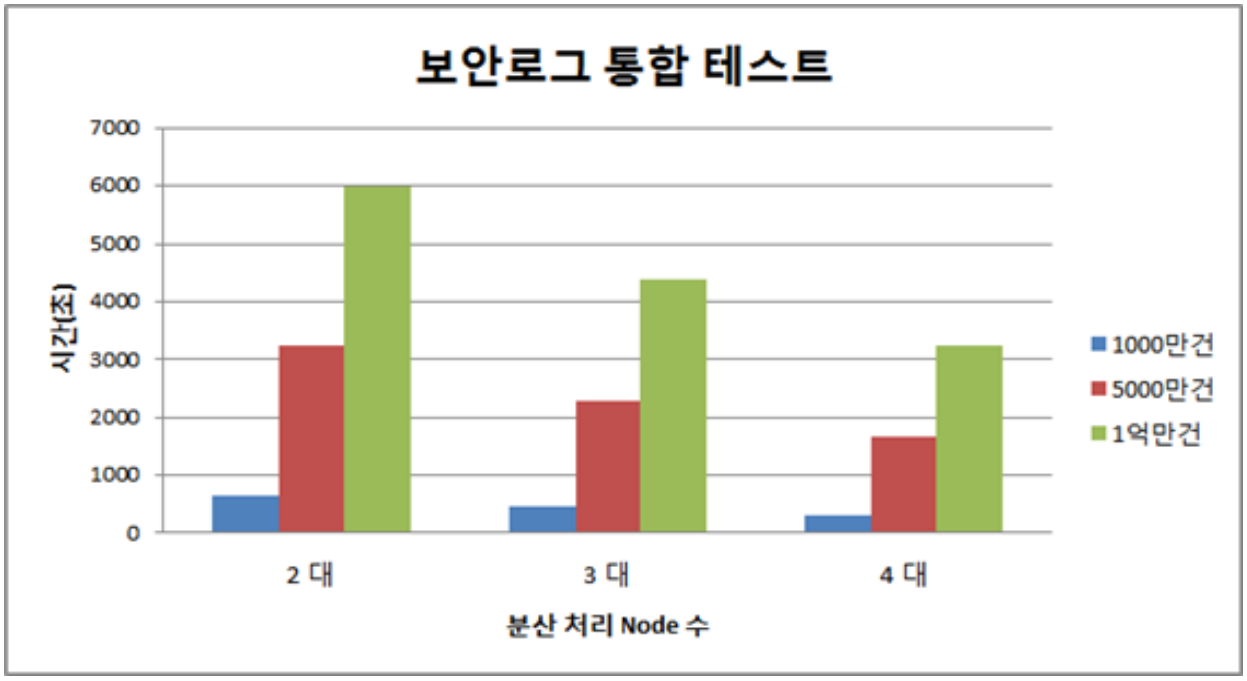
[표 4-3] 데이터 입력 테스트 결과표

구 분	200만	500만	1,000만
HBase 1Node	291	690	1,787
MySQL	954	2,232	5,023
최대 속도차	**663**	**1,542**	**3,236**

(단위 시간: 초)

두 번째 실험인 본 논문에서 제안한 Map-Reduce 기술에 분산 컴퓨팅 환경을 구현하고, 분산 노드의 개수에 따른 제안하는 보안로그 통합 모델의 보안로그 통합 처리시간을 측정하여 비교한 것이다.

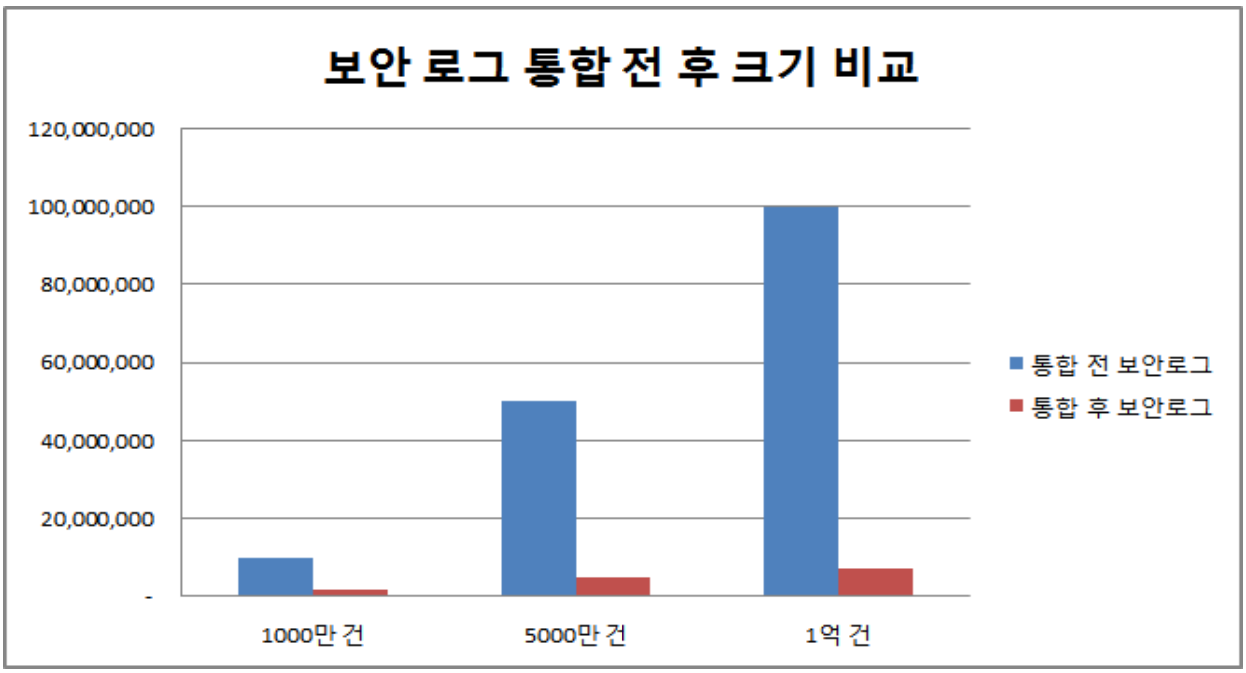
본 테스트를 위해 약 1억 건의 Wikipedia 억세스 보안로그를 1,000만 건, 5,000천만 건, 1억 건으로 분할하여 통합 테스트를 진행하였다. 다음 [그림 4-2]는 보안로그 Map 및 Reduce 과정 10번씩 수행하여 이에 대한 평균처리 시간을 결과 그래프로 표현하였다. 다음 [그림 4-2] 결과를 확인하면 Map-Reduce 기술은 분산 처리하는 Node의 개수가 늘어나면 데이터 처리 속도가 향상됨을 확인 할 수 있다. 이 때문에 클라우드 환경에서 발생하는 대용량의 보안로그도 Node의 개수를 늘림으로써 보다 빠르고 정확한 분석 환경을 제공 될 수 있음을 보여준다.



[그림 4-2] 보안로그 통합 테스트 결과

이는 분산된 환경에서의 확장 가능한 노드의 수가 확보된다면, 예제보다 많은 양의 대용량 보안로그를 근 실시간으로 분석이 가능성이 확인 가능하다.

다음 [그림 4-3]은 보안로그 통합 처리 전과 후에 대한 비교한 그림이다. 그림을 살펴보면 실험에 사용된 보안로그의 통합이 효과적으로 된 것을 확인할 수 있다. 보안로그를 통합하는 처리를 진행함으로 보안로그 저장 공간의 낭비를 획기적으로 줄일 수 있음을 알 수 있다. 이는 이종의 보안로그 필드의 구조 분석을 통하여 공통점이나 중복되는 지점을 찾는다면 새로운 보안로그 발생 시에도 유연한 대처를 할 수 있음을 나타낸다.



[그림 4-3] 보안로그 통합 처리 후 크기 비교

[표 4-4]는 보안로그 통합 처리 전 후 대한 도표이다. 도표를 확인해보면 보안로그의 생성되는 데이터는 보안로그의 내용이 중복되는 내용이나 일반적으로 사용되는 내용이 통합이 가능한 내용이 많고 병합할 Key 선정의 중요성을 나타낸다.

[표 4-4] 보안로그 통합 결과 비교표

구분	통합 전	통합 후	통합 비율
1,000 만	10,000,000	1,712,452	**83 %**
5,000 만	50,000,000	5,098,198	**90 %**
1 억	100,000,000	7,293,475	**93 %**

(단위 : 건)

제안한 방법을 기존 방법론과 비교하면 다음과 같다. 특정 필드 기반 탐지의 경우 새로운 이종의 보안로그 및 새로운 시스템에 대한 통합이 불가능 하나 본 연구는 이종의 보안로그 사용되는 공통된 필드 구조를 이용하여 다양한 이종 보안로그에 대해서도 유연한 통합이 가능하다. 행위 기반 통합 방법은 최근 지능화된 침해사고 사례는 단일 클라이언트 분석이 아닌 침해 사고 발생 시간을 기준으로 전체 시스템 보안로그를 분석해야 한다. 전체 시스템 보안로그를 NoSQL 기반 설계로 다양한 분석의 데이터로 사용이 가능하며 기존 연구과 본 연구의 특징을 비교하면 다음과 같다.

[표 4-5] 기존 통합 방법과 제안된 방법의 비교

구분	특징 필드 기반	행위 기반	본연구
클라우드 환경 지원	불가능	불가능	가 능
대용량 데이터 가용여부	어려움	어려움	가 능
신규 이종 보안로그 추가	어려움	어려움	쉬 움

첫 번째 특징은 클라우드 환경지원 기존 방식의 로그 통합은 로그 데이터 증가로 인한 단일 시스템 성능을 향상시키는(Scale-Up) 방식을 사용하고 있으며 고비용, 확장성, 가용성의 단점을 극복하기 위해 클라우드 환경의 장점인 유연한 확장성 기능을 지원 하도록 구성하였다. 두 번째 특징은 대용량 데이터 처리 가용 여부이다. 실험의 결과로 알 수 있듯 기존 데이터베이스와 데이터 입력 성능의 차이는 Node의 증설에 맞추어 증가한다. 세 번째 특징은 신규 이종 보안로그 추가에 부분이다. 신규 이종의 보안로그 발생 시 공통 필드를 기준으로 데이터 통합 구조를 제안하였고 Map과 Reduce 알고리즘을 통하여 보안로그를 통합하여 전체 시스템 보안로그 데이터 통합으로 분석할 동일 데이터의 크기를 줄임으로 특정 시간에 발생한 사용자 행위 데이터를 효율적으로 제공한다.

제안된 방법은 기존에 보안로그 통합 방법으로 사용된 특정 필드 기반, 행위 기반 방법을 단점에 보완을 목표로 고려하여 설계하여 기존 연구의 가용성, 확장성, 유연성 부분을 개선하였다.

Ⅴ. 결론

오늘날 점차 지능화 돼가는 보안 침해사고의 증가로 인하여 정보 보호의 중요성은 사용자 뿐만 아니라 관리자에게도 강요되고 있다. 특히 클라우드 컴퓨팅 환경의 발달로 신규 서비스는 손쉽게 생성하기 때문에 클라우드 시스템의 집중화 및 거대화되고 있다. 클라우드 컴퓨팅 환경의 특성상 시스템 자원의 공유로 인하여 보안 침해 사고 발생 시 클라우드 환경 전체 내부 시스템에 영향을 끼칠 위험도 또한 높다고 할 수 있다. 이러한 상황에서 시스템을 효과적인 관리를 위해서는 다양한 서비스에서 발생하는 이종의 보안로그 데이터의 지속적으로 수집, 저장, 관리에 대한 새로운 방법이 필요하게 되었습니다.

본 논문에서는 기존의 보안로그 저장소로 사용되던 전통적인 관계형 데이터베이스의 한계를 극복하기 위해 최근 이슈가 되고 있는 대용량 데이터 저장에 획기적인 NoSQL 방식의 데이터베이스를 이용하였다. NoSQL 방식은 기존에 사용된 관계형 데이터베이스보다 가격 및 비용적인 측면에서 유리하고 유연한 분산 처리 환경을 제공하여 데이터 실시간 처리와 확장에 용이하다. 이를 이용하여 대용량 보안로그를 보다 안전하고 효율적으로 수집하는 방법과 수집된 보안로그의 전처리 과정을 통해 특징을 고려하여 통합 시스템 제안하였다.

앞의 실험을 통해 알 수 있었듯이 다량의 보안로그 통합을 위해 이종의 보안로그의 저장 시 NoSQL 플랫폼인 HBase를 기반으로 데이터 구조를 설계하여 유연하고 효과적으로 통합할 수 있는 구조를 제시하였다. 그리고 클라우드 기반 대용량 데이터의 처리 기법인 Map-Reduce를 활용하여 보안로그의 통합을 진행하였다. 진행 결과 Map을 함수에서 Key 설정할 보안로그 필드의 선택의 중요성을 확인하였고 또한 Node 추가를 통하여 대용량 보안로그를 빠르게 통합 저장 처리 가능 시스템을 구축하였다.

따라서 향후 연구는 새로운 이종의 보안로그나 비 정형화된 로그 필드 분석을 효과적으로 하는 방법과 이를 이용하여 더욱 다양한 이종의 보안로그를 취급하고 처리할 수 있는 방안에 대하여 추가적인 연구가 필요할 것으로 보인다. 또한, 더욱 다양한 보안로그를 수집하고 통합한다면 이를 침해 사고 분석 데이터로 사용한다면 시스템 전반적인 보안침해 행위를 파악하고 탐지 해낼 수 있을 것이다.

참고문헌

[1] Jeffrey Dean and Sanjay Ghemawat. "MapReduce: Simplified Data Processing on Large Clusters". Google Inc, 2004.

[2] D Nurmi, R Wolski, C Grzegorczyk "The eucalyptus open source cloud computing system", Cluster Computing and the Grid(CCGRID), pp.124-131, 2009.

[3] G Brunette, R Mogull, "Security Guidance for critical Areas of Focus in Cloud computing V2.1", Cloud Security Alliance, Dec. 2009.

[4] Cloud Security Alliance, "Top Threats to Cloud Computing V1.0", Mar. 2010.

[5] Tal Garfinkel, "A Vitrual Machine Introspection Based Architecture for Intrusion Detection", 2003.

[6] M. Mulazzani, S. Schrittwieser, M. Leithner, M. Huber, E. Weippl, "Dark Clouds on the Horizon: Using Cloud Storage as Attack Vector and Online Slack Space", Proc.of the USENIX Security Symposium, Aug. 2011.

[7] Sarah Shafqat et al. "Conceptual Cloud Computing Employing Identity matrix and Knowledge Warehouse", International Journal of Education and Learning Vol. 1, No. 2, pp.39-48, 2012.

[8] Wei-Yu Chen and Jazz Wang, "Building a Cloud Computing Analysis System for Intrusion Detection System", CLOUD SLAM, Apr. 2009.

[9] Y. Lee, W. Kang, and H. Son, "An Internet Traffic Analysis Method with MapReduce", the 1st IEEE/IETP Workshop on Cloud management, pp.357-361, Apr. 2010.

[10] Borthakur, D., "The Hadoop Distributed File System : Architecture and Design", The Apache Software Foundation, 2007.

[11] Ken Mann, M. Tim Jones, "Distributed computing with Linux and Hadoop", http://www.ibm.com/developerworks/linux/library/l-hadoop/, IBM, 2012 .

[12] M Armbrust, A Fox, R Griffith, AD Joseph "A view of cloud computing" Communications of The Acm Vol. 53, no. 4, pp.51-58, Apr. 2010.

[13] SDS Monteiro, RF Erbacher, “Exemplifying Attack Identification and Analysis in a Novel Forensically Viable Syslog Model“, Third International Workshop on Systematic Approaches to Digital Forensic Engineering pp.57-68, 2008.

[14] Cooley, R., Mobasher, B., and Srivastava, J., “Data preparation for mining world wide web browsing patterns,” Knowledge and Information System, pp.123-132, Vol. 1, No. 1, 1999.

[15] D. Dennig. "An Intrusion Detection Model", IEEE Trans, pp.118-132, 1986.

[16] P Patil, U Patil, “Preprocessing of web server log file for web mining“ World Journal of Science and Technology, Vol. 2, No .3, pp.14-18, 2012.

[17] Asmaa Shaker Ashoor and Prof. Sharad Gore “Importance of Intrusion Detection System (IDS)” International Journal of Scientific & Engineering Research, Vol. 2, Issue. 1, pp.1-4, Jan. 2011.

[18] Hunt, P., Konar, M., Junqueira, F.P., and Reed, B., "ZooKeeper: Wait-free Coordination for Internet-Scale Systems", Proc of Unix Annual Technical Conference, pp.1-14, 2010.

[19] Eric E. Schadt, Michael D. Linderman, Jon Sorenson, Lawrence Lee and Garry P. Nolan “Cloud and heterogeneous computing solutions exist today for the emerging big data problems in biology” Nature Reviews Genetics, Mar. 2011.

[20] DG Feng, M Zhang, Y Zhang, Z Xu “Study on cloud computing security” Journal of Software, pp.71-83, 2011.

[21] S Subashini, V Kavitha, "A survey on security issues in service delivery models of cloud computing", Journal of Network and Computer Applications, Elsevier, pp.1-11, 2011.

[22] K SO, “Cloud computing security issues and challenges”, International Journal of Computer Networks, Vol. 3, Issue. 5, pp.247-255, 2011.

[23] K Ren, C Wang, Q Wang “Security challenges for the public cloud”, Internet Computing, IEEE, 2012.

[24] Sanjay Ghemawat, Howard Gobioff, & Shun-Tak Leung. "The Google File System", Google Inc, 2003.

[25] D Borthakur, J S Sarma, J Gray "Apache Hadoop goes realtime at Facebook"

SIGMOD, pp.1071-1080, 2011.

[26] W Zhou, J Han, Z Zhang, J Dai, "Dynamic Random Access for Hadoop Distributed File System", 32nd International Conference on Distributed Computing Systems Workshops, IEEE, pp.17-22, 2012.

[27] KV Shvachko, "HDFS Scalability: The limits to growth", login, Vol. 35, No. 2, pp.6-16, 2010.

[28] K Zheng, Y Fu "Research on Vector Spatial Data Storage Schema Based on Hadoop Platform" International Journal of Database Theory and Application, Vol. 6, No. 5, pp.85-94, 2013.

[29] K Shvachko, H Kuang, S Radia "The hadoop distributed file system", Mass Storage Systems and Technologies (MSST), IEEE 26th Symposium, 2010.

[30] V Srinivasan, B Bulkowski "Citrusleaf: A Real-Time NoSQL DB which Preserves ACID" The 37th International Conference on Very Large Data Bases Proceedings of the VLDB Endowment, Vol. 4, No. 12, pp1340-1350, 2011.

[31] K Kent, M Souppaya "Guide to computer security log management" NIST special publication, 2006.

[32] 최보민, 공종환, 홍성삼, 한명묵 "NoSQL기반의 MapReduce를 이용한 방화벽 로그 분석기법." 정보보호학회논문지, 제23권, 제4호, pp.667-677, Aug. 2013.

[33] 김완집, 염흥열 "이기종 로그에 대한 통합관리와 IT 컴플라이언스 준수" 한국 정보 보호학회 논문지, 제20권, 제5호, pp.65-73, Oct. 2010.

[34] 이형우 "통합 이벤트 로그 기반 웹 공격 탐지 시스템 설계 및 구현" , 인터넷정보학회 논문지 제11권, 제6호, pp.73-86, Dec. 2010.

[35] J. Shin ,J. Lee, S. Lim, W. Choi, W. Lee, "Integrated Log Extraction Program for an Anomaly Intrusion Detection in Various Environments" 한국IT서비스학회 추계학술대회논문집, pp.511-515, 2009

[36] George, L., "HBase The Definitive Guide, O'ReillyMedia, 2011.

[37] Tom, W., "Hadoop The Definitive Guide", O'ReillyMedia, 2009.

[38] [NoSQL]월간 마이크로소프트웨어 2013년 3월호 Cover Story.

[39] http://en.wikipedia.org/wiki/Big_data

[40] http://www.businesskorea.co.kr/article/1549/big-data-sk-telecom-works-tande

m-local-big-data-provider
[41] HBase, http://hbase.apache.org/
[42] “Cassandra vs MongoDB vs CouchDB vs Redis vs Riak vs HBase vs Membase vs Neo4j comparison” from Kristof Kovacs Blog
[43] http://www.wikibench.eu