MongoDB 환경에서의 데이터보호 및 암호알고리즘별 성능분석에 대한 연구

이선주
고려대학교 컴퓨터정보통신대학원 소프트웨어보안학과
e-mail : uamybest@korea.ac.kr

A Study on Data Protection and Performance Analysis by Cryptographic Algorithm in MongoDB Environment

Sun-Ju Lee
Dept. of Software Security, Korea University Graduate School of Computer & Information Technology

요 약

본 고에서는 약 100 만건의 건강정보를 이용하여 3-Node MongoDB 플랫폼 환경에서 AES, 3DES, ARIA 암호 알고리즘을 이용하여 암호화를 적용하는 방법을 알아본다. 각각의 암호 알고리즘별로 YCSB 성능테스트 툴을 이용하여 다양한 Workload 별로 성능을 테스트를 수행한 뒤, 그 결과를 비교·정리한다.

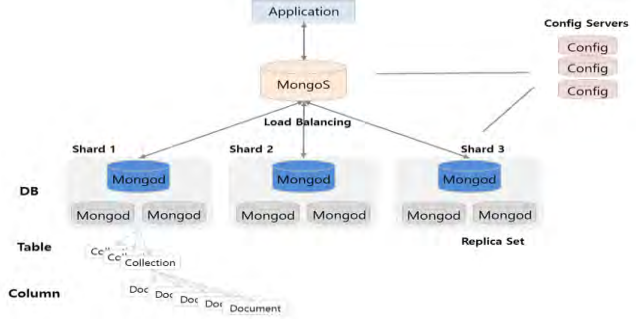
1. 서론

빅데이터 환경에서는 다양한 데이터를 수집하여 대량의 데이터가 저장되는 환경이다. 이러한 저장되는 데이터 유형에는 개인정보와 같이 노출되면 파급영향이 큰 데이터도 함께 저장되어 있다.

그렇지만, 기업 및 공공기관에서 빅데이터 활용을 주저하는 요인 중에는 개인의 민감정보에 대한 암호화를 적용했을때 성능문제로 인해 제대로된 서비스 제공이 힘들다는 측면이 있다. 빅데이터 플랫폼 환경에서 데이터를 저장할때 많이 사용하는 NoSQL DBMS 인 MongoDB 를 이용하여 다양한 암호 알고리즘을 적용할 수 있는 방안과 암호 알고리즘별 성능을 측정하여 그 영향도를 파악하고자 한다.

2. MongoDB 플랫폼 구조

MongoDB 는 다음과 같은 아키텍처로 구성되어 있다. mongos 는 어플리케이션의 요청에 대해 각각의 Shard 서버들의 연결을 해주는 라우터 역할(Load Balancing)을 수행하는 구조로 되어 있다.



[그림 1]MongoDB 아키텍처 구조

MongoDB 의 아키텍처를 기존의 RDBMS 과 비교하여 설명하면, Shard 는 데이터를 저장하는 DB 이며, Collection 은 Table, Document 는 Column 이라고 표현할 수 있다. [표 1]에서 표현된 구성 및 역할에 대해 설명을 하면 다음과 같다.

- mongos 는 Application 의 요청을 받아 Config server 의 데이터의 위치정보를 정보를 참고해 적절한 데이터(mongod) 서버로 요청을 포워딩
- mongod 는 데이터를 저장, 관리하는 서버
- config server 는 Sharding 에 대한 환경 설정 정보 및 데이터 위치정보를 관리
- Replica Set(=Shard)는 이중화를 구현하기 위한 기술로서 하나의 Primary 노드와 2 개 이상의 Secondary 노드로 구성되며, 데이터를 실시간의 서로 공유함으로써 높은 가용성을 지원하는 기술

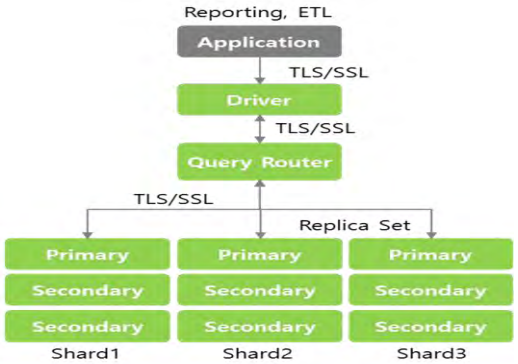
3. MongoDB 의 데이터 보안

3.1 네트워크 구간 암호화

MongoDB 는 네트워크 구간 암호화를 위해 Application Driver 를 통해 데이터 암호화 및 TLS/SSL 을 이용하여 네트워크 구간 암호화 지원(OpenSSL Library 를 이용)한다. 또한 Query Router(mongod)와 Shard(mongos) 및 Node 간에 네트워크 연결시 트래픽 보호를 위해 SSL/TLS 1.1++를 이용하여 중요 데이터에 대한 보안을 한층 강화할 수 있도록 지원하고 있다.

다음의 그림에서와 같이 어플리케이션(사용자)의 요청에 대해 TLS/SSL Driver 를 이용하여 Query

Router(MongoS)와 각 Replica Set(Shard)간의 네트워트 전송시 TLS/SSL 를 적용하여 보안을 할 수 있다.



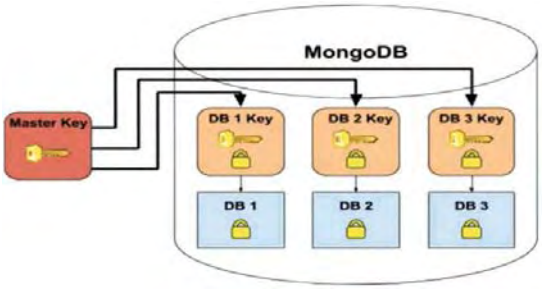
[그림 2]MongoDB 네트워크 구간 암호화

MongoDB 의 TLS/SSL 연결방식에는 4 가지가 있으며, 환경에 맞게 선택적으로 적용할 있도록 되어 있다.

- Disabled : TLS/SSL 사용하지 않음
- allowSSL : 서버간에는 TLS/SSL 사용하지 않고, 들어오는 연결에 대해서 TLS/SS 연결이든 아니든 모두 허용
- preferSSL : 서버간의 연결은 TLS/SSL 사용하고, 들어오는 연결에 대해서는 TLS/SS 연결이든 아니든 모두 허용
- requireSSL : TLS/SSL 암호화 연결만 사용

3.1 네트워크 구간 암호화

MongoDB 의 데이터 저장기술인 WiredTiger Storage Engine 기반의 데이터 암호화를 제공하고 있다. 현재 지원하는 암호 알고리즘은 AES256-CBC, AES256-GCM 암호화 방식만 제공하고 있다. 암호화를 위해 2 가지 형식의 Key 를 사용하고 있는데, 각각의 Replicat Set 를 암호화하기 위한 Master Key 와 Database 를 암호화할 때 사용하는 Internal Key 를 사용하고 있다.



[그림 3]MongoDB 데이터 저장 시 암호화

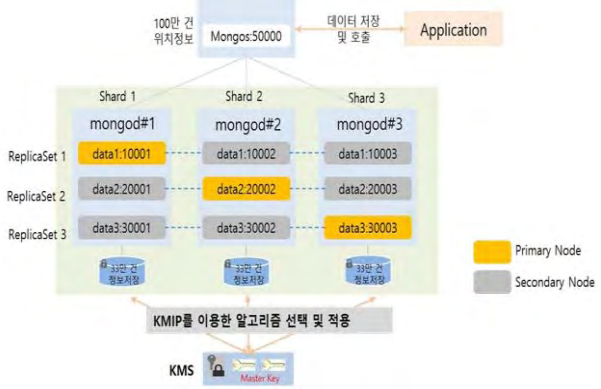
MongoDB 에서 데이터를 암호화하는 과정은 먼저 Mater Key 를 생성한 후 각 Database 를 암호화할 때 사용하기 위한 Internal Key 를 생성하게 된다. 이후 Internal Key 를 이용하여 Data 를 암호화하여 저장하게 되고 Internal Key 를 보호하기 위해 Master key 로 한번 더 암호화를 하는 방식으로 진행하게 된다.

4. 데이터보안 적용 및 성능 테스트환경

4.1 테스트환경

암호화 적용을 위한 테스트환경은 빅데이터 플랫폼 환경에서 일반적으로 사용하는 오픈소스를 기반으로 구성을 하였다. 사내 테스트서버의 하드웨어 용량의 제약사항을 고려하여 최소사양으로 구성하였다.

- 운영체제 : CentOS v7.0(CPU 2, 메모리 8GB, Disk 100GB)
- MongoDB : v3.6.9
- Replica Set : 3-Node 데이터 복제(Primary-Secondary-Secondary)
- Key Management System : Thales v6.0

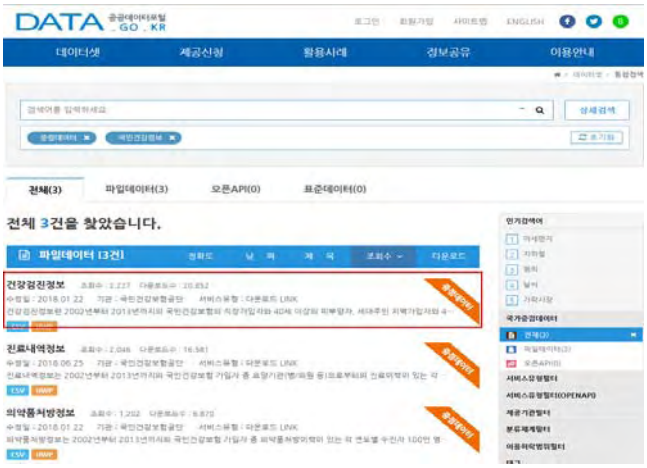


[그림 4]MongoDB 암호화 테스트환경

4.2 테스트 데이터셋

빅데이터시스템은 기존의 legacy 시스템과 다른 많은 데이터를 저장하고 있는 시스템이다. 암호 알고리즘별로 암호화 전·후 테스트를 했을때 성능에 대한 변별력을 확보하기 위해서는 데이터 크기가 중요한 요소로 고려되었다. 빅데이터를 생성해 주는 다양한 Data Generator Tool 이 있지만 시간이 많이 소요되는 점을 고려하여 빅데이터 활성화를 위해 공공데이터 포털사이트(www.data.go.kr)에서 제공해 주는 데이터셋을 활용하였다.

공공데이터 포털사이트에서 제공하는 정보 중 국민건강보험공단이 제공하는 2016 년도의 진료내역정보, 약품처방내역정보, 건강검진정보 약 100 만 건을 mongoDB 에 load 하여 구성하였다.



[그림 5]건강검진정보(2016)

5. 성능테스트 환경

빅데이터 환경에서 YCSB 를 이용하여 성능테스트를 하는 경우 [표 1]과 같이 6 가지 형태의 서로 다른 Wordload 를 적용하면서 성능의 변화를 측정할 수 있다.

Workload Scenario	Operations	Record Selection
A - Update Heavy	Read : 50%, Update : 50%	Zipfian
B – Read Heavy	Read : 95%, Update : 5%	Zipfian
C – Read Only	Read : 100%	Zipfian
D – Read latest	Read : 95%, Insert : 5%	Latest
E – Short ranges	Scan : 95%, Inserts : 5%	Zipfian / Uniform
F – Read-Modify-write	-	Zipfian

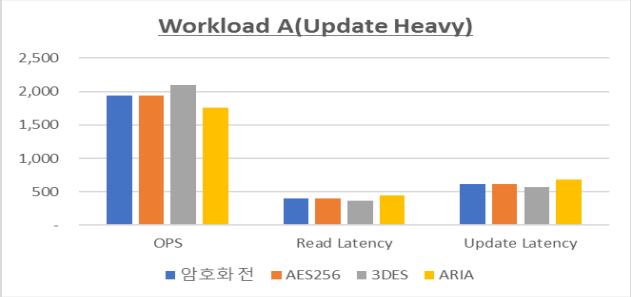
[표 1]YCSB 의 .Workload 유형

각각의 워크로드 적용 시 레코드 선택의 기준은 Zipfian, Latest, Uniform 알고리즘을 선정하였다.

- Uniform : 임의로 레코드를 선정한다.
- Latest : 가장 최근에 삽입된 레코드들을 선정
- Zipfian : 지프의 법칙을 적용하여 일부의 레코드만 선택확률을 높여 선정

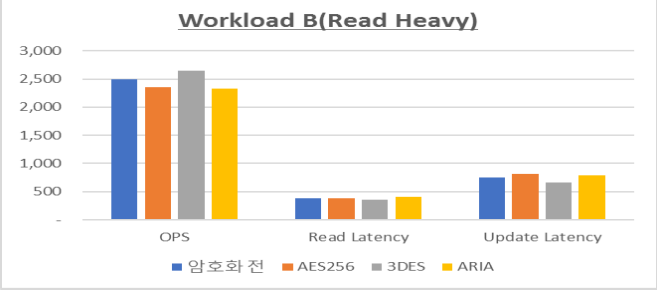
6. 암호 알고리즘별 성능측정 결과

다음으로 6 가지 워크로드를 MongoDB 에 적용하여 수행한 결과를 살펴보도록 한다. 100 만건의 데이터를 저장한 후 에 각 워크로드를 수행했으며, 각 워크로드마다 100,000 개의 연산을 수행하도록 했다.



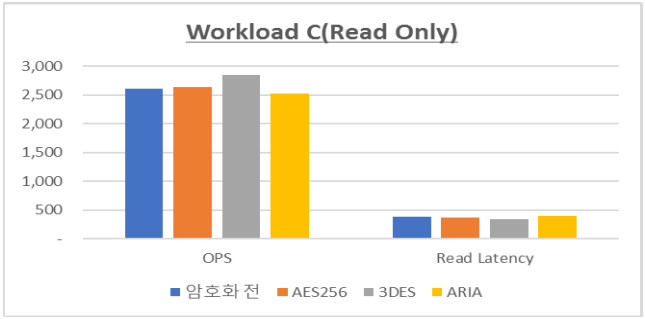
Workload A 는 Update Heavy 가 발생하는 경우로서 Read 50%, Update 50% 조건으로 부하를 발생시킨 결과이다. 성능테스트 결과값은 OPS(Operation Per Second)는 2,000 회 정도로 비슷하게 나왔지만, ARIA 알고리즘의 경우 1,750 OPS 로 상대적으로 낮게 나왔다. Read latency 는 약 400us, Update latency 600us 로서 측정요소별로 차이가 미미한 것으로 나왔다.

Workload A(Update Heavy) 조건에서는 암호화 전과 3 개 암호 알고리즘간 성능차이는 미미하였다.



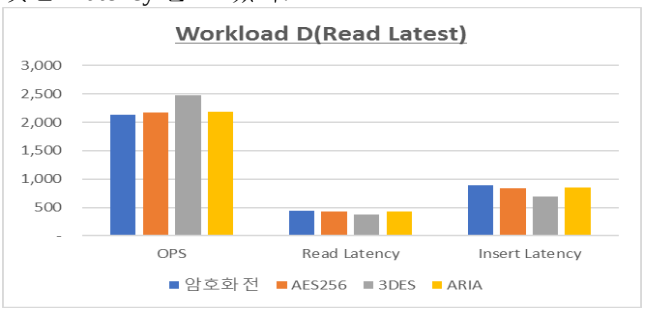
Workload B 는 Read Heavy 가 발생하는 경우로서 Read 95%, Update 5% 조건으로 부하를 발생시킨 결과이다. 성능테스트 결과값은 OPS(Operation Per Second)는 2,500 회 정도로 비슷하게 나왔으며, Read latency 는 약 400us, Update latency 750us 로서 측정요소별로 차이가 미미한 것으로 나왔다.

Workload B(Read Heavy) 조건에서는 암호화 전과 3 개 암호 알고리즘간 성능차이는 미미하였다.



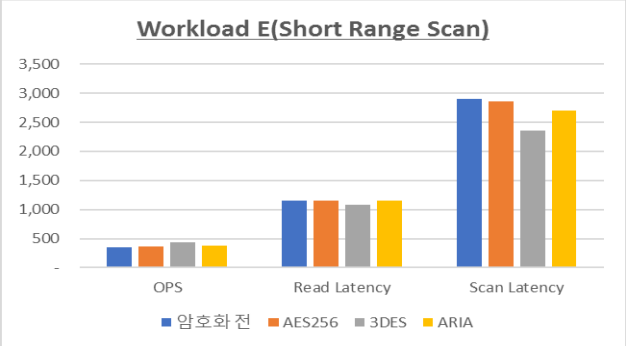
Workload C 는 Read Only 가 발생하는 경우로서 Read 100% 조건으로 부하를 발생시킨 결과이다. 성능테스트 결과값은 OPS(Operation Per Second)는 2,600 회 정도로 비슷하게 나왔으며, Read latency 는 약 400us 로서 측정요소별로 차이가 미미한 것으로 나왔다.

Workload C(Read Only) 조건에서는 암호화 전과 3 개 암호 알고리즘간 성능차이는 미미하였지만, 3DES 의 경우 AES, ARIA 알고리즘에 비해 높은 OPS 와 낮은 Latency 를 보였다.



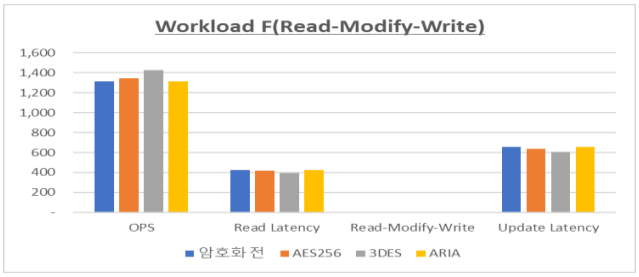
Workload D 는 Read latest 가 발생하는 경우로서 Read 95%, Insert 5% 조건으로 부하를 발생시킨 결과이다. 이는 최근에 Insert 되는 데이터를 가장 많이 Read 될 것이라는 가정에 따라 성능테스트 결과값은 OPS(Operation Per Second)는 2,200 회 정도로 비슷하게 나왔지만, 상대적으로 3DES 알고리즘의 2500 OPS 로 높게 나왔다. Read latency 는 약 450us, Insert latency 850us 로서 측정요소별로 차이가 미미한 것으로 나왔다.

Workload D(Read latest) 조건에서는 3DES 알고리즘을 사용하여 암호화를 적용하는경우에 가장 큰 성능을 발휘하였다.



Workload E 는 Short Range Scan 가 발생하는 경우로서 Scan 95%, Insert 5% 조건으로 부하를 발생시킨 결과이다. 성능테스트 결과값은 OPS(Operation Per Second)는 350 ~ 400 회 정도로 낮게 나왔으며, Read latency 는 약 1,100us, Scan latency 2,300 ~ 2800us 로서 100 만건의 데이터에서 범위검색한 결과가 성능에 많은 영향을 주는 것으로 나왔다.

Workload E(Short Range Scan) 조건에서는 암호화 전과 3 개 암호 알고리즘간 성능차이는 미미하였다.



Workload F 는 Read-Modify-Write 가 발생하는 경우로서 저장되어 있는 데이터를 읽고 변경후 다시 저장하는 조건으로 부하를 발생시킨 결과이다. 성능테스트 결과값은 OPS(Operation Per Second)는 1,300 회 정도로 비슷하게 나왔으며, Read latency 는 약 400us, Read-Modify-Write 1.0us, Update latency 650us 정도로 전반적으로 성능이 떨어지는 것으로 나왔다.

Workload F(Read-Modify-Write) 조건에서는 암호화 전과 3 개 암호 알고리즘간 성능차이는 미미하였다.

7. 결론

빅데이터 플랫폼 중 MongoDB 환경에서 암호화 적용 후 암호 알고리즘별로 성능테스트를 수행한 결과 암호화 전 및 알고리즘별로 성능차이가 미미한 것을 확인하였다. 이를 통해 빅데이터 시스템을 도입하고자 하는 기업 및 공공기관에서 MongoDB 내에 저장되어 있는 민감한 정보에 대해 법적 컴플라이언스 충족을 위해 암호화로 인한 성능저하를 우려를 해소할 수 있는 기회가 되었으면 한다.

본 연구에서는 테스트환경의 자원제약으로 인해 최소환경으로 구성하여 성능테스트를 수행하였다. 빅데이터 플랫폼은 Scale-out 형태로 하드웨어 자원이 증가하는 구조로서 Shard 를 5-node, 7-node,.... 등으로 증가를 시켰을때 성능의 변화 추이에 대해 연구가 필요할 것으로 보인다.

참고문헌

[1]docs.mongodb.com
[2]www.data.go.kr
[3]TTA Journal 빅데이터 벤치마크 현황, 임태형
[4]Cloud Serving Systems, Brian F. Cooper, 2010
[5]www.vormetric.co.kr
[6]MongodDB 와 MySQL 대용량 데이터 처리 비교를 통한 NoSQL 활용 방안 연구, 숭실대, 김은기, 2016
[7[NoSQL 데이터베이스의 IO 워크로드 분석, 한양대, 정성수,2015