Kafka Configuration설정

Apache Kafka Configuration : http://kafka.apache.org/documentation.html#configuration
consumer.properties
producer.properties
zookeeper.properties
server.properties

#. Zookeeper 설정 : config/zookeeper.properties

- zookeeper의 myid설정밑 zookeeper Cluster를 위한 추가설정 / Cluster구축시에 myid의값과 server.#의 값이 값아야 한다.
- Kafka의 대부분의 동작은 Zookeeper와 연계되어 있기 때문에 Zookeeper 없이는 Kafka를 구동할 수 없다. 이 때문에 Kafka 패키지를 받으면 패키지 안에 Zookeeper도 함께 들어있다. 물론 Kafka 패키지 안에 들어있는 Zookeeper 대신 Zookeeper 최신 버전 패키지를 받아서 사용해도 되지만, 패키지 안에 들어있는 Zookeeper는 해당 Kafka 버전과 잘 동작하는 것이 검증된 버전이므로 개인적으로는 패키지 안에 들어있는 Zookeeper의 사용을 권장한다.
그리고 Zookeeper 패키지에 들어있는 sh 파일명과 Kafka 패키지 안에 들어있는 sh 파일명이 다르다.

```
# the directory where the snapshot is stored.
dataDir=/tmp/zookeeper          # 이위치밑에 myid파일을 만들고 그 안에 유니크한 숫자값을 넣어야한다
# the port at which the clients will connect
clientPort=2181
# disable the per-ip limit on the number of connections since this is a non-production config
maxClientCnxns=0

###### Added ############  # 인스턴스 간 통신을 할 때 필요한 설정을 추가
#initLimit=10
#syncLimit=3
#server.1=175.126.56.165:2888:3888   #  server.#에서 #은 인스턴스의 ID
#server.2=175.126.56.166:2888:3888
```

zookeeper 설정 참고 : http://zookeeper.apache.org/doc/trunk/zookeeperAdmin.html#sc_configuration

Property	Default	Description
dataDir	/tmp/zookeeper	the location to store the in-memory database snapshots and, unless specified otherwise, the transaction log of updates to the database. 각 인스턴스들은 설정 파일만 가지고서는 자신의 ID가 무엇인지 알 수 없다. 이 때문에 인스턴스의 ID를 dataDir 디렉토리 아래의 myid라는 파일에 명시해 주어야 한다. 예를 들어 1번 인스턴스의 myid 파일에는 1이 입력되어 있어야 하므로 다음의 커맨드를 실행하여 myid 파일을 생성하면 된다. echo 1> /tmp/zookeeper/myid
clientPort	2181	the port to listen for client connections
maxClientCnxns	0	Limits the number of concurrent connections (at the socket level) that a single client, identified by IP address, may make to a single member of the ZooKeeper ensemble. This is used to prevent certain classes of DoS attacks, including file descriptor exhaustion. The default is 60. Setting this to 0 entirely removes the limit on concurrent connections.
tickTime	2000	the basic time unit in milliseconds used by ZooKeeper. It is used to do heartbeats and the minimum session timeout will be twice the tickTime.
initLimit		Amount of time, in ticks (see tickTime), to allow followers to connect and sync to a leader. Increased this value as needed, if the amount of data managed by ZooKeeper is large. initLimit is timeouts ZooKeeper uses to limit the length of time the ZooKeeper servers in quorum have to connect to a leader

syncLimit		The entry syncLimit limits how far out of date a server can be from a leader.
server.x=[hostname]:nnnnn[:nnnnn]		인스턴스가 3개인 클러스터를 구축할 예정이므로 server.1, server.2, server.3을 설정해 준다. server.#에서 #은 인스턴스의 ID There are two port numbers nnnnn. The first followers use to connect to the leader, and the second is for leader election. The leader election port is only necessary if electionAlg is 1, 2, or 3 (default). If electionAlg is 0, then the second port is not necessary. If you want to test multiple servers on a single machine, then different ports can be used for each server.

#. Broker(Kafka server) 설정 : config/server.properties

- Kafka 클러스터 내의 broker 리스트는 Zookeeper가 관리하므로 설정 파일에는 다른 broker에 대한 정보를 입력할 필요 없이 자신의 broker ID만 명시
- 별도의 파일에 인스턴스의 ID를 명시해야 하는 Zookeeper와는 달리 Kafka는 설정 파일의 broker.id라는 항목에 인스턴스의 ID를 아래와 같이 명시

```
############################# Server Basics #############################
# The id of the broker. This must be set to a unique integer for each broker.
broker.id=0
############################# Socket Server Settings #############################
# The port the socket server listens on
port=9092
# Hostname the broker will bind to. If not set, the server will bind to all interface
host.name=localhost
# Hostname the broker will advertise to producers and consumers. If not set, it uses
# value for "host.name" if configured.  Otherwise, it will use the value returned fr
# java.net.InetAddress.getCanonicalHostName().
#advertised.host.name=<hostname routable by clients>
# The port to publish to ZooKeeper for clients to use. If this is not set,
# it will publish the same port that the broker binds to.
#advertised.port=<port accessible by clients>
# The number of threads handling network requests
num.network.threads=3
# The number of threads doing disk I/O
num.io.threads=8
# The send buffer (SO_SNDBUF) used by the socket server
socket.send.buffer.bytes=102400

# The receive buffer (SO_RCVBUF) used by the socket server
socket.receive.buffer.bytes=102400
# The maximum size of a request that the socket server will accept (protection against OOM)
socket.request.max.bytes=104857600

############################# Log Basics #############################
# A comma seperated list of directories under which to store log files
log.dirs=/tmp/kafka-logs      # 로그파일 저장위치 (로그파일이 안 생기거나 기동이 제대로 안되면 확인)
# The default number of log partitions per topic. More partitions allow greater
# parallelism for consumption, but this will also result in more files across
# the brokers.
num.partitions=1
# The number of threads per data directory to be used for log recovery at startup and flushing at
shutdown.
# This value is recommended to be increased for installations with data dirs located in RAID array.
num.recovery.threads.per.data.dir=1
############################# Log Flush Policy #############################
# Messages are immediately written to the filesystem but by default we only fsync() to sync
# the OS cache lazily. The following configurations control the flush of data to disk.
# There are a few important trade-offs here:
```

```
#    1. Durability: Unflushed data may be lost if you are not using replication.
#    2. Latency: Very large flush intervals may lead to latency spikes when the flush does occur as there will be a lot of data to flush.
#    3. Throughput: The flush is generally the most expensive operation, and a small flush interval may lead to exceessive seeks.
# The settings below allow one to configure the flush policy to flush data after a period of time or
# every N messages (or both). This can be done globally and overridden on a per-topic basis.
# The number of messages to accept before forcing a flush of data to disk
#log.flush.interval.messages=10000
# The maximum amount of time a message can sit in a log before we force a flush
#log.flush.interval.ms=1000

############################# Log Retention Policy #############################
# The following configurations control the disposal of log segments. The policy can
# be set to delete segments after a period of time, or after a given size has accumulated.
# A segment will be deleted whenever *either* of these criteria are met. Deletion always happens
# from the end of the log.
# The minimum age of a log file to be eligible for deletion
log.retention.hours=24                # 메시지의 수명. 수명이 지나면 메시지가 삭제
# A size-based retention policy for logs. Segments are pruned from the log as long as the remaining
# segments don't drop below log.retention.bytes.
#log.retention.bytes=1073741824
# The maximum size of a log segment file. When this size is reached a new log segment will be created.
log.segment.bytes=1073741824
# The interval at which log segments are checked to see if they can be deleted according
# to the retention policies
log.retention.check.interval.ms=300000
# By default the log cleaner is disabled and the log retention policy will default to just delete segments after their retention expires.
# If log.cleaner.enable=true is set the cleaner will be enabled and individual logs can then be marked for log compaction.
log.cleaner.enable=false

############################# Zookeeper #############################
# Zookeeper connection string (see zookeeper docs for details).
# This is a comma separated host:port pairs, each corresponding to a zk
# server. e.g. "127.0.0.1:3000,127.0.0.1:3001,127.0.0.1:3002".
# You can also append an optional chroot string to the urls to specify the
# root directory for all kafka znodes.
zookeeper.connect=localhost:2181
# zookeeper.connect=175.126.56.165:2181,175.126.56.166:2181/cluster01  # 클러스터구축시

# Timeout in ms for connecting to zookeeper
zookeeper.connection.timeout.ms=6000

############################# Added  #############################
auto.create.topics.enable=true
delete.topic.enable=true
default.replication.factor=2
```

```
message.max.bytes=1000000
```

http://kafka.apache.org/081/documentation.html#brokerconfigs

Property	Default	Description
broker.id		Each broker is uniquely identified by a non-negative integer id. This id serves as the broker's "name" and allows the broker to be moved to a different host/port without confusing consumers. You can choose any number you like so long as it is unique.
log.dirs	/tmp/kafka-logs	A comma-separated list of one or more directories in which Kafka data is stored. Each new partition that is created will be placed in the directory which currently has the fewest partitions.
port	9092	The port on which the server accepts client connections.
host.name	175.126.56.165	Hostname of broker. If this is set, it will only bind to this address. If this is not set, it will bind to all interfaces, and publish one to ZK.
num.partitions	1	The default number of partitions per topic if a partition count isn't given at topic creation time.
log.retention.{minutes,hours}	7days	The amount of time to keep a log segment before it is deleted, i.e. the default data retention window for all topics.
zookeeper.connect	175.126.56.165:2181, 175.126.56.166:2181/cluster01	
zookeeper.connection.timeout.ms	6000	The maximum amount of time that the client waits to establish a connection to zookeeper.
auto.create.topics.enable		
delete.topic.enable		
default.replication.factor		
message.max.bytes		

#. consumer.properties

분산으로 설치된 경우는 zookeeper ip만 여러 개 적어주면된다.

```
# Zookeeper connection string
# comma separated host:port pairs, each corresponding to a zk
# server. e.g. "127.0.0.1:3000,127.0.0.1:3001,127.0.0.1:3002"
zookeeper.connect=127.0.0.1:2181                                    # zookeeper 접속 ip
# zookeeper.connect=172.27.241.212:2181,172.27.175.60:2181,172.27.139.205:2181 # zookeper
클러스터일때
# timeout in ms for connecting to zookeeper
zookeeper.connection.timeout.ms=6000          # zookeeper접속 타임아웃
#consumer group id
group.id=test-consumer-group                  # consumer group id
#consumer timeout
#consumer.timeout.ms=5000
```

#. producer.properties

분산으로 설치하는 경우는 metadata.broker.list를 분산 할 노드 수만큼 적어준다.

```
############################## Producer Basics ##############################
# list of brokers used for bootstrapping knowledge about the rest of the cluster
# format: host1:port1,host2:port2 ...
metadata.broker.list=localhost:9092
# metadata.broker.list=1:172.27.241.212:9092,2:172.27.175.60:9092,3:172.27.139.205:9092 # 분산으로
설치하는 경우는 metadata.broker.list를 분산 할 노드 수만큼

# name of the partitioner class for partitioning events; default partition spreads data randomly
#partitioner.class=
# specifies whether the messages are sent asynchronously (async) or synchronously (sync)
producer.type=sync                            #  메시지 동기식 비동기식 (서버까지) 전송 설정 –sync,
async
# specify the compression codec for all data generated: none, gzip, snappy, lz4.
# the old config values work as well: 0, 1, 2, 3 for none, gzip, snappy, lz4, respectively
compression.codec=none                          # 압축 유무 : 0, 1, 2 for none, gzip, snappy,
respectivally
# message encoder
serializer.class=kafka.serializer.DefaultEncoder # message encoder
# allow topic level compression
#compressed.topics=

############################## Async Producer ##############################
# maximum time, in milliseconds, for buffering data on the producer queue
#queue.buffering.max.ms=
# the maximum size of the blocking queue for buffering on the producer
#queue.buffering.max.messages=
# Timeout for event enqueue:
# 0: events will be enqueued immediately or dropped if the queue is full
# -ve: enqueue will block indefinitely if the queue is full
# +ve: enqueue will block up to this many milliseconds if the queue is full
#queue.enqueue.timeout.ms=
# the number of messages batched at the producer
#batch.num.messages=
```