Configuring Kafka log4j appender with Apache Ranger Audits

Apache Ranger by design provides configurable audits destinations. One of destinations is slf4j logging interface, which means audits can be streamed into any logging framework that is bound to slf4j. With this feature, Ranger can support streaming audits into popular messaging bus e.g. Kafka to provide realtime data feeds for various monitoring system.

Enabling audits logging is as simple as adding some configurations in logging properties files and adding a few kafka libraries.

Here is an example of configuring Hive logging properties file to enable sending audit events to Kafka messaging bus.

hive-log4j.properties

```
#
# kafka Appender
#
### for Ranger 0.4.0
log4j.logger.com.xasecure.audit.provider.Log4jAuditProvider=INFO,KAFKA_HIVE_AUDIT
log4j.appender.KAFKA_HIVE_AUDIT=kafka.producer.KafkaLog4jAppender
log4j.appender.KAFKA_HIVE_AUDIT.BrokerList=sandbox.hortonworks.com:6667
log4j.appender.KAFKA_HIVE_AUDIT.Topic=hive_audit_log
log4j.appender.KAFKA_HIVE_AUDIT.layout=org.apache.log4j.PatternLayout
log4j.appender.KAFKA_HIVE_AUDIT.layout.ConversionPattern=%d{ISO8601} %-5p [%t]: %c{2} (%F:%M(%L)) - %m%n
log4j.appender.KAFKA_HIVE_AUDIT.ProducerType=async
```

For Kafka 0.8.1.x, the following jar files should be put into $HIVE_DIR/lib

kafka libraries

```
kafka_2.10-0.8.1.2.2.4.2-2.jar
scala-library-2.10.4.jar
metrics-core-2.2.0.jar
```

In consumer side, some deserializer should be there to parse normalized audit records

hive audit parser

```
public class AuditLogJsonDeserializer extends JsonDeserializer<HiveAuditLogDataModel> {
        private static Logger LOG = LoggerFactory.getLogger(AuditLogJsonDeserializer.class);
        @Override
        public HiveAuditLogDataModel deserialize(JsonParser jp, DeserializationContext ctxt)
                        throws IOException, JsonProcessingException {
                HiveAuditLogDataModel model = new HiveAuditLogDataModel();
                JsonNode node = jp.getCodec().readTree(jp);
                String resource = node.get("resource").asText();
                // split resource to database, table, and column
                String[] tmp = resource.split("/");
                if(tmp.length >= 1){
                        model.db = tmp[0];
                        if(tmp.length >= 3){
                                model.table = tmp[1];
                                model.column = tmp[2];
                        }
                }
                model.action = node.get("action").asText();
                model.clientIP = node.get("cliIP").asText();
                SimpleDateFormat formatter = new SimpleDateFormat("yyyyMMdd-HH:mm:ss.SSS-Z");
                try{
                        model.timestamp = formatter.parse(node.get("evtTime").asText()).getTime();
                }catch(Exception ex){
                        LOG.error("fail converting evtTime in hive audit log", ex);
                }
                model.user = node.get("reqUser").asText();
                return model;
        }
}
```