KIP-44 - Allow Kafka to have a customized security protocol

Status

Current state: *Under Discussion*

Discussion thread: *user thread*, *dev thread*

JIRA: *Will create a JIRA after discussion*

Released: <Kafka Version>

Please keep the discussion on the mailing list rather than commenting on the wiki (wiki discussions get unwieldy fast).

Motivation

At the moment Kafka supports 3 main types of security protocol: Plain, SSL, SASL and its variant SASL_SSL and SASL_PLAINTEXT. Behind-the-scenes Kerberos is the only supported authentication mechanism and SSL is the only supported transport protocol. Anyone who is willing to enable security in Kafka they need to properly config Kerberos and SSL. This is sometimes not desirable for users who already have their own security mechanism installed e.g. token based authentication or other SASL mechs. Due to current design of Kafka users with their own choice of authentication mechanism have no way to reuse what they have but have to employ Kerberos. It will be good for Kafka to provide a plug-able way for users to implement their own security mechanism.

Public Interfaces

A new security protocol is introduced in *org.apache.kafka.common.protocol.SecurityProtocol*

```
CUSTOMIZD(4, "CUSTOMIZED", false)
```

A new config is added to client properties named *customized.network.channelbuilder* which can be set to an implementation of *org.apache.kafka.common.network.ChannelBuilder.* This config takes effect only when *security.protocol* is set to *CUSTOMIZED*. Once *CUSTOMIZED* is used on client side a new listener needs to be added in broker: *CUSTOMIZED://host:port* and *security.inter.broker.protocol* is set to *CUSTOMIZED* if *CUSTOMIZED* security protocol is the only supported protocol.

With the new property *customized.network.channelbuilder* introduced users are able to provide their own implementation of *ChannelBuilder* which contains their own logic of authentication.

Proposed Changes

ChannelBuilder

Since *org.apache.kafka.common.network.ChannelBuilder* is exposed to public constructor is not a good way to pass parameters. Method *configure* is a better place to pass in necessary parameters

```
/**
 * Configure this class with the given securityProtocol, mode, loginType and configs
 */
void configure(SecurityProtocol securityProtocol, Mode mode, LoginType loginType, Map<String, ?> configs)
throws KafkaException;
```

ChannelBuilders

When *SecurityProtocol* is set to *CUSTOMIZED* ChannelBuilders will read the class name from *customized.network.channelbuilder* and then instantiate it.

```
public static ChannelBuilder create(SecurityProtocol securityProtocol, Mode mode, LoginType loginType,
Map<String, ?> configs) {
    ChannelBuilder channelBuilder;

    switch (securityProtocol) {
        case SSL:
            requireNonNullMode(mode, securityProtocol);
            channelBuilder = new SslChannelBuilder();
            break;
        case SASL_SSL:
        case SASL_PLAINTEXT:
            requireNonNullMode(mode, securityProtocol);
            if (loginType == null)
                throw new IllegalArgumentException("`loginType` must be non-null if `securityProtocol` is `" +
securityProtocol + "`");
            channelBuilder = new SaslChannelBuilder();
            break;
        case CUSTOMIZED:
            Class<?> channelBuilderClass = (Class<?>) configs.get(CommonClientConfigs.CUSTOMIZED.NETWORK.
CHANNELBUILDER);
                        channelBuilder = (ChannelBuilder) Utils.newInstance(channelBuilderClass);
            break;
        case PLAINTEXT:
        case TRACE:
            channelBuilder = new PlaintextChannelBuilder();
            break;
        default:
            throw new IllegalArgumentException("Unexpected securityProtocol " + securityProtocol);
    }

    channelBuilder.configure(securityProtocol, mode, loginType, configs);
    return channelBuilder;
}
```

With above changes users who want to include their own authentication logic can set *security.protocol* to *CUSTOMIZED*, *customized.network.channelbuilder* to *XYZChannelBuilder* where the authentication and secure transport logic resides and add a new listener *CUSTOMIZED://host:port* in broker.

Compatibility, Deprecation, and Migration Plan

It will not impact any of existing clients. When clients upgrade to new version, they can set customized.network.channelbuilder

Rejected Alternatives

New SASL mechanism

As discussed in KIP-43 an alternative way is to provide a plug-able SASL. But this implementation is bound to SASL implementation only. Authentication users supply must comply with SASL. This KIP provides higher level of abstraction as it is security protocol agnostic and up to users how they want to implement it. This KIP can certainly coexist with KIP-43 which is more focus on SASL mechanism extension.