KIP-437: Custom replacement for MaskField SMT

Status

Current state: Adopted

Discussion thread: here

Vote thread: here

JIRA: *Unable to render Jira issues macro, execution error.*

PULL REQUEST: #6284

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

Motivation

The existing MaskField SMT uses null value equivalent for all types of fields masked.

We want to introduce custom replacement value for any string or numeric fields, which may be optionally specified in config.

Use cases:

- mask out any *Personally Identifiable Information* (**PII**) with custom replacement
  - SSN ***-***-****
  - IP XXX.XXX.XXX.XXX
  - Telephone number +0-000-000-0000
  - etc

'replacement' param value, if specified in config, will be applied to all fields in the 'fields' param list. This way user can mask any string or numeric value with any literal of his choice.

Public Interfaces

New param 'replacement' will be added to the MaskField SMT config. Param is optional so this change is backward compatible.

name	description	type	default	valid values	importance
replacement	custom value replacement, that will be applied to all 'fields' values (numeric or non-empty string values convertable to the 'fields' type(s)).	String	null	non-empty string with numeric or text value, which can be converted to 'fields' type(s)	low

Example config:

config.properties

```
transforms=SSNMask,IPMask,PhoneMask

transforms.SSNMask.type=org.apache.kafka.connect.transforms.MaskField$Value
transforms.SSNMask.fields=ssn
transforms.SSNMask.replacement=***-***-****

transforms.IPMask.type=org.apache.kafka.connect.transforms.MaskField$Value
transforms.IPMask.fields=ipAddress
transforms.IPMask.replacement=xxx.xxx.xxx.xxx


transforms.PhoneMask.type=org.apache.kafka.connect.transforms.MaskField$Value
transforms.PhoneMask.fields=office,mobile
transforms.PhoneMask.replacement=+0-000-000-0000
```

With config described above there will be 3 transformations, first and second will replace value in a single field, while third will be applied to both 'office' and 'phone' fields.

Restrictions

⚠️ If replacement provided cannot be converted to the target field type or field type is not supported then org.apache.kafka.connect.errors.DataException will be thrown.

ℹ️ Only numeric (byte, short, int, long, float, double, BigInteger, BigDecimal) and String fields can be replaced with custom value.

ℹ️ Replacing Date, Map or List with custom value seems to be useless and makes conversion logic more complicated.

Proposed Changes

Add 'replacement' param to config definition:

MaskField.java

```
private static final String REPLACEMENT_CONFIG = "replacement";

public static final ConfigDef CONFIG_DEF = new ConfigDef()
        .define(FIELDS_CONFIG, ConfigDef.Type.LIST, ConfigDef.NO_DEFAULT_VALUE, new NonEmptyListValidator(),
ConfigDef.Importance.HIGH, "Names of fields to mask.")
        .define(REPLACEMENT_CONFIG, ConfigDef.Type.STRING, null, new ConfigDef.NonEmptyString(), ConfigDef.
Importance.LOW, "Custom value replacement, that will be applied to all"
                + " 'fields' values (numeric or non-empty string values only).");
```

Define mapping functions for custom replacement value:

MaskField.java

```
private static final Map<Class<?>, Function<String, ?>> REPLACEMENT_MAPPING_FUNC = new HashMap<>();

static {
    REPLACEMENT_MAPPING_FUNC.put(Byte.class, v -> Values.convertToByte(null, v));
    REPLACEMENT_MAPPING_FUNC.put(Short.class, v -> Values.convertToShort(null, v));
    REPLACEMENT_MAPPING_FUNC.put(Integer.class, v -> Values.convertToInteger(null, v));
    REPLACEMENT_MAPPING_FUNC.put(Long.class, v -> Values.convertToLong(null, v));
    REPLACEMENT_MAPPING_FUNC.put(Float.class, v -> Values.convertToFloat(null, v));
    REPLACEMENT_MAPPING_FUNC.put(Double.class, v -> Values.convertToDouble(null, v));
    REPLACEMENT_MAPPING_FUNC.put(String.class, Function.identity());
    REPLACEMENT_MAPPING_FUNC.put(BigDecimal.class, BigDecimal::new);
    REPLACEMENT_MAPPING_FUNC.put(BigInteger.class, BigInteger::new);
}
```

Define replacement function:

MaskField.java

```
private static Object maskWithCustomReplacement(Object value, String replacement) {
    Function<String, ?> replacementMapper = REPLACEMENT_MAPPING_FUNC.get(value.getClass());
    if (replacementMapper == null) {
        throw new DataException("Cannot mask value of type " + value.getClass() + " with custom replacement.");
    }
    try {
        return replacementMapper.apply(replacement);
    } catch (NumberFormatException ex) {
        throw new DataException("Unable to convert " + replacement + " (" + replacement.getClass() + ") to
number", ex);
    }
}
```

Compatibility, Deprecation, and Migration Plan

The new configuration property in this proposal is backward compatible, so any existing connector configurations that use the MaskField SMT will continue to work as-is with no behavioral changes. To use the replacement feature, such existing connector configurations will need to be modified to add a new `replacement` property.

Rejected Alternatives

First version of the improvement, proposed in the PR, used JSON map of maskField replacement, this approach had both pros and cons:

➕ JSON-like definition for custom replacement makes MaskField SMT more functional, as user can provide replacements for different fields using single transformation like it was done for default replacements;

⛔ JSON-like definition is complicated for the user;

⛔ JSON-like definition is more error-prone;

⛔ JSON-like definition is less readable;

⛔ JSON-like definition processing makes code more complex;

⛔ JSON-like definition does not comply with existing configurations for anything in Connect.

Based on the reasons stated above, we choose using single replacement for all mask fields provided, as it is always possible to use multiple MaskField SMT instances in a connector.