# KIP-978: Allow dynamic reloading of certificates with different DN / SANs



## Status

**Current state**: *Accepted*

**Discussion thread**: *here*

**JIRA**: *here*

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

## Motivation

When dynamically reconfiguring TLS truststores and keystores of Kafka listeners, Kafka performs the following validations:

- Checks if the Subject / Distinguished Name (DN) of the old and new keystore is the same as before
- Checks if the Subject Alternative Names (SANs) of the new keystore contain all names from the old keystore (i.e. adding new SANs is allowed, removing old SANs is not)
- If the truststore / keystore being updated is used by the inter-broker listener, it also tries to perform a *test* TLS handshake to validate that the keystore and truststore are compatible with each other and will not break the inter-cluster communication.

While these validations do not seem to be explicitly mentioned in the KIP-226 / KAFKA-6241, the follow-up issues (KAFKA-10279, KAFKA-14770) seem to suggest that these checks are there to protect the user from unintentionally breaking the TLS communication. While this might indeed protect the users in some situations, it also prevents them from changing the certificates when desired without restarting the brokers. For example:

- When running Kafka in a dynamic environment where DNS names (SANs) are changing often.
- When moving to a brand new CA and server certificate.
- When removing a SAN name that is not used anymore from the certificate and having the old certificate with the old SAN might be considered a security risk.

These changes today require a restart of the brokers. And in some cases - such as changing the CA - even multiple restarts:

- Establish trust for the new CA (first rolling update)
- Move to the new server certificate (second rolling update)

In situations like this, being able to update the keystores dynamically could be a big advantage. It would cause less disruptions and make the operation faster. It would also make it easier to use short-lived certificates even if their DN or SANs change and update them dynamically without the need for restart and thus making the Kafka cluster more secure.

Kafka brokers currently don't terminate any existing connections when the certificates are updated. The new certificates will be used only for new connections. So when the users need to update the CA and the server certificates for some critical security reasons such as compromised CA, they would either need to perform a rolling update of the Kafka cluster to terminate any existing connections or handle it on some different layer outside of Kafka. For example by terminating the existing connections on the network layer. Termination of the existing connections by Kafka itself is not part of this KIP.

## Public Interfaces

This change introduces two new broker configuration options named `ssl.allow.dn.changes` and `ssl.allow.san.changes`. They are described in the next section.

## Proposed Changes

This KIP proposes adding two new configuration options `ssl.allow.dn.changes` and `ssl.allow.san.changes`:

- When `ssl.allow.dn.changes` is set to true, the check for DN changes during dynamic configuration updates will be skipped.
- When `ssl.allow.san.changes` is set to true, the check for SAN changes during dynamic configuration updates will be skipped.

Using two separate options provides more flexibility if some users would want to disable only one of the checks. The default value of both options will be `false`. That way, the default behavior would not change and any users will still have the certificates validated unless they explicitly disable it. The new options will not have any impact on the TLS handshake check done for inter-broker listeners. It will be still executed to make sure the keystore and truststore match even if the DN and SAN check is disabled.

The `ssl.allow.dn.changes` and `ssl.allow.san.changes` options will not be dynamically configurable.

# Compatibility, Deprecation, and Migration Plan

The default behavior for any existing and new users will not change. Only users who explicitly set the new options to true would be affected by this proposal.

# Rejected Alternatives

## Disabling the checks completely

Another option would be to remove the existing DN and SAN check completely. While it might protect the users in some situations, it would still allow them to break the TLS setup in some situations. However, that would mean a change for any user who relies on this check. So I rejected this alternative and proposed the new configuration option instead.

## Using one option for both checks

Another considered variant was using a single option `ssl.allow.dn.and.san.changes` to disable both DN and SAN checks at once. This would provide less flexibility if users want to disable only one of the checks.