JetStream Consumers Multiple Filters

Metadata	Value
Date	2023-01-18
Author	@Jarema
Status	Approved
Tags	jetstream, client, server

Context and Problem Statement

Initially, JetStream Consumers could have only one Filter Subject. As the number of feature requests to specify multiple subjects increased, this feature was added. That could also reduce the number of Consumers in general.

Context

Server PR: https://github.com/nats-io/nats-server/pull/3500

Design

Client side considerations

To implement the feature without any breaking changes, a new field should be added to Consumer config, both on the server and the clients.

1. The new field is:
   `FilterSubjects []string json:"filter_subjects"`
2. Subjects can't overlap each other and have to fit the interest of the Stream. In case of overlapping subjects, error (10136) will be returned.
3. Only one of `FilterSubject` or `FilterSubjects` can be passed. Passing both results in an error (10134)
4. Until future improvements, only old JS API for consumer creation can be used (the one without `FilterSubject`) in consumer create request. Using new API will yield an error (10135).
5. Each client, to support this feature, needs to add a new field that Marshals into a json array of strings.
6. To ensure compatibility with old servers that are not aware of `filter_subjects` field, client's should check the returned info (from update or create) if the filters are set up properly.

Example

```
{
  "durable_name": "consumer",
```

```
  "filter_subjects": ["events", "data"]
}
```

6. Client does not have to check if both, single and multiple filters were passed, as server will validate it. Client should add new errors to return them in client language idiomatic fashion.

Server side

To make this change possible and reasonable peformant, server will have a buffer of first message for each subject filtered and will deliver them in order. After delivering message for a subject, buffer for that subject will be filled again, resulting in close to no overhead after the initial buffer fill.

This can be optimized but will not affect API.