Cortex™

Product Description

V1.2-2022

HENDRIKX ITC

Trademarks and Permissions

Cortex, 1OPTIC and other Hendrikx ITC trademarks are trademarks of Hendrikx ITC, Ltd. All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services, and features are stipulated by the contract made between Hendrikx ITC and the customer. All or part of the products, services, and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees, or representations of any kind, either express or implied. The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Hendrikx ITC, Ltd.

Address:	Hendrikx ITC Spoorlaan 21K Tilburg, Noord-Brabant The Netherlands
Website:	https://www.hendrikx-itc.nl
Email:	support@hendrikx-itc.nl

HENDRIKX ITC

Contents

1. Product Positioning 3
2. Product Characteristics 3
   2.1 Low Latency 3
   2.2 Multi-protocol 3
   2.3 Filtering 3
   2.4 Deduplication 3
   2.5 Message-Based Consumer Informing 3
   2.6 Data Retention Rules 3
   2.7 Audit Logs 4
   2.8 Relational Database Backed State 4
3. Architecture 4
   3.1 Position 4
   3.2 Software Architecture 5
   3.3 Virtualization 5
4. Use Cases 6
   4.1 Inter-application Data Exchange 6
   4.2 Archival 6
5. Configuration 6

HENDRIKX ITC

1. Product Positioning

Cortex is a high-performance file broker that supports multiple protocols for consuming and publishing data between systems that require file-based data exchange. It provides event-based, low latency distribution with filtering and deduplication.

2. Product Characteristics

2.1 Low Latency

Cortex provides low latency data distribution by providing event-based handling of incoming data and a compiled code engine with high-performance in-memory communication channels.

2.2 Multi-protocol

Cortex supports multiple protocols for both the producing and consuming side, so that maximum flexibility is offered for connecting producers and consumers.

2.3 Filtering

For consumers that only require a subset of data of a specific source, advanced file property- or content-based filtering can be configured.

2.4 Deduplication

Sources that sometimes by design or by accident send the same data multiple times can be configured so that the data is only sent once to the consumers. The method for detecting duplicate data is configurable up to a byte-level comparison based on hashes.

2.5 Message-Based Consumer Informing

Consumers can be informed of new data using a message queue. This makes it possible for pull-consumers to be informed immediately when new data becomes available.

2.6 Data Retention Rules

For all data handled by Cortex, data retention rules can be defined so that company policies can be enforced automatically.

2.7 Audit Logs

All data distributions are logged so that it can always be traced which consumer received what data. Based on hashes, these logs are tamper-proof. When the delivered file is altered on the consumer side, this can always be proven based on the stored hash.

2.8 Relational Database Backed State

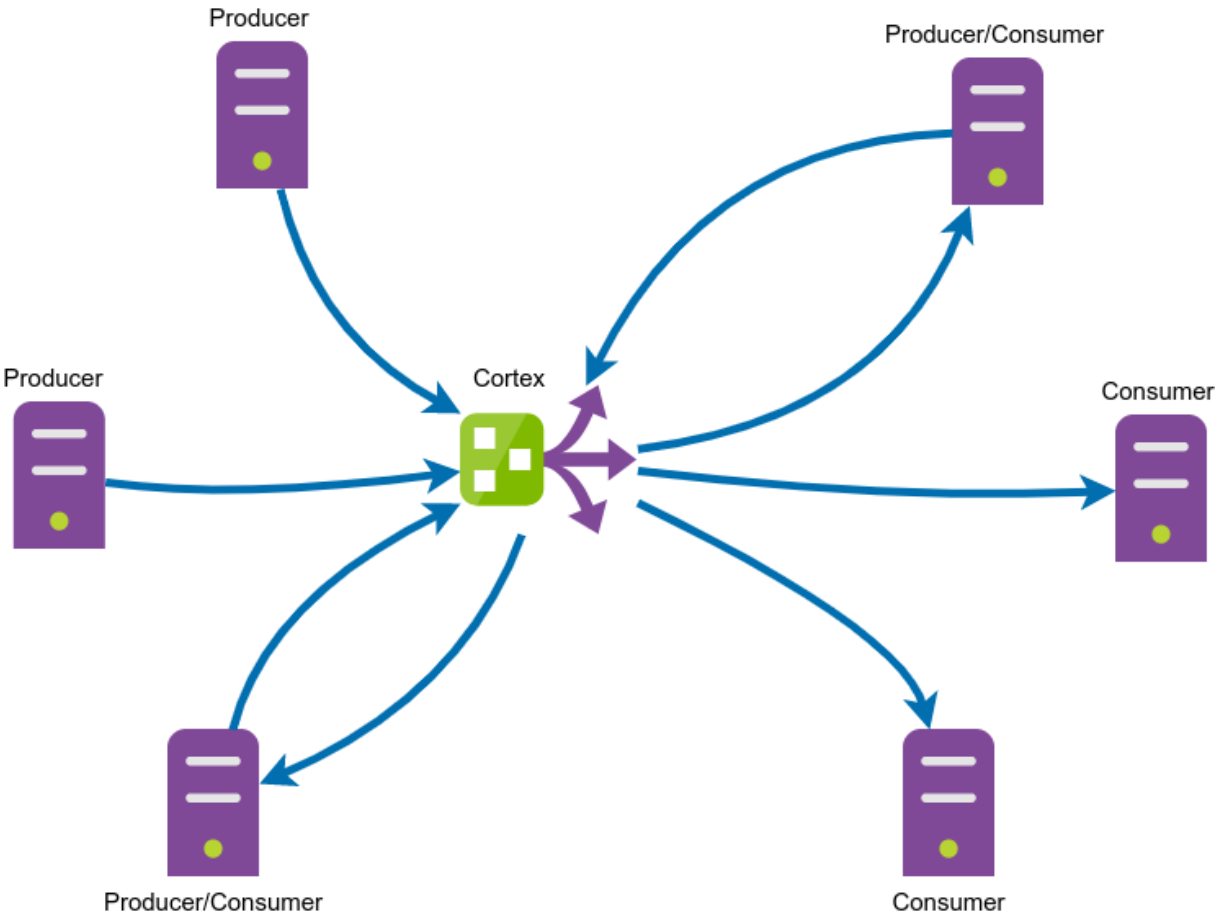
The state of Cortex is backed by a PostgreSQL relational database so that all state transitions are securely persisted to disk.

3. Architecture

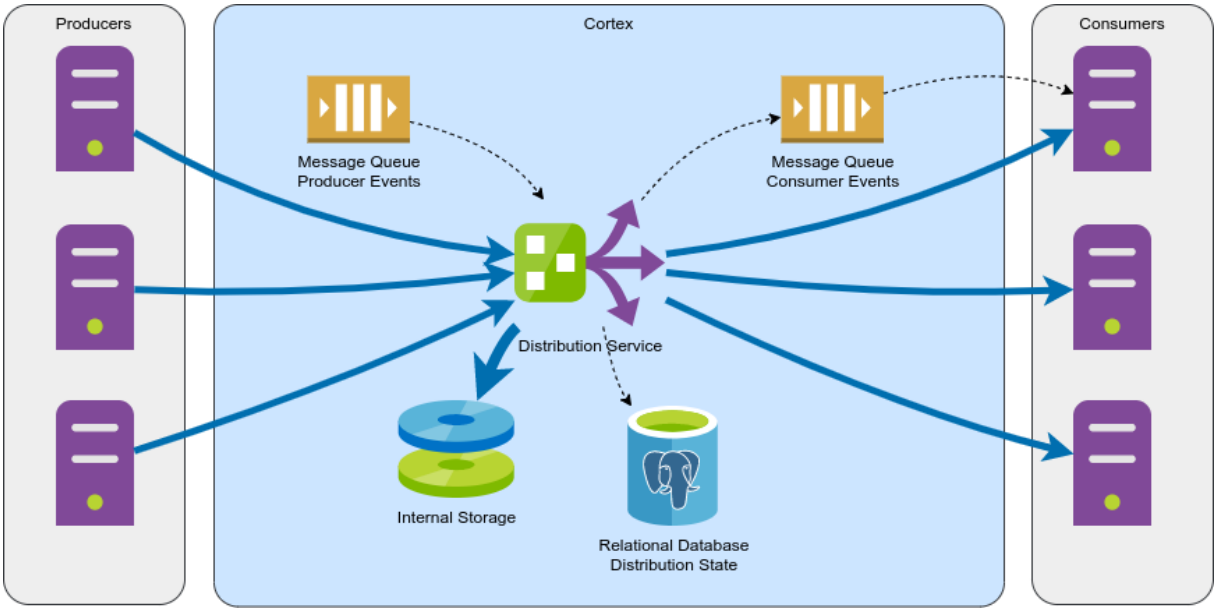
Cortex consists of a core broker service that is responsible for the actual distribution of the data and several supporting services.

3.1 Position

The position of Cortex is at the center of any set of data exchange parties.



3.2 Software Architecture



3.2.1 Components

1. Distribution Service - The service responsible for the actual distribution from producers to consumers.
2. Internal Storage - Storage area for all the data that is received by Cortex and held until the end of the configured retention period.
3. Relational Database Distribution State - The relational database engine that maintains all persistent states of Cortex. This includes what files have been delivered to Cortex and all their properties. It also includes what files have been delivered to what consumers.
4. Message Queue Producer Events - A queue for information of newly available data on producers in the case of pull data streams.
5. Message Queue Consumer Events - A queue for information of newly available data for consumers in the case of pull data streams.

3.3 Virtualization

All components of Cortex are designed to be able to run in a virtualized environment. Advantages of a Cortex deployment on a virtualized environment (non-exhaustive list):

1. Easy to scale up or down when the requirements for the system change.
2. Easy to create a strong logical separation of the components by setting them up on multiple virtual machines.
3. More options for dividing the load of the individual components.

4. Use Cases

Cortex is a very generic data broker solution, but in this section, a number of specific use cases are provided.

4.1 Inter-application Data Exchange

One of the most common cases is the exchange of data between 2 or more applications. 1 or more applications are producers and 1 or more applications are consumers.

4.2 Archival

The use of an archive target for a data source is common practice in any configuration, but it can be a use case in itself. By configuring a local directory target for a data source with data retention that fits your archive requirements, it is trivial to create an archive for a data stream. This can be used as an endpoint of the data or 'tapped' from a stream between applications.

5. Configuration

The configuration of Cortex is done using YAML-based configuration files that are human-readable and are designed to integrate well with configuration management systems (e.g. Puppet, Ansible, SaltStack).

Component	vCPUs	Memory
Distribution Service	6	16GB
Internal Storage	-	-
Relational Database Distribution State	4	16GB
Message Queue Producer Events	2	4GB
Message Queue Consumer Events	2	4GB