

SYNERGY 2026
DEVPARTNER CONFERENCE

Streaming Integration Platform: The Full Toolkit

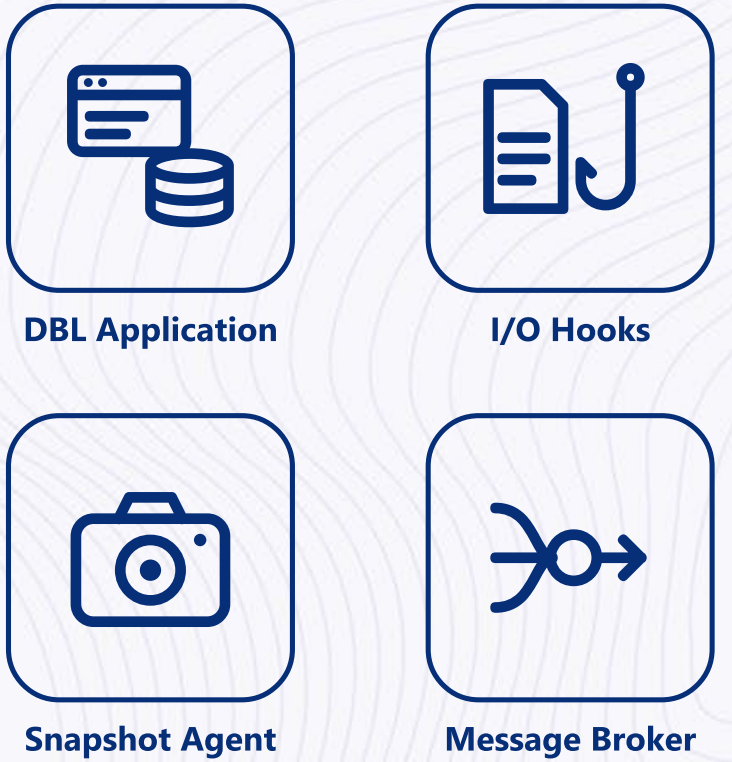
Steve Ives, Solutions Architect

- A new product from Synergex
- Enables DBL applications to transition into distributed systems
- Enhances fault tolerance, concurrency, and extensibility
- Real-time data flow across modern systems with minimal (if any) change to existing code
- Focus on performance & resilience
- Usable from most DBL environments
- Fair warning: We'll be getting "into the weeds"



System of Record Components

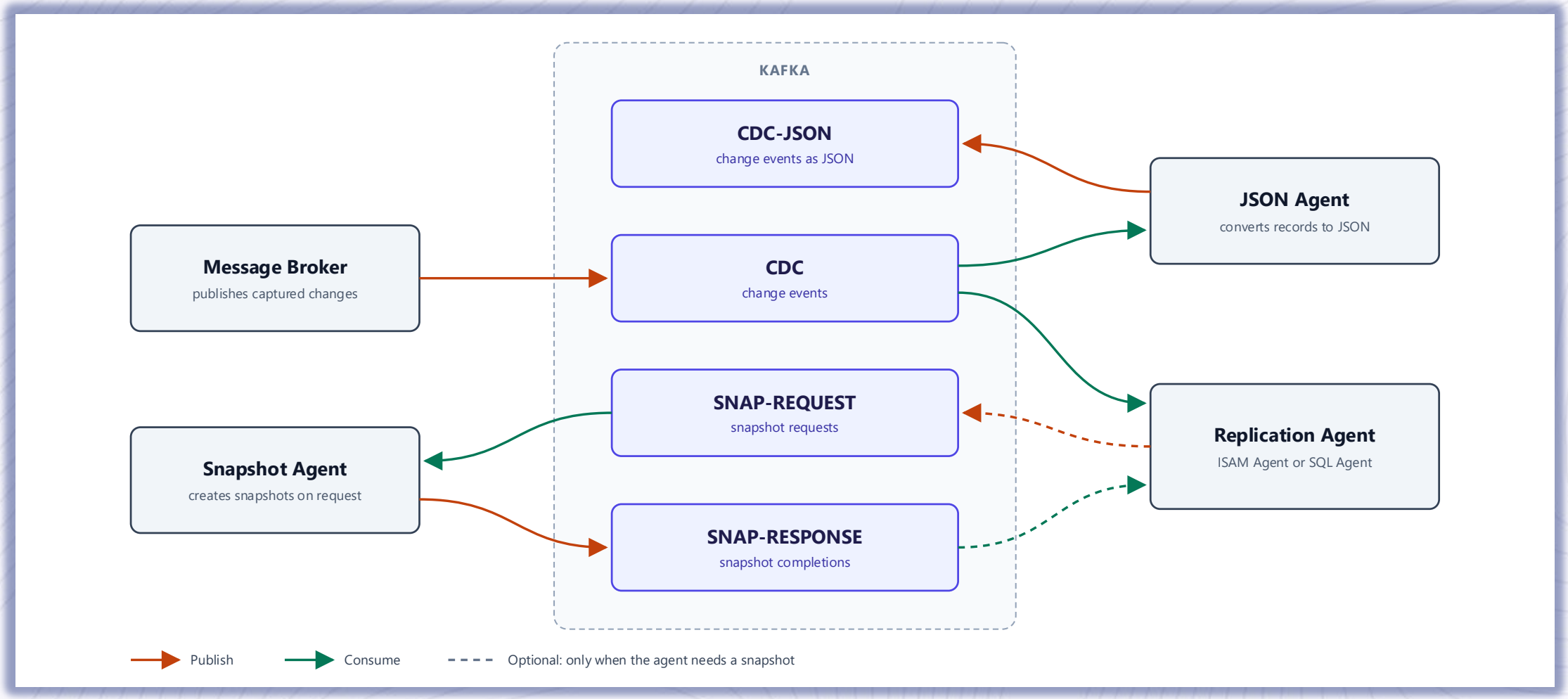
- Your DBL application and SDMS or RMS data
- SIP Components
  - I/O hooks
  - Snapshot Agent
  - Message Broker
- Minimal change to your code, none with global I/O hooks
- It’s all about Change Data Capture (CDC) and subsequent processing



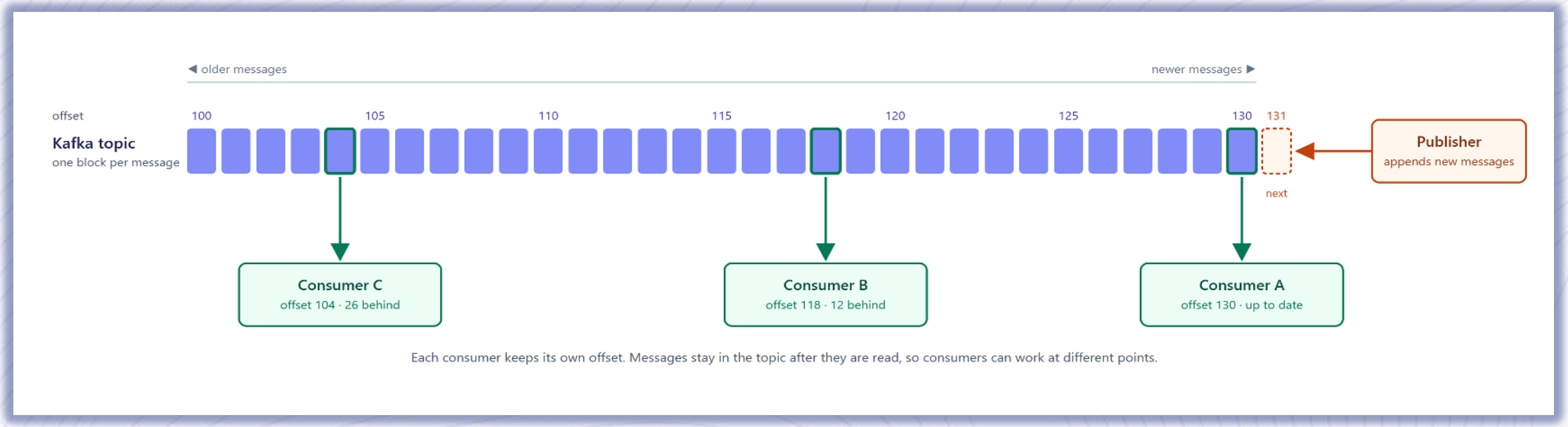
The Messaging Backbone

- High-throughput, distributed & resilient event streaming platform
  - Designed for real-time data pipelines and messaging at scale
  - Durable & ordered logs of messages organized in “topics”
- Like and different from message queues
  - Durable log, not a queue that empties
  - Multiple independent consumers can process the same messages
  - Consumers can rewind and replay messages
- Kafka connects components without coupling them
  - Producers & consumers never talk directly
  - Slow or stopped consumers never hold up the application or other consumers

How We Use Kafka

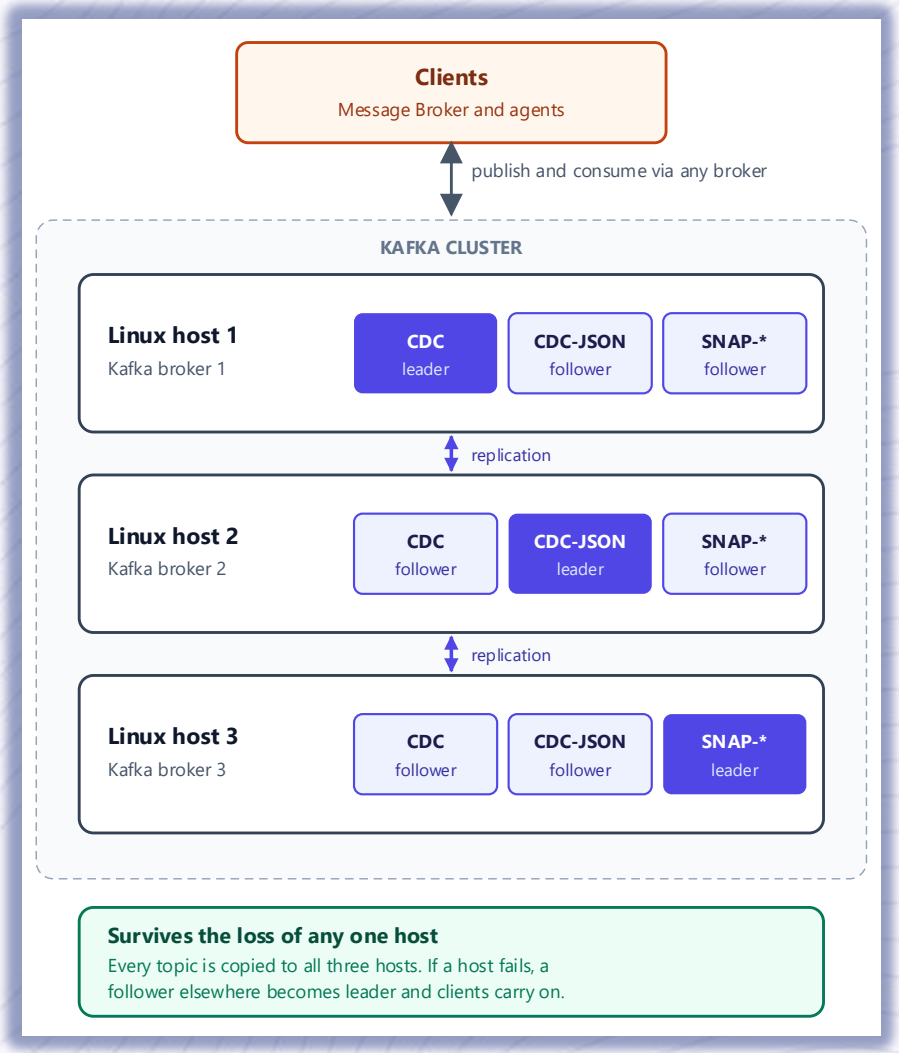


Topic Offsets



- Publishers
  - Add messages at the “log end offset” (highest offset +1)
- Consumers
  - Commit messages after processing (increments “committed offset”)
  - Consume next message using committed offset
  - Broker maintains committed offset for each consumer (group)

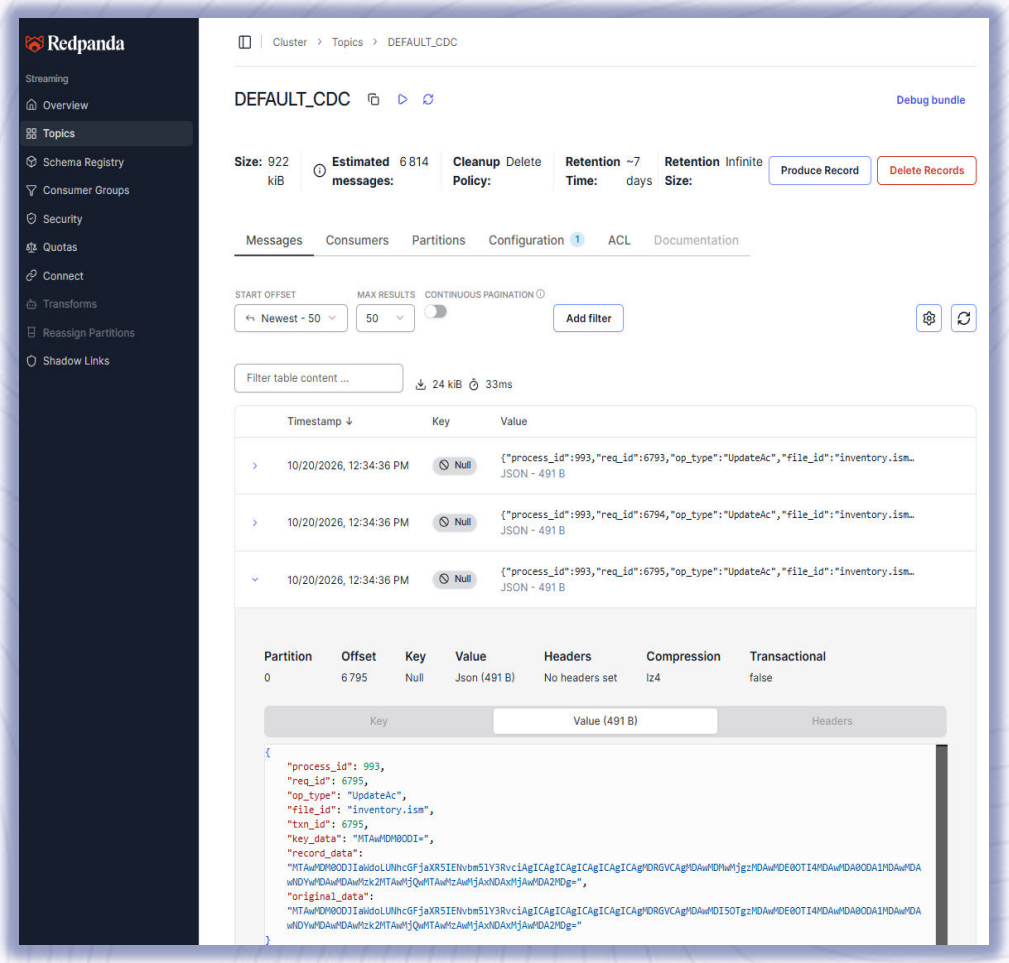
Deploying Kafka



- Kafka instances (brokers) run on Linux
- For production
  - Three or more brokers for fault-tolerance
  - Workload distributed across brokers
  - Three-node cluster survives loss of one node
- Other workloads (DEV, QA, etc.)
  - One broker is acceptable
  - Local VM or container
  - Not great in WSL

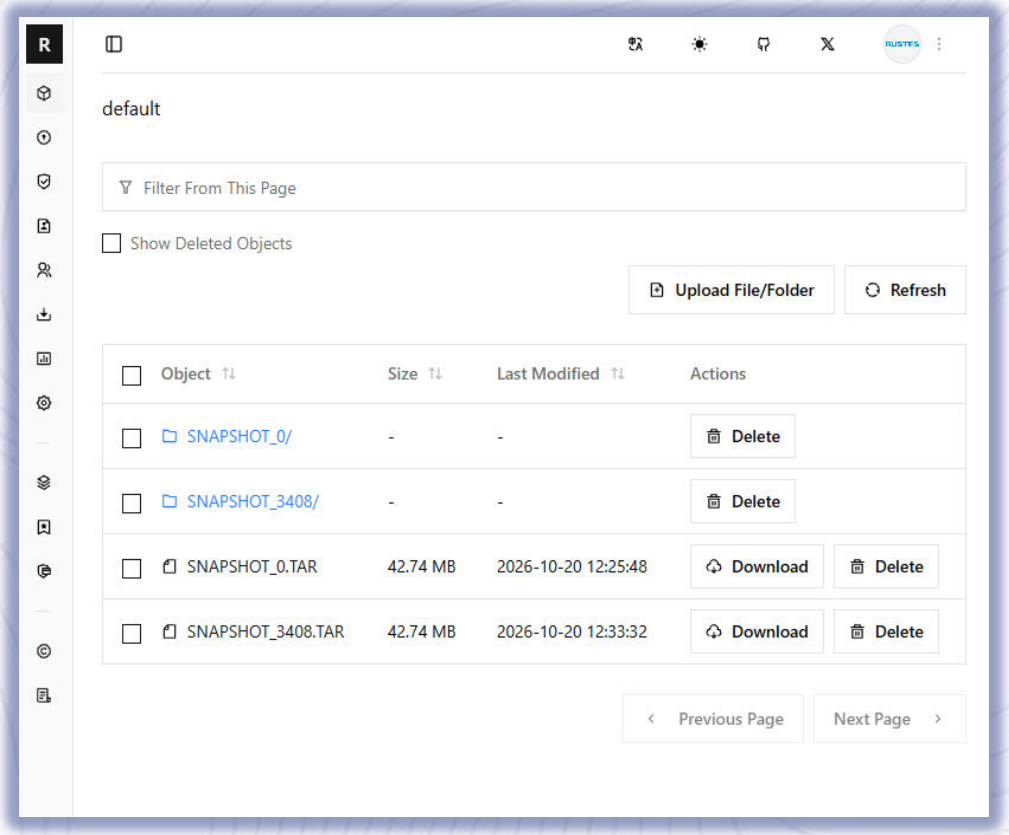
Redpanda

- Drop-in replacement for Kafka
  - Same protocol, API, & client libraries
  - Better performance, lower latency
  - Fewer servers
  - Easier to install, configure, & operate
- Community edition (free)
  - Fully functional for what SIP needs
  - Documentation & community support
- Enterprise edition (paid)
  - Additional features
  - 24x7 support with SLAs

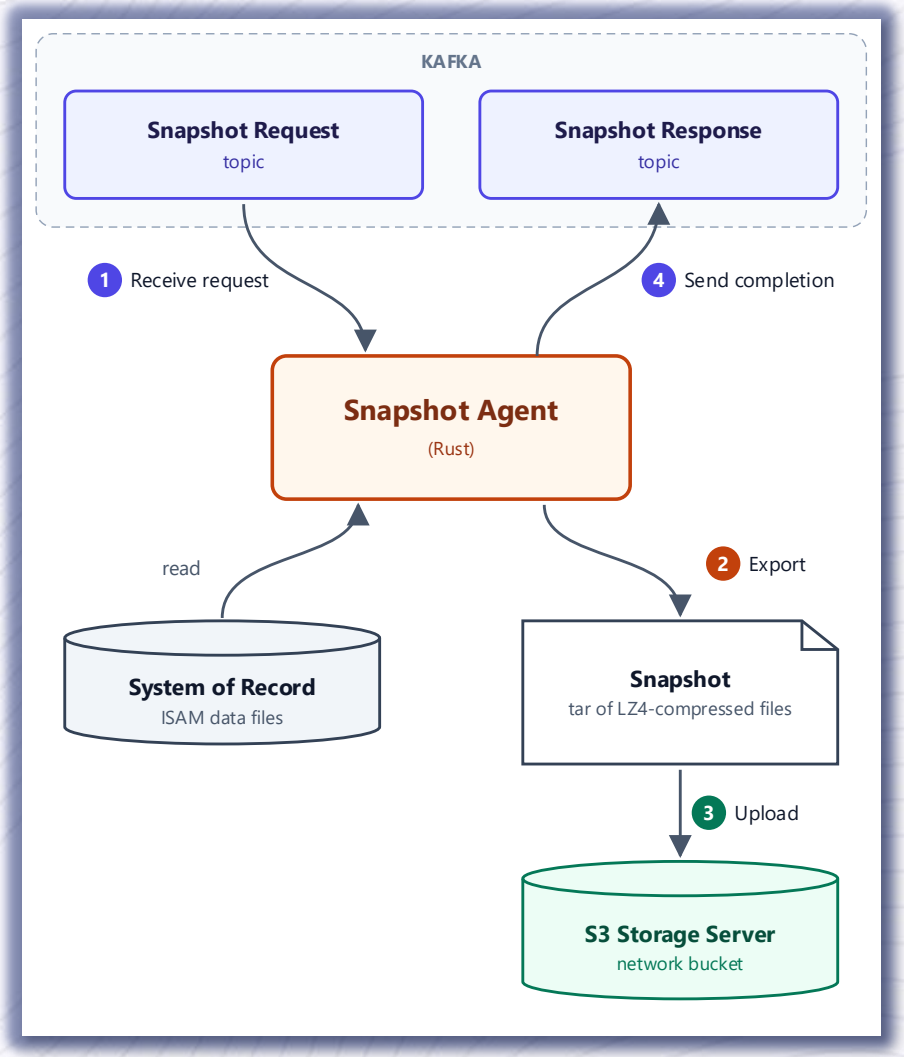

Redpanda
Streaming
Overview
Topics
Schema Registry
Consumer Groups
Security
Quotas
Connect
Transforms
Reassign Partitions
Shadow Links
Cluster › Topics › DEFAULT_CDC
DEFAULT_CDC
Debug bundle
Size: 922 kiB
Estimated messages: 6 814
Cleanup Policy: Delete
Retention Time: ~7 days
Retention Size: Infinite
Produce Record
Delete Records
Messages
Consumers
Partitions
Configuration 1
ACL
Documentation
START OFFSET
Newest - 50
MAX RESULTS
50
CONTINUOUS PAGINATION
Add filter
Filter table content ...
24 kiB
33ms
Timestamp
Key
Value
10/20/2026, 12:34:36 PM
Null
{"process_id":993,"req_id":6793,"op_type":"UpdateAc","file_id":"inventory.ism…
JSON - 491 B
10/20/2026, 12:34:36 PM
Null
{"process_id":993,"req_id":6794,"op_type":"UpdateAc","file_id":"inventory.ism…
JSON - 491 B
10/20/2026, 12:34:36 PM
Null
{"process_id":993,"req_id":6795,"op_type":"UpdateAc","file_id":"inventory.ism…
JSON - 491 B
Partition
Offset
Key
Value
Headers
Compression
Transactional
0
6 795
Null
Json (491 B)
No headers set
lz4
false
Key
Value (491 B)
Headers
{
"process_id": 993,
"req_id": 6795,
"op_type": "UpdateAc",
"file_id": "inventory.ism",
"txn_id": 6795,
"key_data": "MTAwMDM0ODI=",
"record_data":
"MTAwMDM0ODIiaWdoLUNhcGFjaXR5IENvbm51Y3RvciAgICAgICAgIC

Snapshots

- A full copy of the entire data set
  - Created by Snapshot Agent on system of record
  - Created over time while app is active
  - Not guaranteed consistent without catch-up
- ISAM Agent: Counted files
  - LZ4 compressed into a single TAR file
- SQL Agent: Parquet files
  - Columnar, compressed, schema-embedded file format optimized for fast analytical queries on large datasets
- Stored in a network S3 storage server
  - RustFS, MinIO, SeaweedFS, others



Snapshot Agent



- On demand, snapshot the complete data set and upload it to network storage
- Processing loop
  - Wait for a snapshot request
  - Note current CDC topic offset
  - Export data set, highly compressed
  - Upload snapshot to network storage
  - Send snapshot response message
- Exclusive access NOT required
- Snapshots are associated with a precise point in time in the CDC timeline

Snapshot Request & Response Messages

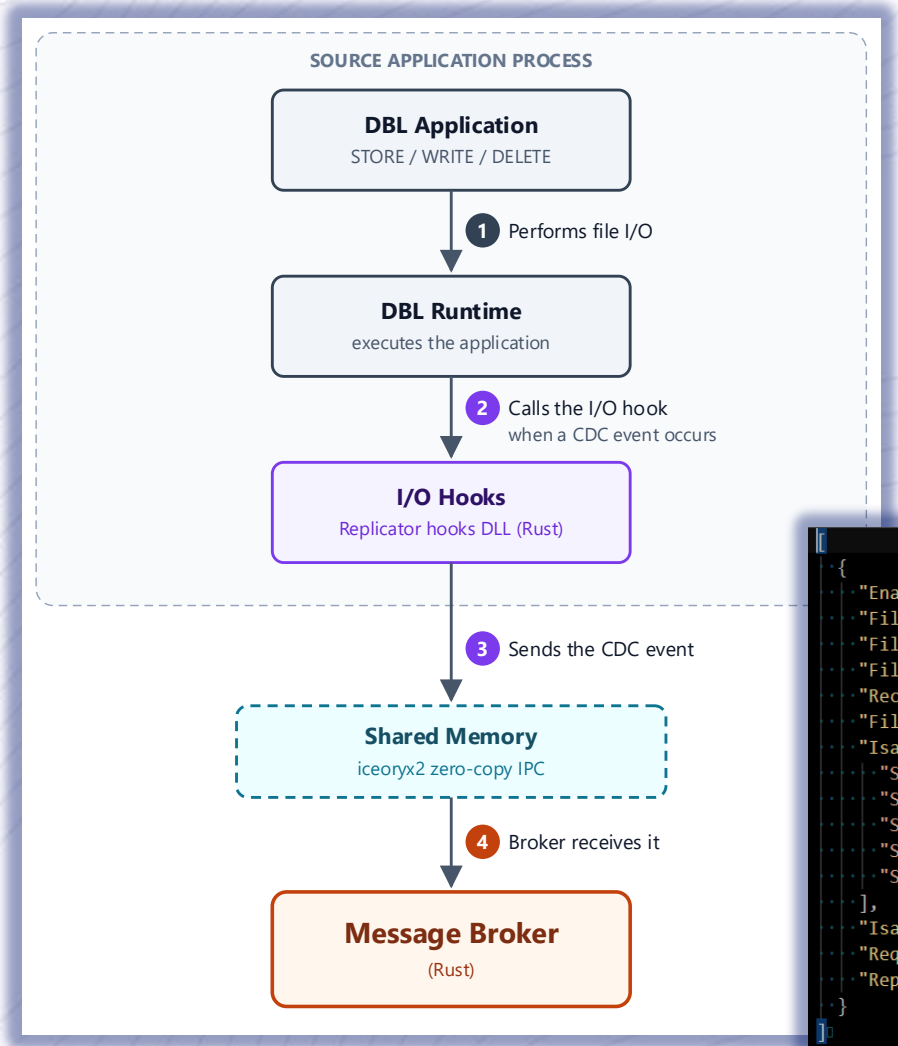
SYNERGY 2026 DEVPARTNER CONFERENCE

```
{
    "RequestId": "fb039130-d0f6-455a-a165-4af1dafbbd1b",
    "RequestTime": "2026-10-20T19:33:30.4021265Z",
    "Format": "Counted"
}
```

- Counted file snapshot
- Id properties match
- CDC offset identifies snapshot
- UTC timestamps

```
{
    "Version": 1,
    "Created": "2026-10-20T19:33:32.287319593Z",
    "Id": "fb039130-d0f6-455a-a165-4af1dafbbd1b",
    "Format": "Counted",
    "BlobPathCounted": "SNAPSHOT_3408.TAR",
    "BlobPathParquet": "",
    "MetaBlobPath": "",
    "Offsets": [
        {
            "Topic": "DEFAULT_CDC",
            "Partition": 0,
            "Offset": 3408
        }
    ]
}
```

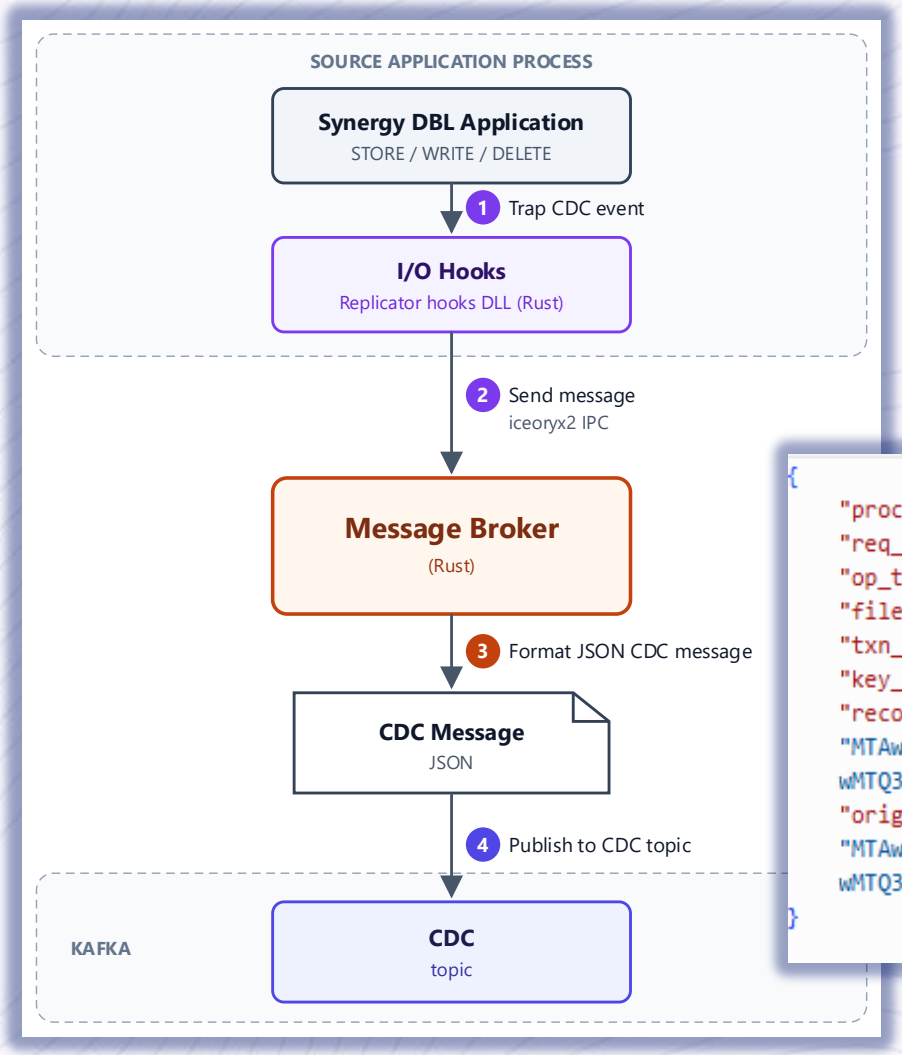
Change Data Capture



- I/O hooks process file system events and forward data to Message Broker
- Included files configured by metadata created by MetaDataTool

```
[
  {
    "Enabled": true,
    "FileName": "EMPLOYEE.ISM",
    "FileSpec": "SIP_DAT:EMPLOYEE.ISM",
    "FileType": "ISAM",
    "RecordLength": 300,
    "FileAttributes": "FIXED,PAGE=4096",
    "IsamKeys": [
      "START=1,LENGTH=8,NAME='EMP_ID',TYPE=ALPHA,ORDER=A,DENSITY=75,NODUPS,NOMODIFY,ASCEND",
      "START=49,LENGTH=15,NAME='EMP_DEPT',TYPE=ALPHA,ORDER=A,DENSITY=50,DUPS,ATEND,MODIFY,ASCEND",
      "START=29,LENGTH=20,NAME='EMP_LAST_NAME',TYPE=ALPHA,ORDER=A,DENSITY=50,DUPS,ATEND,MODIFY,ASCEND",
      "START=218,LENGTH=2,NAME='STATE_CODE',TYPE=ALPHA,ORDER=A,DENSITY=50,DUPS,ATEND,MODIFY,ASCEND",
      "START=220,LENGTH=5,NAME='ZIP_CODE',TYPE=ALPHA,ORDER=A,DENSITY=50,DUPS,ATEND,MODIFY,ASCEND"
    ],
    "IsamReplicationKeyNum": 0,
    "RequiresReplicationKey": false,
    "ReplicationKeyPosition": 0
  }
]
```

Message Broker



- Receives CDC data from I/O hook
- Creates CDC message (JSON with record data)
- Reliably publishes to CDC topic


```
{
    "process_id": 993,
    "req_id": 6793,
    "op_type": "UpdateAc",
    "file_id": "inventory.ism",
    "txn_id": 6793,
    "key_data": "MTAwMDM0ODA=",
    "record_data":
    "MTAwMDM0ODBTdGFuZGFyZCBQdW1wICAgICAgICAgICAgICAgICAgICAgICAgICAgMDVCWCAgMDAwMDI3OTY2MDAwMDE0MDc1MDAwMDAyMzc5MDAwMDA
    wMTQ3MDAwMDAwMzk0MTAwMDk0MTAwMzAwMTczNDcxMjAxNzA0Mjk=",
    "original_data":
    "MTAwMDM0ODBTdGFuZGFyZCBQdW1wICAgICAgICAgICAgICAgICAgICAgICAgICAgMDVCWCAgMDAwMDI3Njg5MDAwMDE0MDc1MDAwMDAyMzc5MDAwMDA
    wMTQ3MDAwMDAwMzk0MTAwMDk0MTAwMzAwMTczNDcxMjAxNzA0Mjk="
}
```

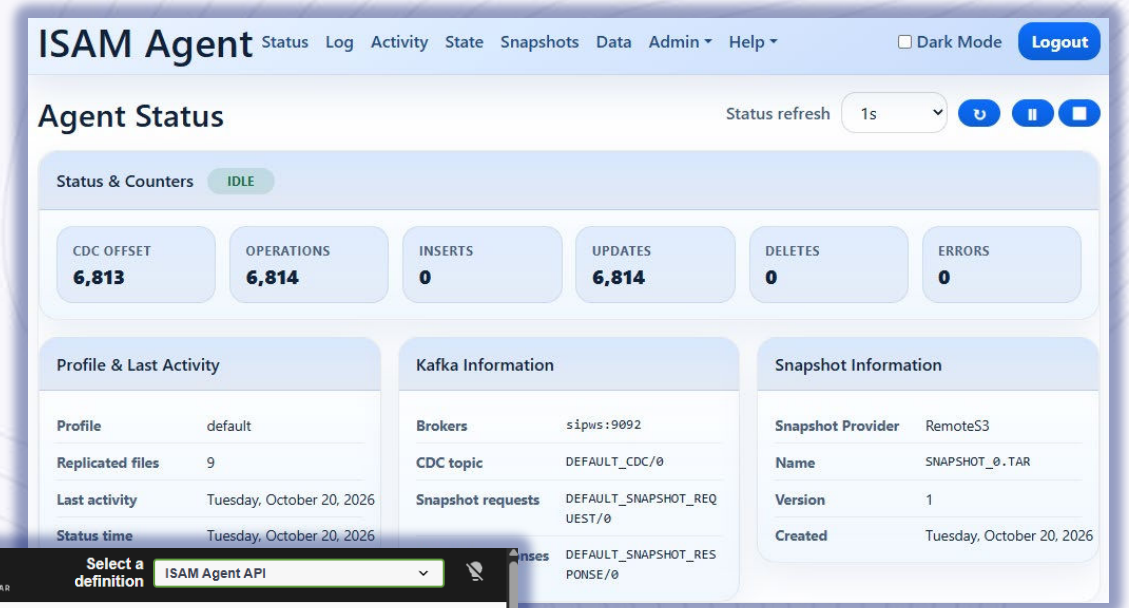
Target System Operations

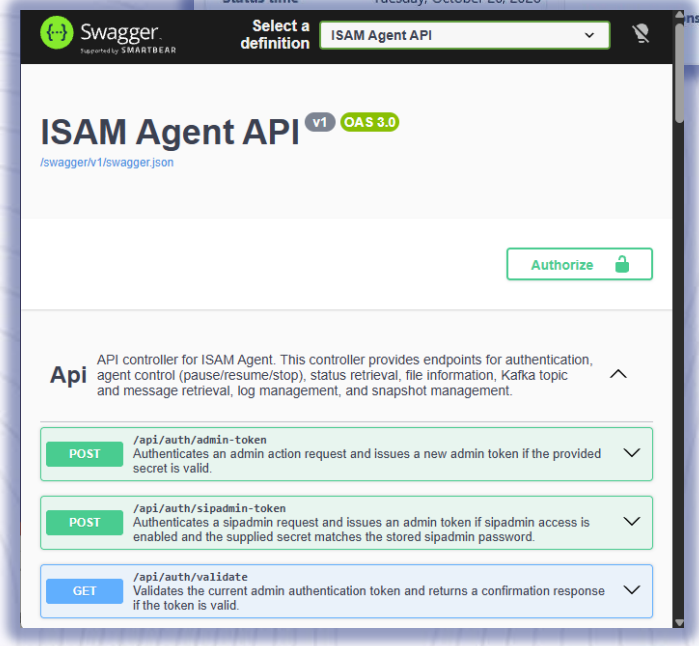
- First-time agent startup
  - No replica data present
  - Hydrate replica from latest snapshot
- Subsequent agent startup
  - Data consistency checks
  - Restore latest snapshot if necessary
- Catch-up if necessary
  - Last CDC offset < current CDC offset
- Agent main processing loop
  - Wait for CDC message
  - Process the message



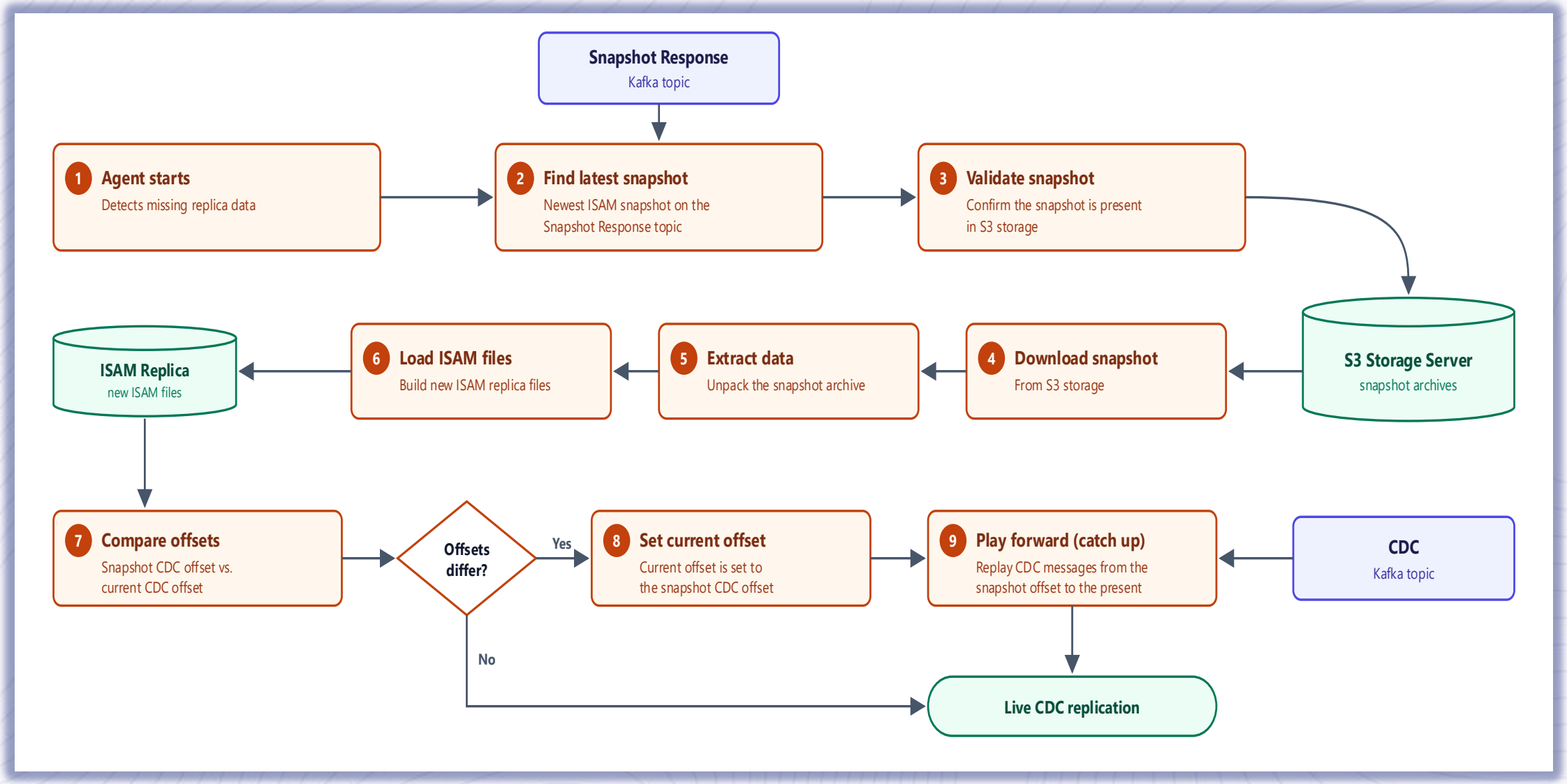
ISAM Agent

- Creates and maintains a data replica in local SDMS files
  - Offload read-only workloads
  - DR hot standby
- Driven from ISAM metadata seen earlier
- ASP.NET Application
  - Browser-based management UI
  - Remote automation via REST API
  - Supported on Windows or Linux

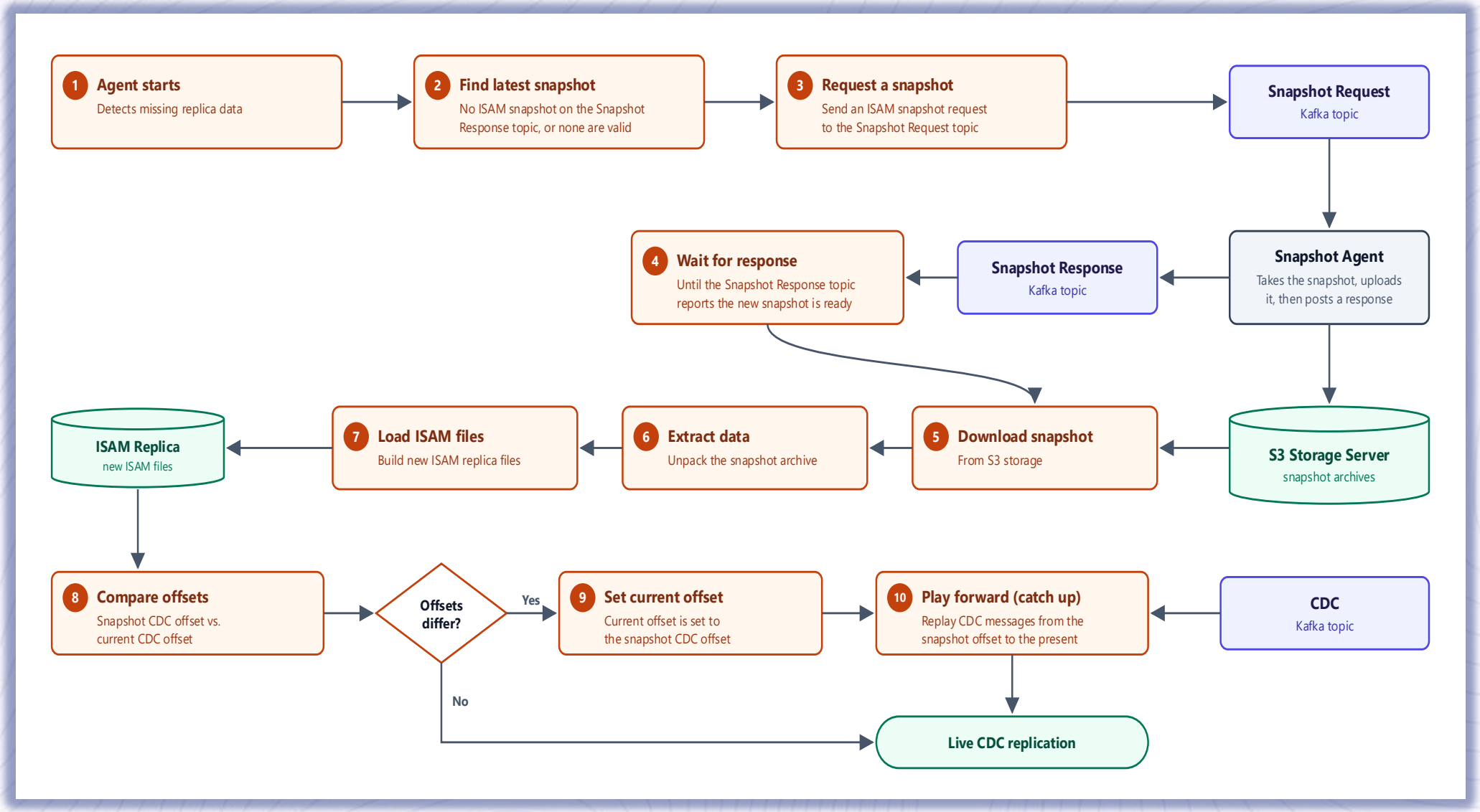




ISAM Agent: Initial Startup With Snapshot

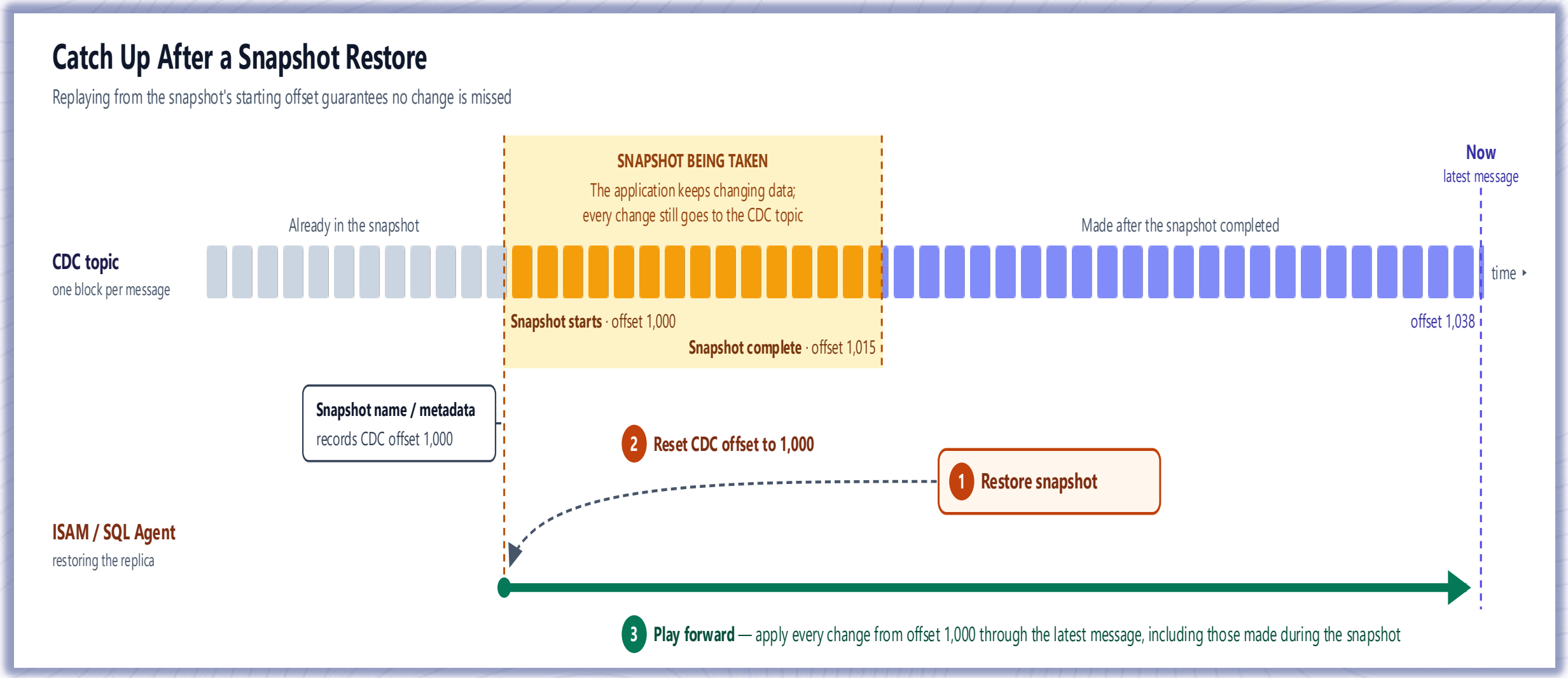


ISAM Agent: Initial Startup Without Snapshot



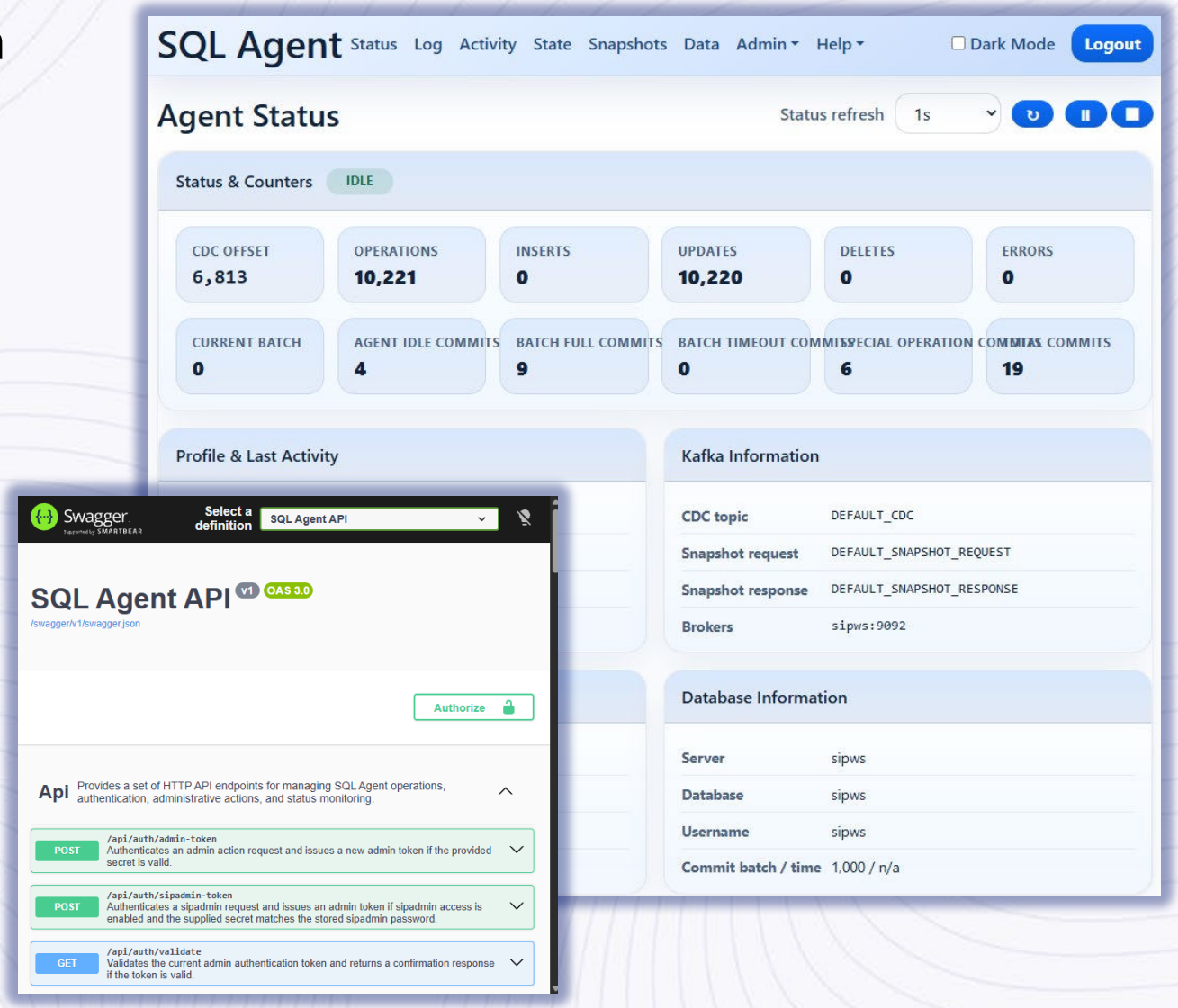
Catch Up After Snapshot Restore

SYNERGY 2026 DEVPARTNER CONFERENCE

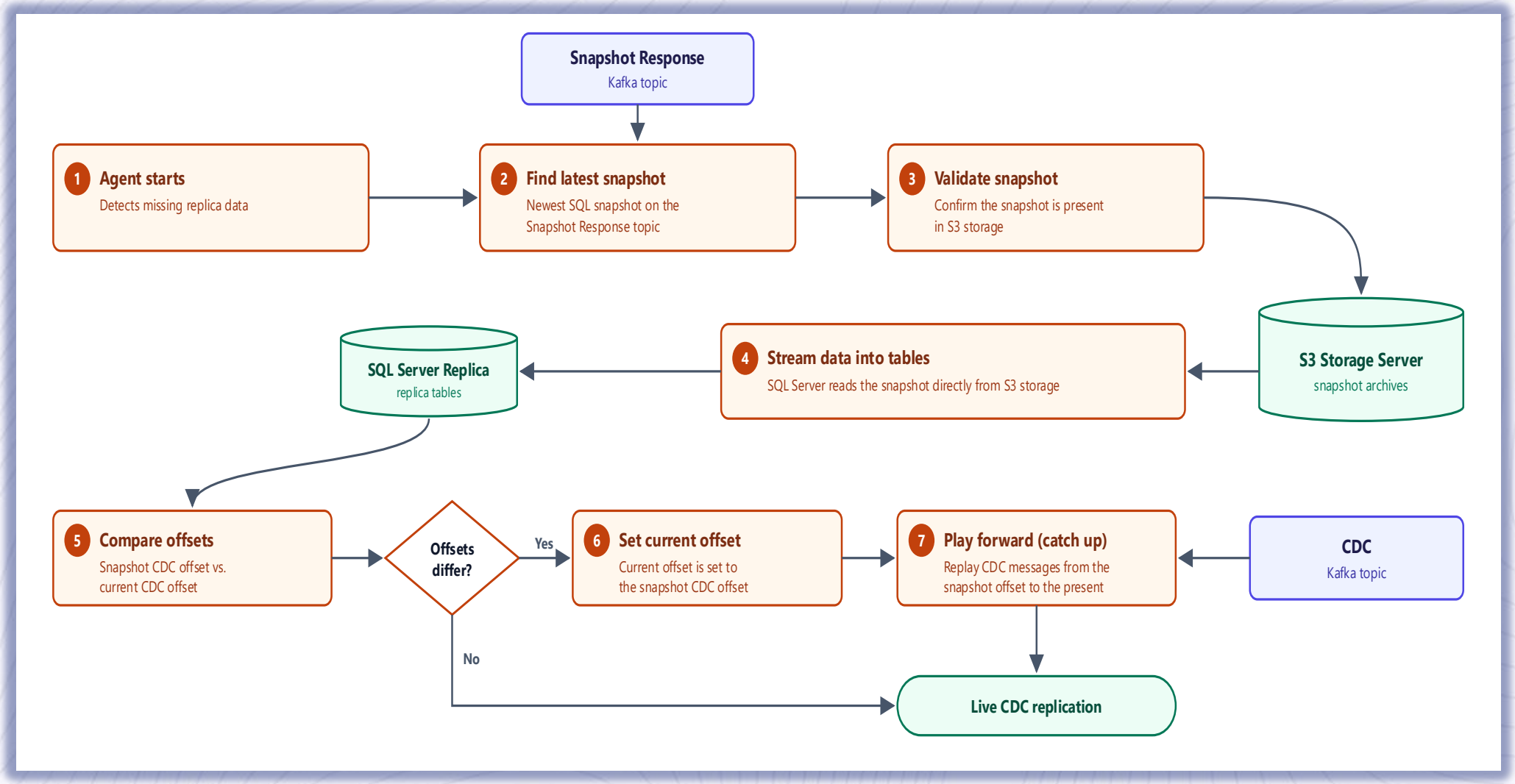


SQL Agent

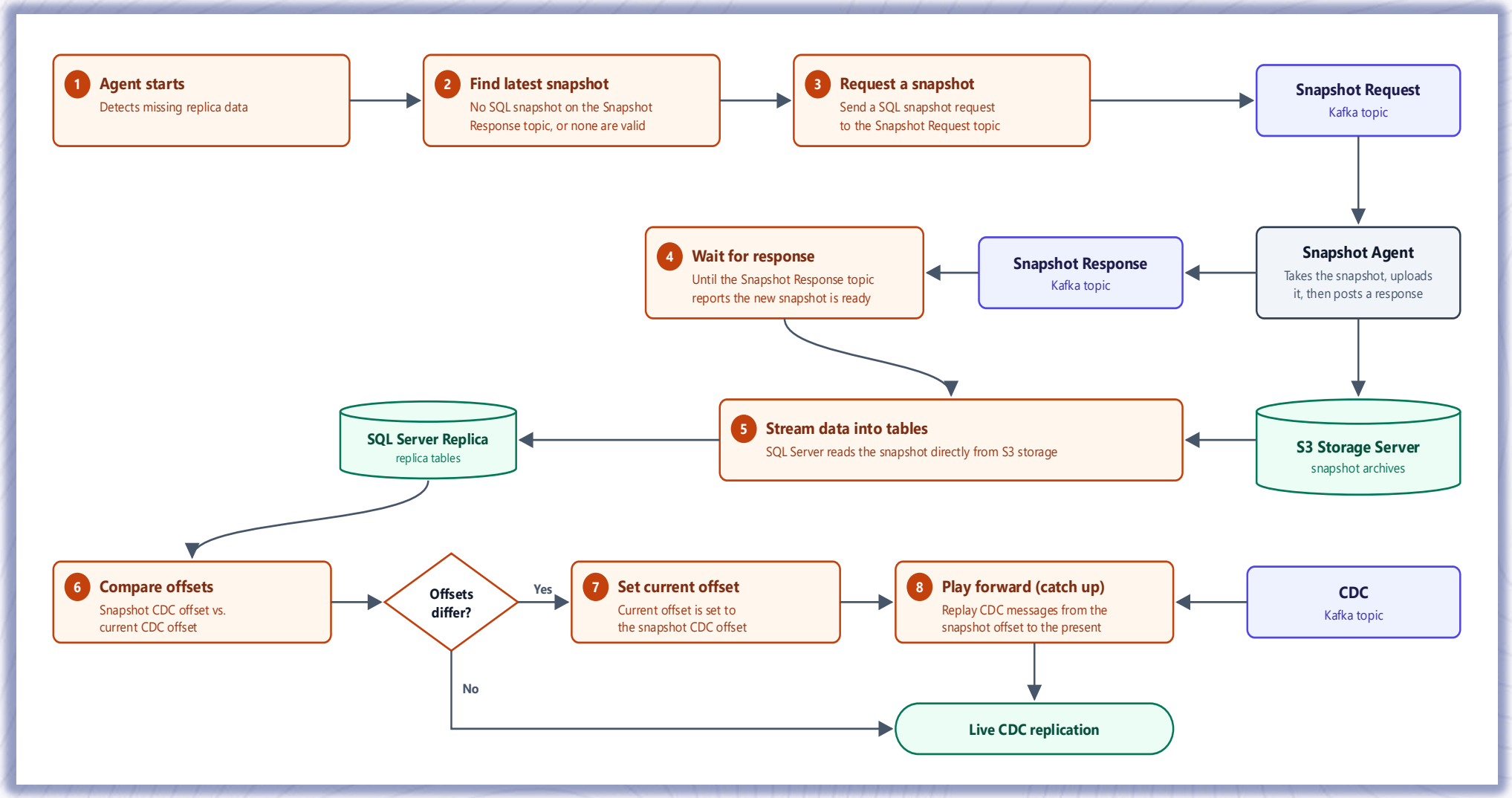
- Creates and maintains a data replica in Microsoft SQL Server
- Driven from SQL metadata generated from Synergy repository
- Requirements
  - SQL Server 2022 with PolyBase
  - SQL Server 2025
- ASP.NET Application
  - Browser-based management UI
  - Remote automation via REST API
  - Supported on Windows or Linux



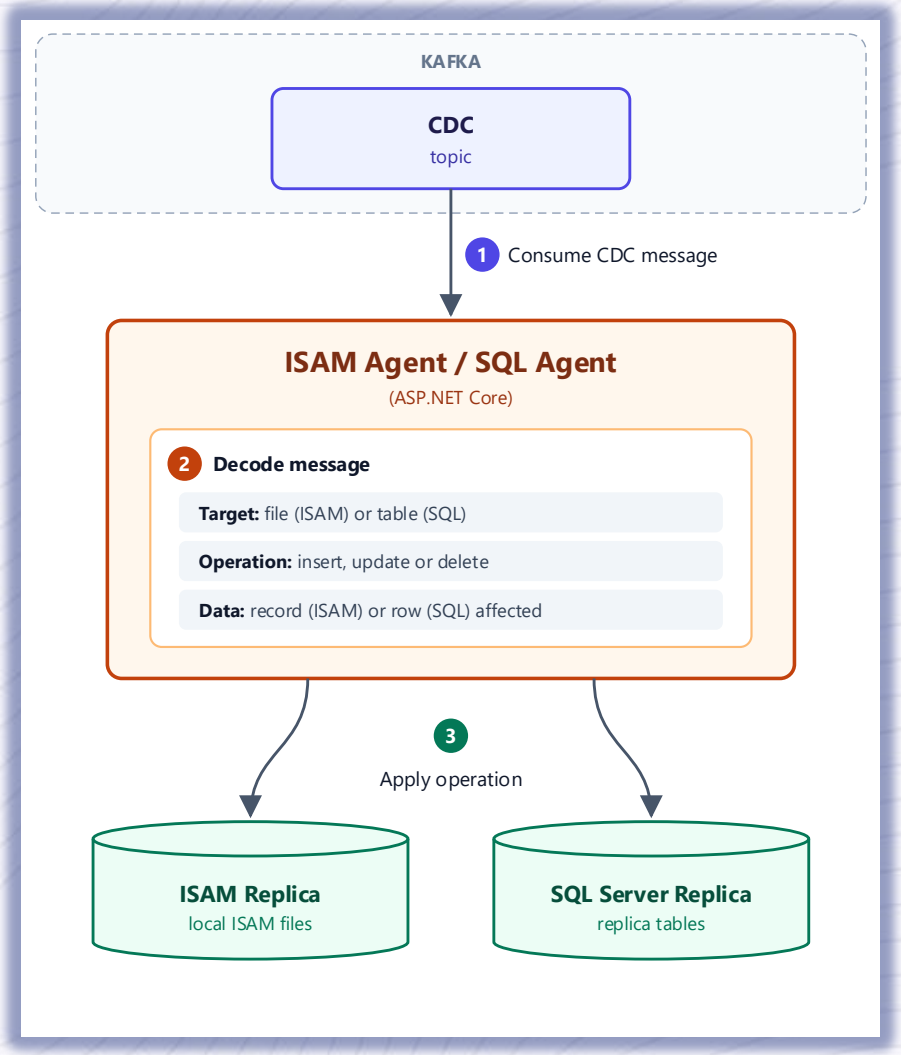
SQL Agent: Initial Startup With Snapshot



SQL Agent: Initial Startup Without Snapshot



ISAM & SQL Agents: CDC Processing



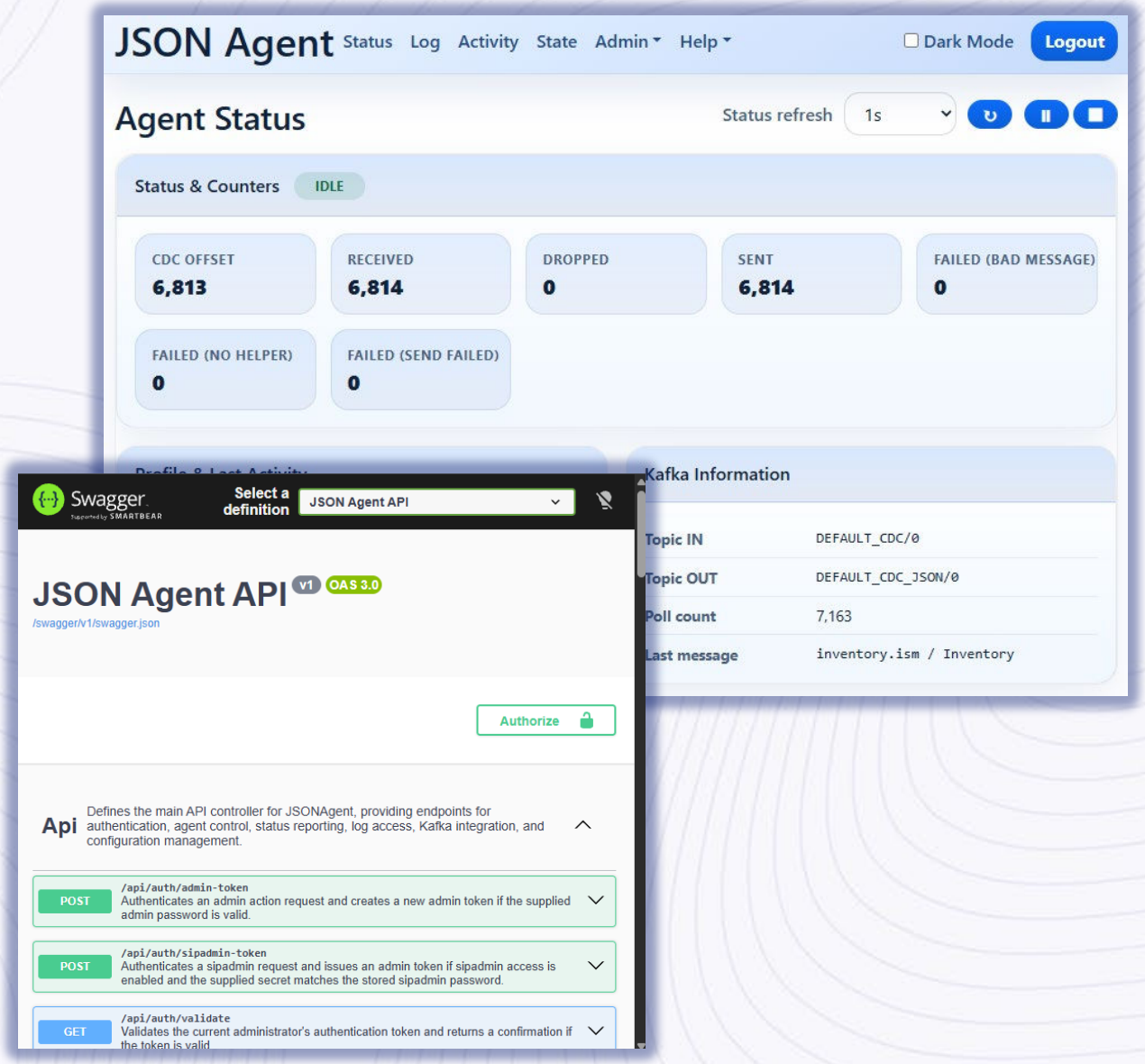
- Main processing loop
  - Consume CDC message
  - Determine necessary action
  - Apply action to local replica
  - Commit CDC message
- Not critical components, unless the consumers of the replicas are critical
  - If shut down, picks up where it left off

JSON Agent

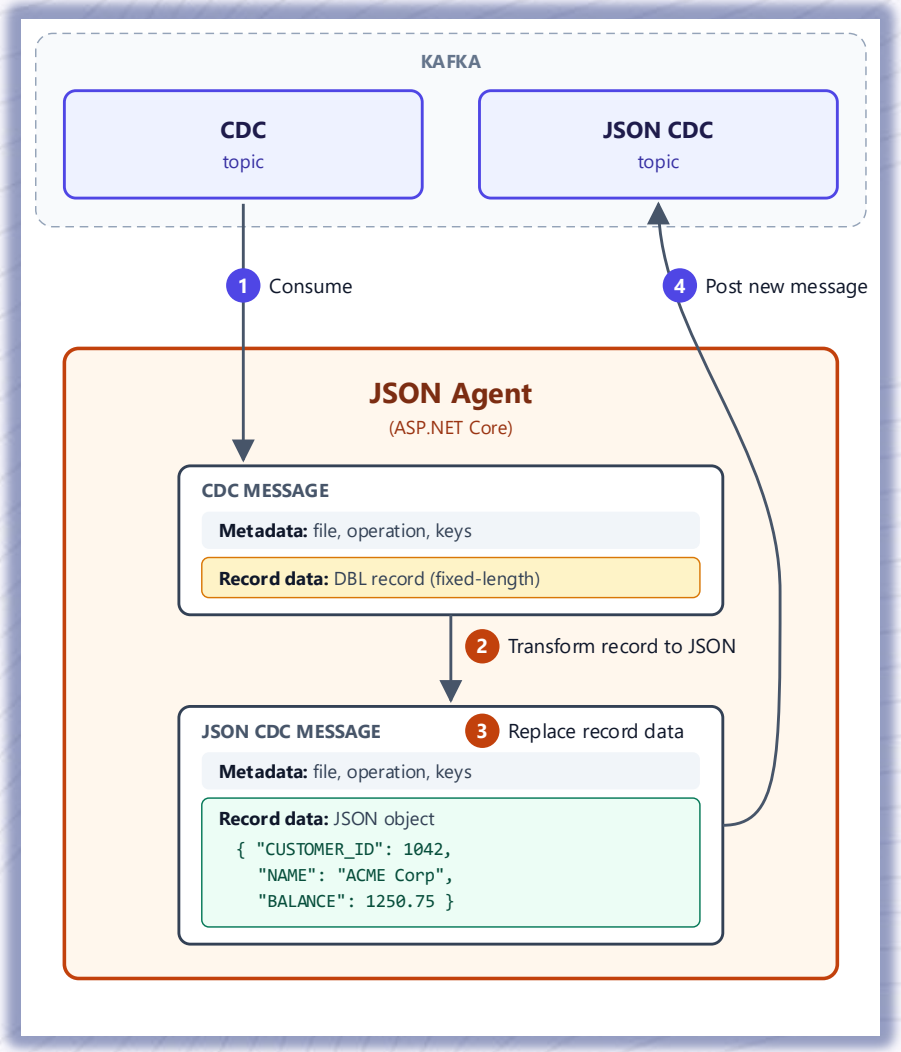
- Reformats CDC messages as JSON
- Posts to another topic

- Requires custom record helper classes in a “plug-in” assembly

- ASP.NET application
  - Browser-based management UI
  - Remote automation via REST API
  - Supported on Windows or Linux



JSON Agent: CDC Processing



- Consume CDC message
- Decode base-64 record(s)
- Serialize record(s) to JSON object(s)
- Replace CDC message records with JSON objects
- Publish message to JSON CDC topic
- Commit message original CDC message

Kafka Log Retention & Message Lifetime

- Broker defaults, overridable per topic
  - Time-based: hours, minutes, or milliseconds (default 168 hours)
  - Size-based: bytes (default -1 = unlimited)
- Hitting either limit triggers deletion
  - Checked periodically (default 5 min)
- Deletion by log segment; only closed segments
  - Active segment never removed
  - Some data can outlive the limit
- NEEDS CAREFUL CONSIDERATION

What Must **NEVER** Happen

- Oldest CDC offset > latest snapshot CDC offset
  - No longer possible to catch up after restoring a snapshot
- Latest snapshot response offset (for each snapshot type) removed
  - Agents can’t find snapshots if needed
- If it ever happens
  - IMMEDIATELY take new snapshot(s)
  - No planned agent outages, fingers crossed for no unplanned ones!
- AND … outage of **Message Broker**, **Kafka**, or **comms between them** while **application is active**
  - Out-of-sync replicas and an invalid CDC record
  - Rehydrate all agents from a NEW snapshot

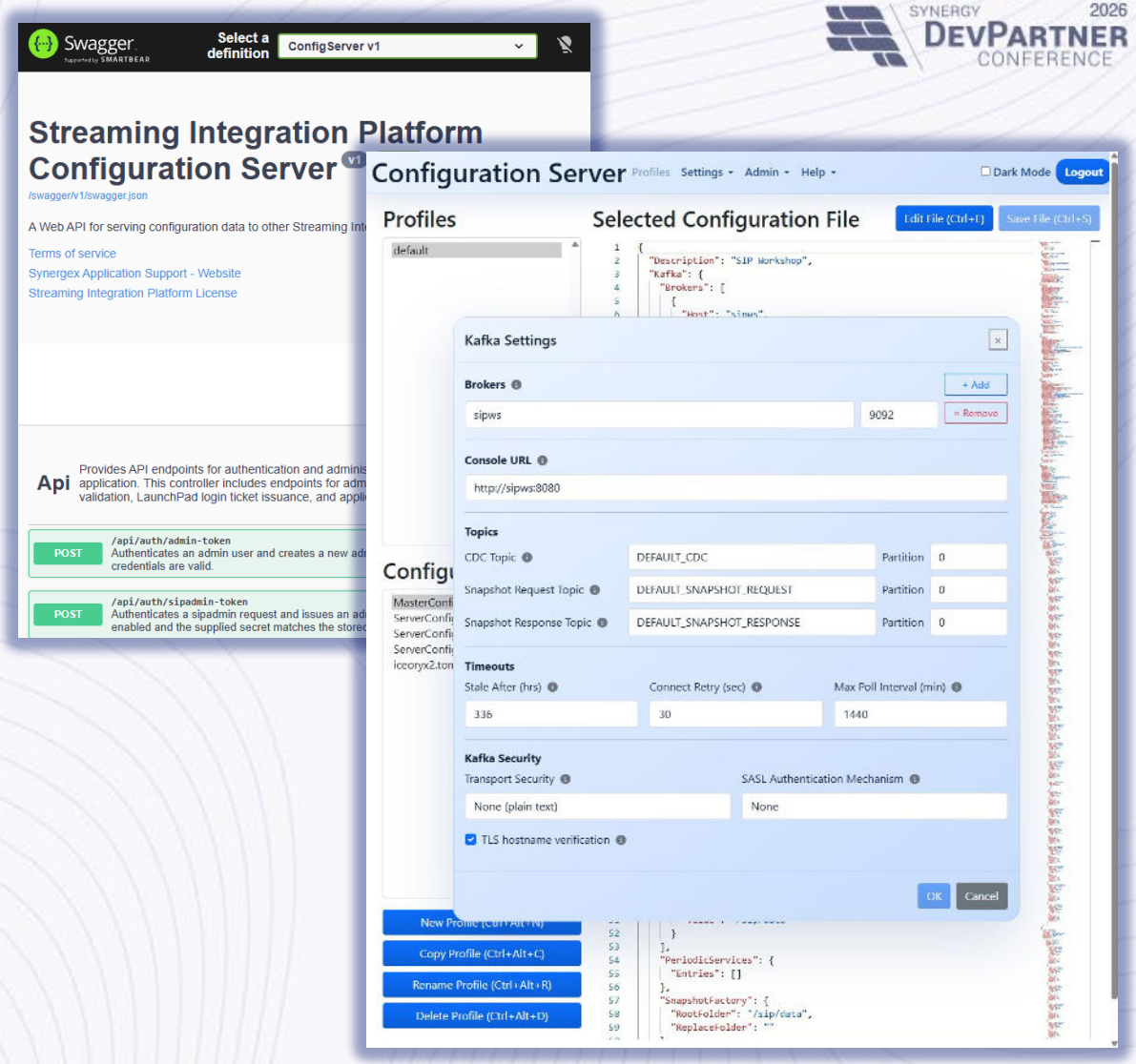
Agent Logging

- Log to console
  - All components log to the console
  - Recorded in system logs when running as a service
- Log to file
  - Supported by ISAM, SQL, & JSON agents
  - Mostly the same content as console messages
- Email notifications
  - Some in ISAM, SQL & JSON Agents, & Config Server
- Windows Event Log
  - Some support in SQL Agent, needs extending

Configuration Server

- Centralized management & delivery of configuration files & TLS certificates
  - Message Broker (not OpenVMS)
  - Snapshot Agent (not OpenVMS)
  - ISAM Agent
  - SQL Agent
  - JSON Agent
  - Configuration Server (TLS)

- Without Configuration Server, manually create, maintain, distribute, & synchronize on multiple systems.



Implementing SIP in Your Environment

- All deployment scenarios require ISAM metadata
  - From SDMS files or FDL files by MetaDataTool
- ISAM Agent
  - No additional requirements
- SQL Agent
  - SQL Helper Classes: CodeGen from repository
  - SQL metadata: from repository by MetaDataTool
- JSON Agent
  - Record Helper Classes: CodeGen from repository

SIP Developers Kit

- Package with everything needed to implement SIP in an application
  - Visual Studio solution with sample integration projects
  - CodeGen templates for SQL and record helpers
  - Binaries for I/O hooks, supporting code, & dependencies
  - Source for I/O hooks & supporting code
  - Integration guide

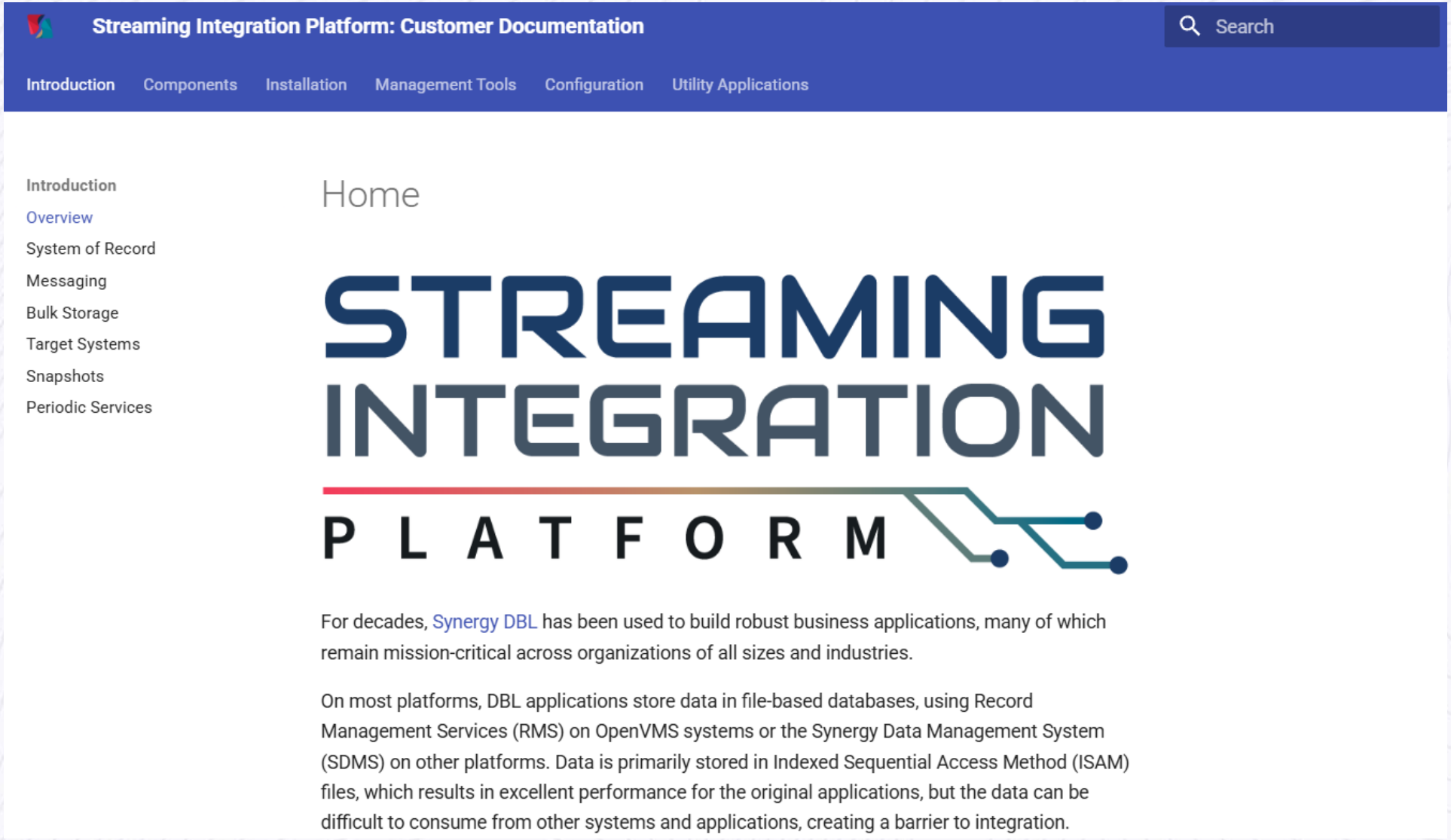
- Currently being built, available soon

SYNERGY 2026 DEVPARTNER CONFERENCE

Demonstration

Streaming Integration Platform

https://sip.docs.synergex.com



The Afternoon's Workshop

Start out with…

- A working application
  - Browser front end
  - Harmony Core API
  - DBL back end and data
- And pre-installed tooling
  - Microsoft SQL Server
  - Redpanda & Redpanda Console
  - RustFS (S3 storage server)
  - DbGate (database client)
  - SMTP4DEV (email server & UI)

You will…

- Install Configuration Server & configure environment
- Create Kafka topics
- Create ISAM & SQL metadata
- Install SIP base environment & create initial snapshots
- Install ISAM Agent & replicate to ISAM
- Install SQL Agent & replicate to SQL
- Install JSON Agent & transpose CDC messages
- Alter data and see replication in action
- And more…



Who has the first question?

Streaming Integration Platform