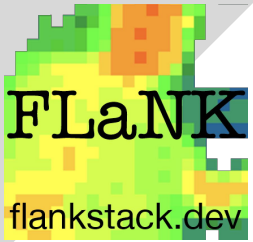
CLOUDERA



Talk 1: Flink SQL For Continuous SQL/ETL/Apps
Talk 2: Apache NiFi DevOps

Timothy Spann - Principal DataFlow Field Engineer

25-March-2021

https://www.meetup.com/futureofdata-princeton/

@PaasDev

Upcoming Events

- 7/April/2021 - CDF - Ask the Experts
- 15/April/2021 - Real-time Streaming Pipelines with FLaNK DataConLA
- 21/April/2021 - Emerging Tech Day
- 22/April/2021- Demo Jam NiFi
- 27/April/2021 - Developer Week Europe
- 06/May/2021 - Continuous SQL with SQL Stream Builder

- https://www.meetup.com/pro/futureofdata/
- https://www.linkedin.com/pulse/2021-schedule-tim-spann/
- https://www.meetup.com/futureofdata-newyork

CLOUDERA

Welcome to Future of Data - Virtual - 25/March/2021

FUTURE OF DATA

https://www.meetup.com/futureofdata-princeton/
https://www.meetup.com/futureofdata-newyork/
https://www.meetup.com/futureofdata-philadelphia/

From Big Data to AI to Streaming to Containers to Cloud to Analytics to Cloud Storage to Fast Data to Machine Learning to Microservices to ...

@PaasDev

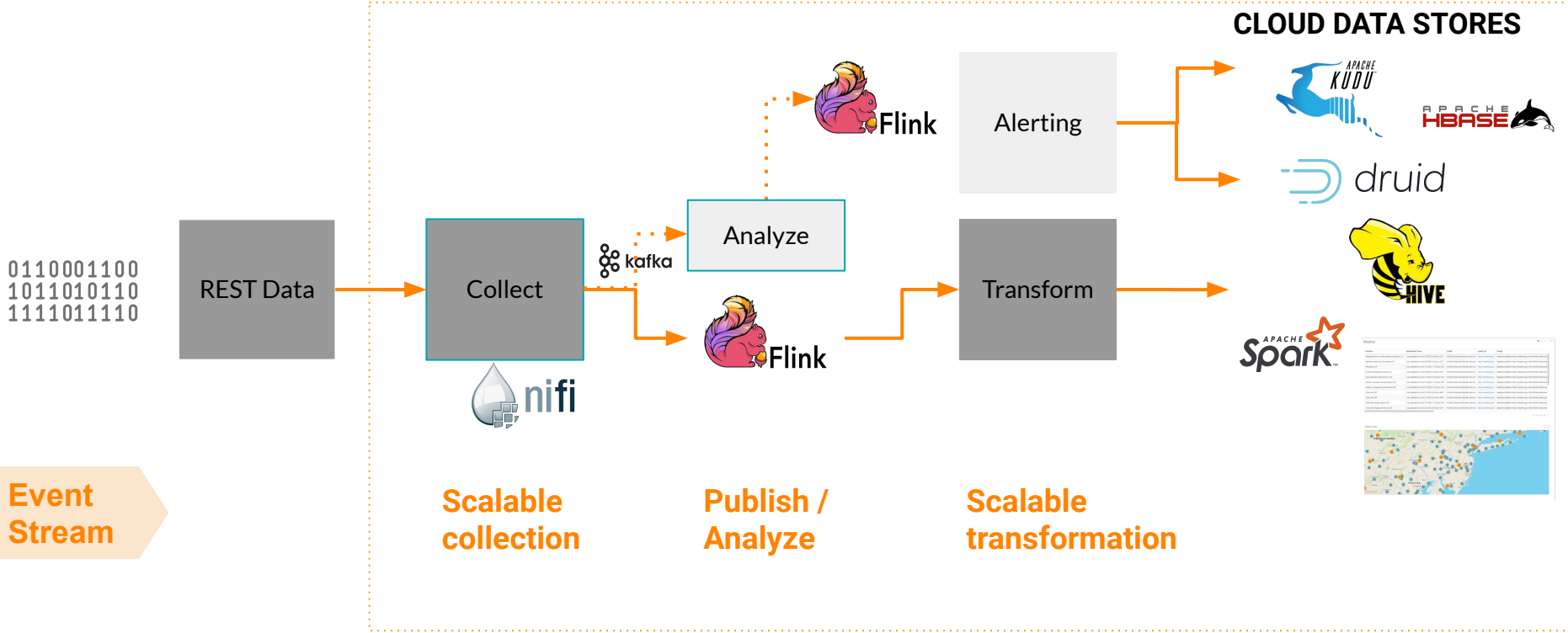

FLaNK
flankstack.dev


Tim
SPANN
https://github.com/tspannhw
https://www.datainmotion.dev/

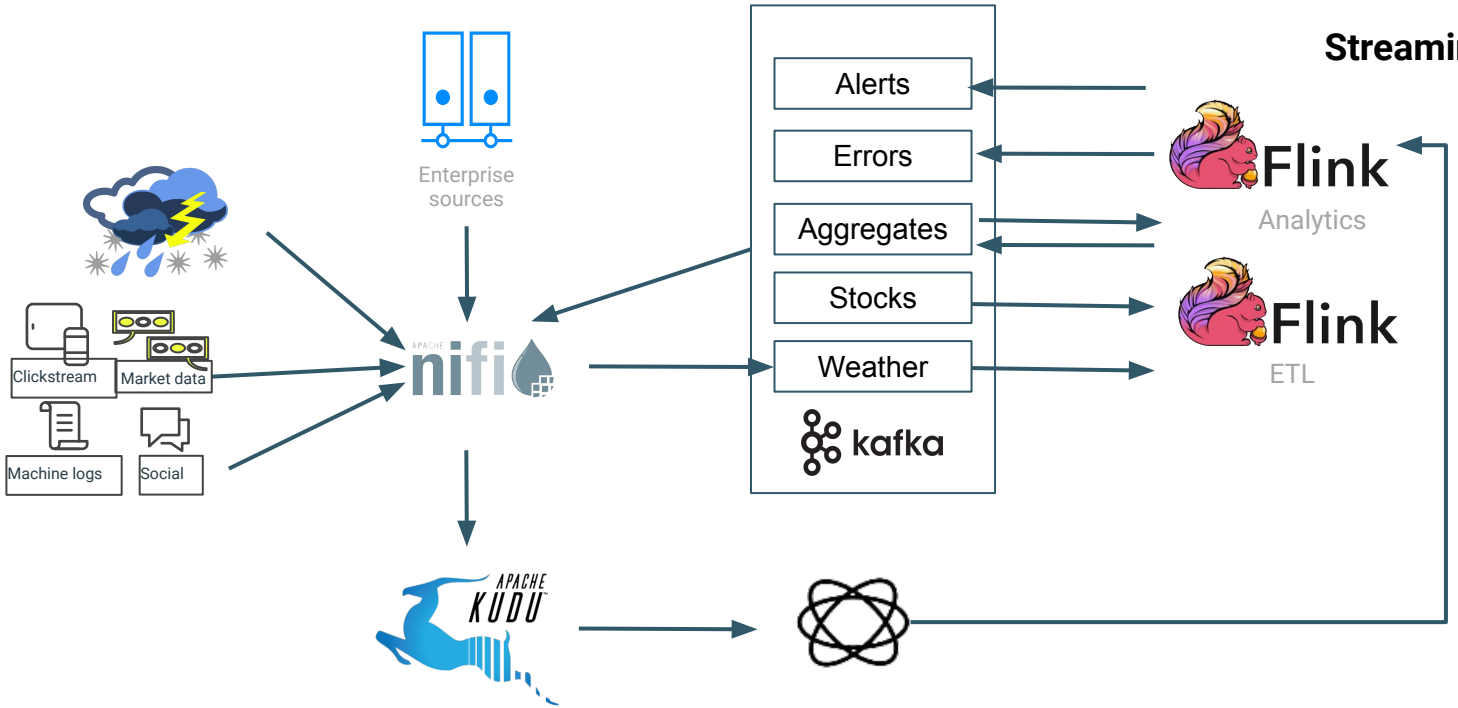

FLANK
NIFI - KAFKA - FLINK

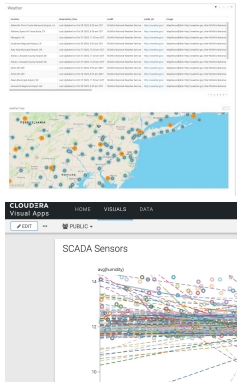
Simplifying the User Experience

Demo Scenario

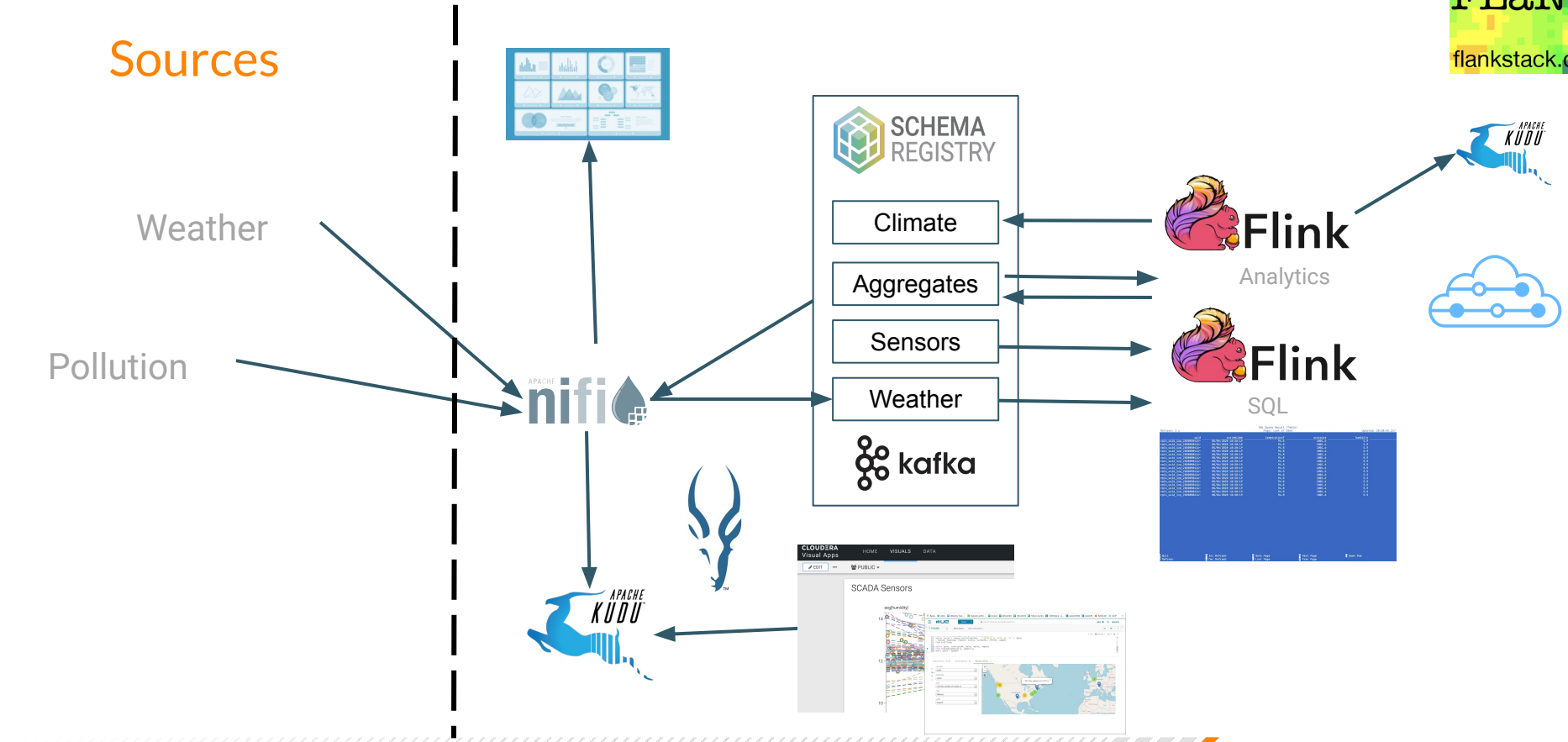


End to End Demo Pipeline

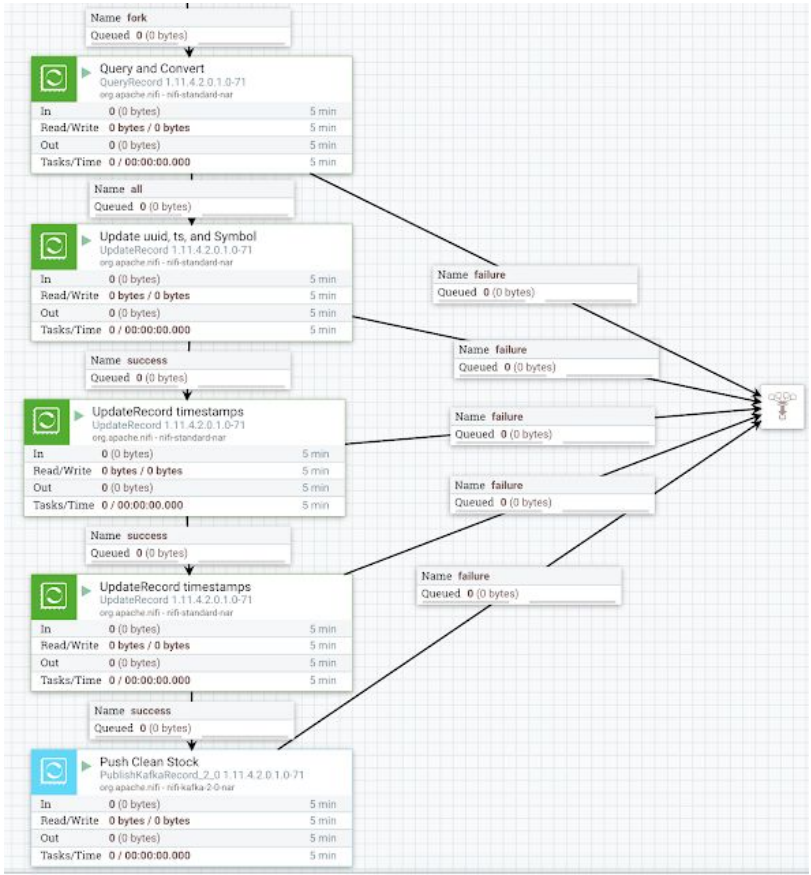


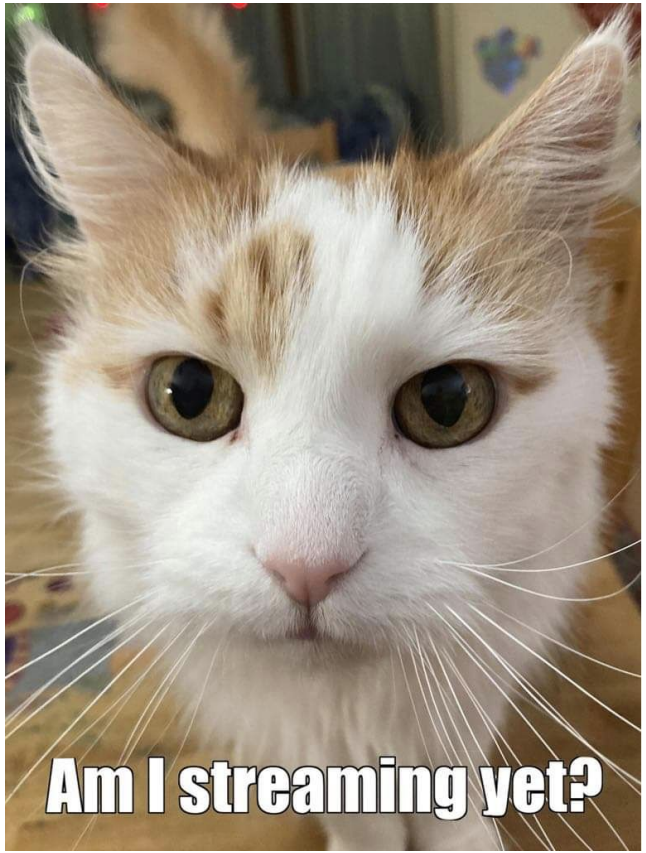


Weather Streaming Pipeline

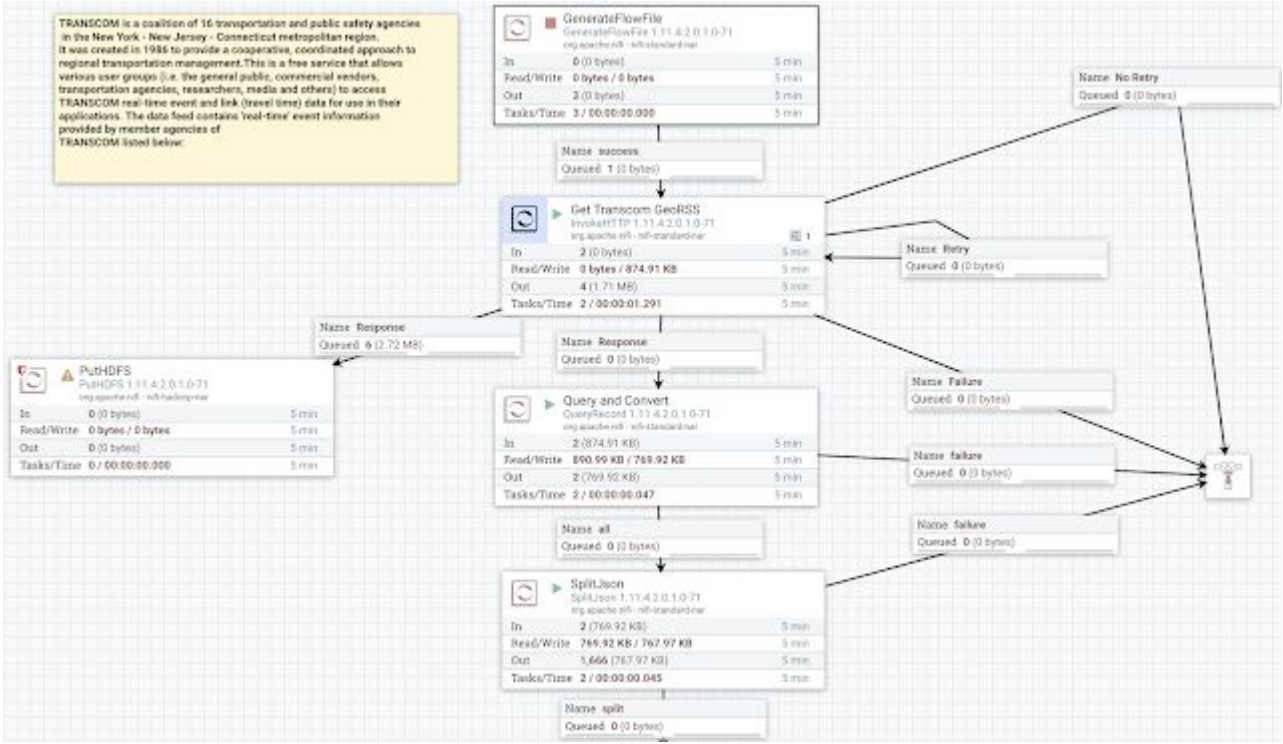


https://github.com/tspannhw/SmartStocks





https://www.datainmotion.dev/2021/01/flank-real-time-transit-information-for.html



CLOUDERA

Flink SQL Examples

```
INSERT INTO weathernj
SELECT CAST(`location` as STRING) `location`,
CAST(station_id as STRING) station_id, latitude,     longitude,
CAST(observation_time as STRING) observation_time,
CAST(weather as STRING) weather,
CAST(temperature_string as STRING) temperature_string,
    temp_f,     temp_c,     relative_humidity, CAST(wind_string  as STRING) wind_string,
CAST(wind_dir as STRING) wind_dir,
    wind_degrees,     wind_mph,     wind_kt,      pressure_in,
CAST(dewpoint_string as STRING) dewpoint_string, dewpoint_f,     dewpoint_c
FROM weather
WHERE
   `location` is not null and `location` <> 'null' and trim(`location`) <> '' and `location` like '%NJ';
```

Flink SQL Examples

```
INSERT INTO global_sensor_events

SELECT scada.uuid, scada.systemtime ,
scada.temperaturef ,
scada.pressure , scada.humidity ,
scada.lux , scada.proximity ,
scada.oxidising ,
scada.reducing , scada.nh3 ,
scada.gasko,energy.`current`,
energy.voltage ,energy.`power` ,
Energy.`total`,energy.fanstatus

FROM energy, scada

WHERE
   scada.systemtime >= energy.systemtime;
```


FLaNK
flankstack.dev

Apache NiFi DevOps

https://nifi.apache.org/docs/nifi-docs/html/toolkit-guide.html#nifi_CLI

Apache NiFi DevOps

FLaNK
flankstack.dev

Make sure you check out Apache NiFi 1.13.2. There are some great new features.

https://www.datainmotion.dev/2021/02/new-features-of-apache-nifi-1130.html

Today we will discuss some DevOps around NiFi with NiFi CLI, REST and NiPyAPI.

Some cool tools in the toolkit like diagnostics, threaddump. https://www.datainmotion.dev/2019/11/nifi-toolkit-cli-for-nifi-110.html

Apache NiFi REST API

https://nifi.apache.org/docs/nifi-docs/rest-api/index.html

/nifi-api

NiFi Rest Api 1.13.2

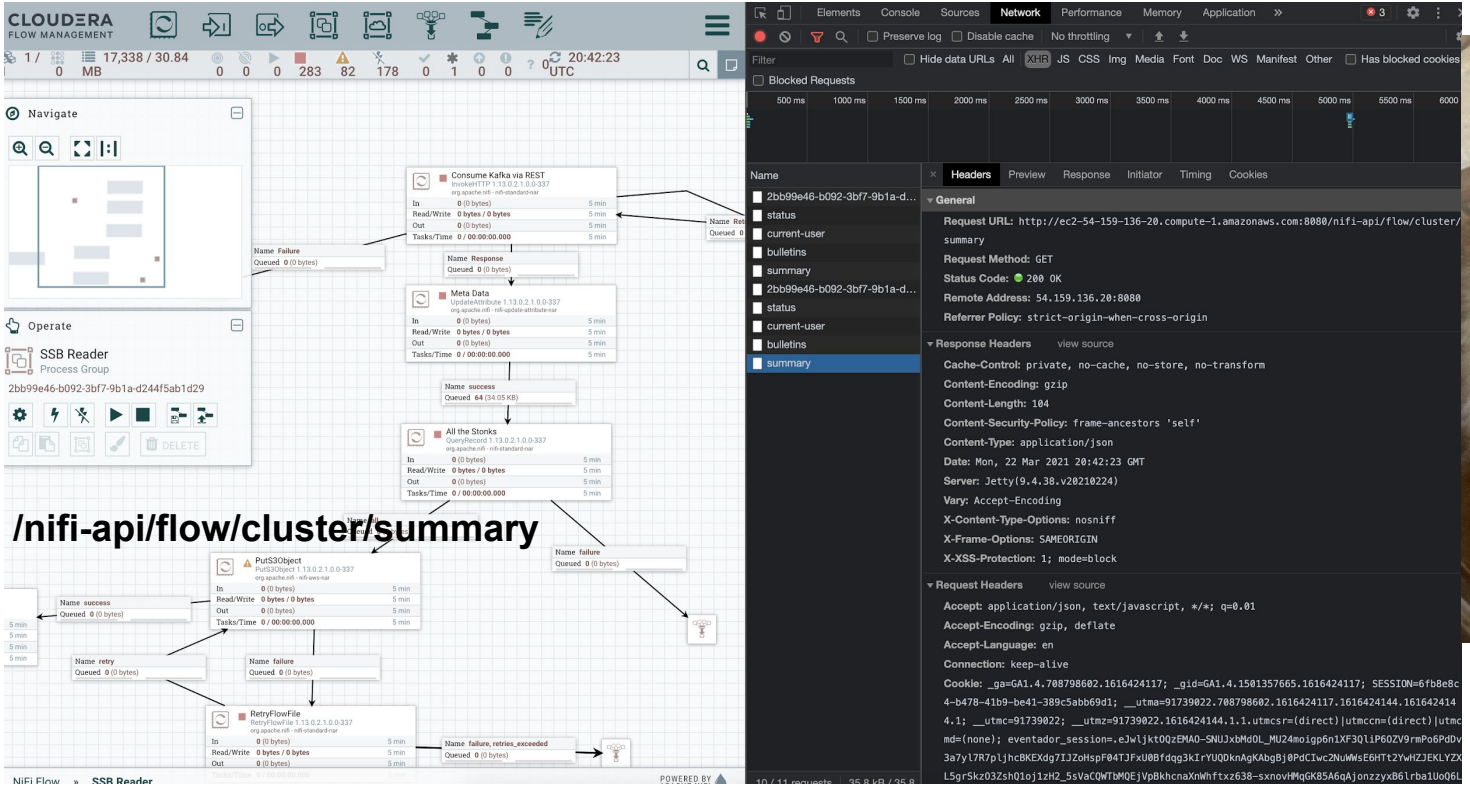
The Rest Api provides programmatic access to command and control a NiFi instance in real time. Start and stop processors, monitor queues, query provenance data, and more. Each endpoint below includes a description, definitions of the expected input and output, potential response codes, and the authorizations required to invoke each service.

Access	*User authentication and token endpoints*
Controller	*Get controller configuration, Manage the cluster, Create reporting tasks*
Controller Services	*Manage controller services, Update controller service references*
Reporting Tasks	*Manage reporting tasks*
Counters	*Get counters, Reset counters*
Flow	*Get the data flow, Obtain component status, Query history*
Process Groups	*Create components, Instantiate a template, Upload a template*
Processors	*Create a processor, Set properties, Schedule*
Connections	*Create a connection, Set queue priority, Update connection destination*
FlowFile Queues	*View queue contents, Download flowfile content, Empty queue*
Input Ports	*Create an input port, Set remote port access control*
Output Ports	*Create an output port, Set remote port access control*
Remote Process Groups	*Create a remote group, Enable transmission*
Labels	*Create a label, Set label style*
Funnels	*Manage funnels*
Provenance	*Query provenance, Search event lineage*
Provenance Events	*Download content, Replay*
Tenants	*Add users and group, Group users*
Policies	*Get policies, Create policies*
Resources	*Get resources*
Site to Site	*Get available ports, Get peers*
Data Transfer	*Send data, Receive data*

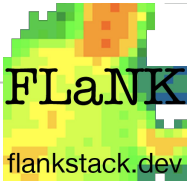
{"clusterSummary":{"connectedNodes":"1 / 1","connectedNodeCount":1,"totalNodeCount":1,"connectedToCluster":true,"clustered":true}}

Apache NiFi REST API

https://nifi.apache.org/docs/nifi-docs/rest-api/index.html



Apache NiFi REST API - Examples



http://localhost:8080/nifi-api/resources
http://localhost:8080/nifi-api/flow/cluster/summary
http://localhost:8080/nifi-api/flow/config
http://localhost:8080/nifi-api/flow/controller/bulletins
http://localhost:8080/nifi-api/flow/history/?offset=1&count=100
http://localhost:8080/nifi-api/flow/processor-types
http://localhost:8080/nifi-api/flow/reporting-tasks
http://localhost:8080/nifi-api/flow/status
nifi-api/system-diagnostics
nifi-api/flow/controller/bulletins
nifi-api/flow/process-groups/root
nifi-api/flow/process-groups/root/controller-services
nifi-api/flow/process-groups/root/status
nifi-api/flow/process-groups/7a01d441-0164-1000-ec7a-54109819f084

Apache NiFi DevOps

Some of this is a bit tricky. I will show you what we can do with DataFlow Experience in a future meetup. Time to hit the command line.

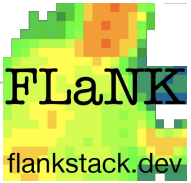
```
cli.sh nifi pg-enable-services -u
http://edge2ai-1.dim.local:8080
--processGroupId root
```

Apache NiFi CLI

nifi pg-list -u http://edge2ai-1.dim.local:8080

#	Name	Id	Running	Stopped	Disabled	Invalid
1	Copy of DF -> DE Webinar	e1ab4d12-dc10-3d55-b888-4e8d0ffd8e27	0	16	0	8
2	Credit Card	ac83ea0c-8f5c-3461-bffe-bf54d96a0490	0	0	5	0
3	DF -> DE Webinar	626876c4-f323-3aa8-933c-020d948370ea	0	17	8	0
4	Fixed Width	caf9f2f4-420a-3bc0-8e3a-a3992cda4b24	0	6	0	16
5	Hbase Processing	d5247783-f5dd-3ca3-8fec-a7d3b3f5fba0	0	0	7	0
6	Kafka to Websockets	e557a6a2-6b13-32bd-9948-bc047475f89e	0	22	0	2
7	Logs	94c0839d-47a8-389d-8866-a18c6893dcf4	0	0	3	0
8	Prices Kafka to Kudu	df1e7acc-8e02-31f6-b5e0-ddea2e013bb9	0	0	5	0
9	Process Events Scada Logs	d3a74c70-5c42-3544-9e5d-7825c2744a22	0	10	0	2
10	Process Sensor Data	5a6bb204-0178-1000-ffff-ffffc104ea42	0	0	9	0
11	Process Sensor Data 2	30e036a8-dd82-392d-8017-777880e697a3	0	0	8	1
12	REST Ingest Meds	93b9b6fb-03d7-3280-a528-59d247c86110	0	19	16	0
13	Read Slack	bcf362b6-a958-346b-b002-e11c22e1de69	0	24	0	4
14	SSB Reader	2bb99e46-b092-3bf7-9b1a-d244f5ab1d29	0	5	0	1
15	SSB Stocks Processing	7500ab5a-fcc7-3964-8bb8-4c2f7a1933cf	0	3	0	2
16	SSB Weather Processing	ccf2c0ef-6df5-30ec-8219-089c98a91c3f	0	1	0	2
17	Satellite Data	e6bbfd06-5ea4-3a39-ba20-d9ad481f73a9	0	0	7	0
18	Sensors MQTT	d2b25567-0cd5-31f5-901e-0bda0299e072	0	2	8	0
19	Simudyne	8edb0546-51b5-33f1-bb0f-1c277065b1e8	0	0	0	0
20	Status	19b070ce-3831-35d0-895a-e664c8a053bd	0	12	0	1
21	Stock	babfc87b-83c7-337a-9872-6ce628c42a7e	0	3	38	0
22	Stocks 2	e37daae6-f579-3551-ba28-fe2eb35c02ca	0	17	0	1
23	Stocks via Finnhub	64849d95-c8e2-38cb-9d00-3e9363017647	0	22	3	2
24	Stocks via Polygon	262075c1-fb52-3a09-bf5b-fe0c0d0fd6a2	0	19	0	5
25	TRANSCOM Data	41913dff-31fa-3098-8826-f6882ba7fc79	0	17	0	3
26	Ticket Data	8bd497a8-0800-34ee-a880-5930cb3249a6	0	1	4	1
27	Travel Warning	2e277203-d323-3cf0-a7ab-0f72eadc3c6a	0	0	3	0
28	Tweets	7dcd17de-6e4f-3f0e-9d53-57a2cfd9db6d	0	1	13	0
29	Twitter to Kudu	dcb6fac0-4f64-34b1-9e01-474f079a26d6	0	0	16	0
30	Validate and Kafka Produce	eba5adbb-dc3f-365e-a9fb-d7ad35a3c68e	0	7	0	2
31	Weather Data	5dd7d437-1ba6-3526-b176-9c51bc117d4f	0	6	0	1
32	aws test	63717dd8-51cc-38cf-8f1b-39b91eddbfe4	0	15	0	13
33	rp4 FLANK	361dafe4-eeaf-3c73-b2f9-ee45669cf0d2	0	23	25	12
34	weather	675887a8-d6e4-30b9-8475-ab5090338c29	0	12	0	2

CLI Tips



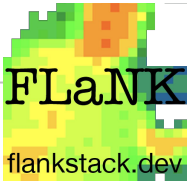
You can run interactive or command at a time.

For output:

- -ot simple
- -ot json

You can loop calls with Shell Script, DevOps tools, NiFi and more.

CLI Overall Commands



nifi **cluster-summary** -u http://localhost:8080
nifi **get-services** -u http://localhost:8080
nifi **get-reporting-task** --baseUrl http://edge2ai-1.dim.local:8080 -verbose --reportingTaskId 07914d9f-1ce3-1174-0000-000039db6547 -ot json
nifi **get-reporting-tasks** --baseUrl http://edge2ai-1.dim.local:8080 -verbose

https://github.com/tspannhw/CloudDemo2021

CLI Process Groups Commands

nifi **pg-list** -u http://localhost:8080
nifi **pg-enable-services** -u http://edge2ai-1.dim.local:8080 --processGroupId root
nifi **pg-status** -u http://localhost:8080 --processGroupId 6608ff51-89bb-3d66-4caf-f86ecaea950d
nifi **pg-status** -u http://localhost:8080 --processGroupId root
nifi **pg-get-services** -u http://localhost:8080 --processGroupId root
nifi **pg-enable-services** -u http://localhost:8080 --processGroupId root
nifi **pg-start** -u http://edge2ai-1.dim.local:8080 -pgid 2c1860b3-7f21-36f4-a0b8-b415c652fc62

CLI Parameter Contexts Commands

nifi **list-param-contexts** -u http://edge2ai-1.dim.local:8080 -verbose
nifi **pg-set-param-context** -u http://edge2ai-1.dim.local:8080 -verbose -pgid 2c1860b3-7f21-36f4-a0b8-b415c652fc62 -pcid 39f0f296-0177-1000-ffff-ffffdccb6d90
nifi **export-param-context** -u http://localhost:8080 -verbose --paramContextId 8067d863-016e-1000-f0f7-265210d3e7dc
nifi **import-param-context** -u http://localhost:8080 -i $f

CLI Registry Commands

registry **list-buckets** -u http://edge2ai-1.dim.local:18080 -verbose
registry **create-flow** -verbose -u http://server:18080 -b 250a5ae5-ced8-4f4e-8b3b-01eb9d47a0d9 --flowName iotFlow
registry **import-flow-version** -verbose -u http://server:18080 -f a5a4ac59-9aeb-416e-937f-e601ca8beba9 -i iot-1.json
registry **list-flows** -u http://server:18080 -b 250a5ae5-ced8-4f4e-8b3b-01eb9d47a0d9
registry **list-flows** -bucketId 36cb79a4-f735-4f77-ba55-606718a9c3c9 -u http://localhost:18080
registry **export-flow-version** -f 5ebc2183-954e-4887-a28c-9d0ee54a02ed -o rainbow.json -ot json

https://dzone.com/articles/devops-for-apache-nifi-17-and-more

Apache NiFi DevOps

FLaNK
flankstack.dev

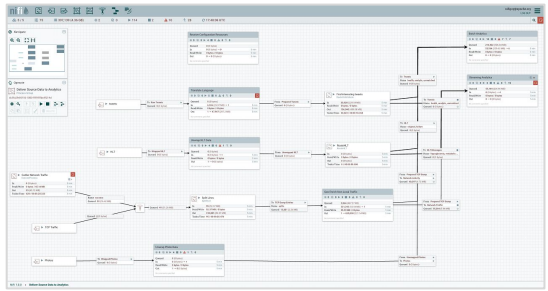
- https://github.com/tspannhw/CloudDemo2021
- https://www.datainmotion.dev/2021/01/automating-starting-services-in-apache.html
- https://github.com/tspannhw/EverythingApacheNiFi
- https://github.com/tspannhw/BackupRegistry
- https://dev.to/tspannhw/backup-and-restore-nifi-registry-templates-14m
- https://dev.to/tspannhw/using-nifi-cli-to-restore-nifi-flows-from-backups-18p9
- https://nifi.apache.org/docs/nifi-docs/html/toolkit-guide.html#nifi_CLI
- https://nipyapi.readthedocs.io/en/latest/
- https://www.datainmotion.dev/2019/04/simple-apache-nifi-operations-dashboard.html
- https://www.datainmotion.dev/2020/09/devops-working-with-parameter-contexts.html

CLOUDERA DATAFLOW DATA-IN-MOTION PLATFORM

CLOUDERA FLOW AND EDGE MANAGEMENT

Enable easy ingestion, routing, management and delivery of any data anywhere *(Edge, cloud, data center)* to any downstream system with built in end-to-end security and provenance

minifi APACHE nifi



APACHE NIFI registry

Advanced tooling to industrialize flow development *(Flow Development Life Cycle)*

ACQUIRE

kafka, HIVE, Spark, APACHE STORM, Solr, SQL, Flink, DynamoDB, cassandra, amazon KINESIS, elastic, S3, mongoDB, slack

- FTP
- SFTP
- HL7
- UDP
- XML
- HTTP
- EMAIL
- HTML
- IMAGE
- SYSLOG

PROCESS

HASH	EVALUATE
MERGE	EXECUTE
EXTRACT	GEOENRICH
DUPLICATE	SCAN
SPLIT	REPLACE
ENCRYPT	TRANSLATE
TALL	CONVERT
ROUTE TEXT	
ROUTE CONTENT	
ROUTE CONTEXT	
ROUTE RATE	
DISTRIBUTE LOAD	

DELIVER

FTP, SFTP, HL7, UDP, XML, HTTP, EMAIL, HTML, IMAGE, SYSLOG

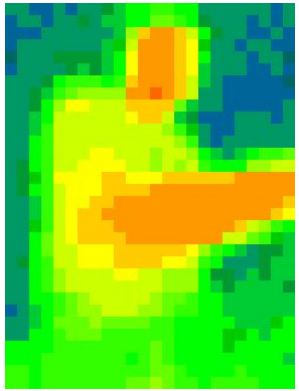
kafka, HIVE, Spark, APACHE STORM, Solr, SQL, Flink, DynamoDB, cassandra, amazon KINESIS, elastic, S3, mongoDB, slack

- Over 300 Prebuilt Processors
- Easy to build your own
- Parse, Enrich & Apply Schema
- Filter, Split, Merger & Route
- Throttle & Backpressure

- Guaranteed Delivery
- Full data provenance from acquisition to delivery
- Diverse, Non-Traditional Sources
- Eco-system integration

TIM SPANN

PRINCIPAL FIELD ENGINEER



THANK YOU