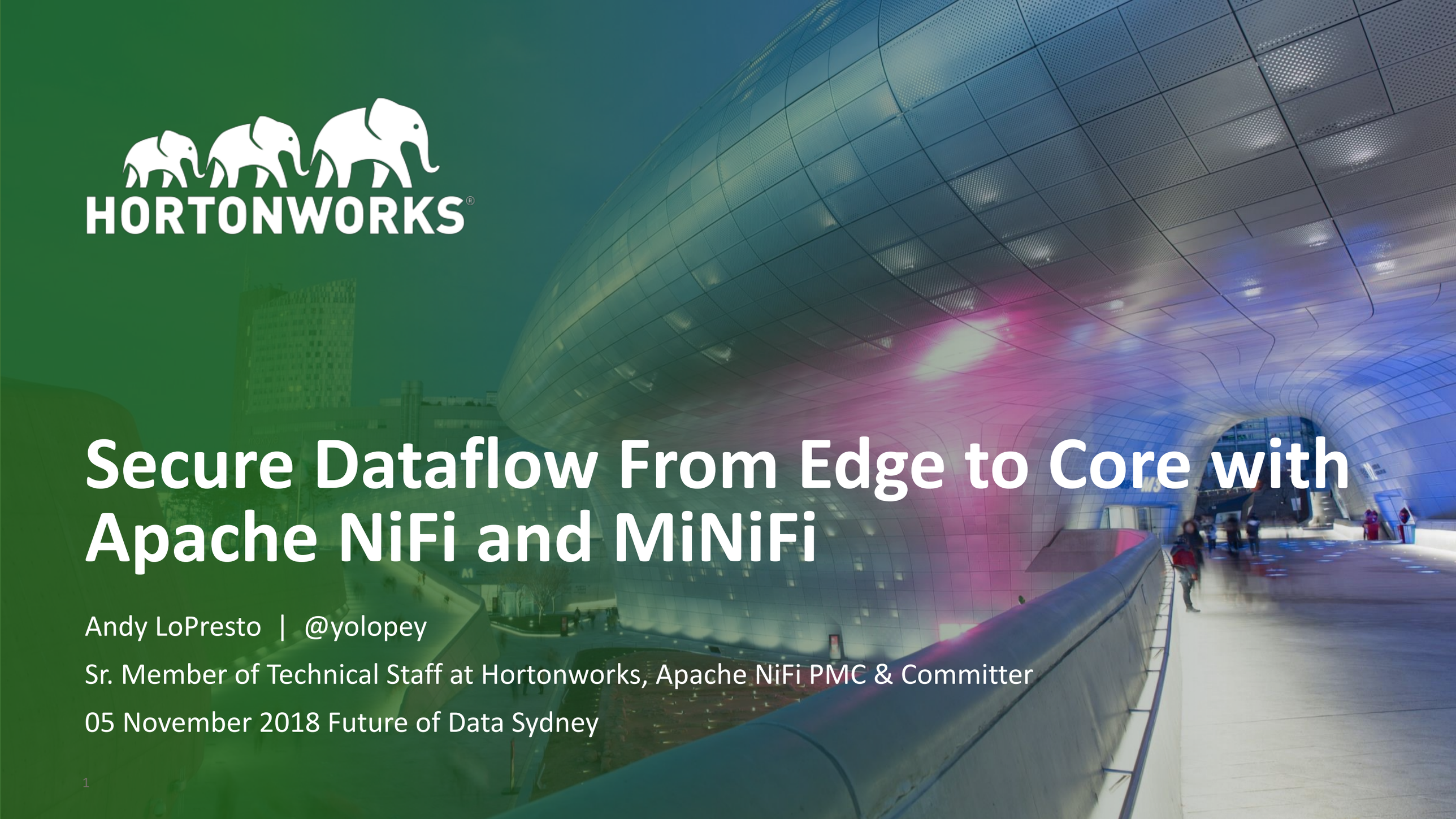


Secure Dataflow From Edge to Core with Apache NiFi and MiNiFi

Andy LoPresto | @yolopey

Sr. Member of Technical Staff at Hortonworks, Apache NiFi PMC & Committer

05 November 2018 Future of Data Sydney

Acknowledgement of Country

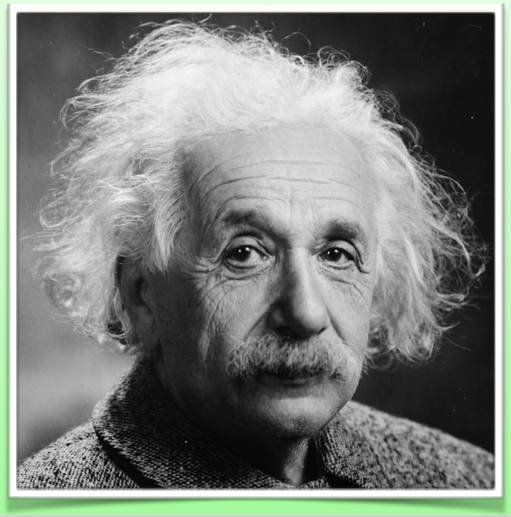
We acknowledge the traditional owners of the land on which we meet today, the Gadigal People of the Eora Nation and pay our respects to Elders past and present.

HORTONWORKS

Gauging Audience Familiarity With NiFi

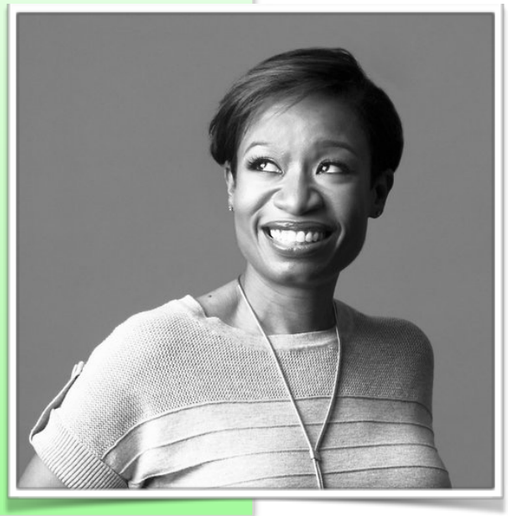
"What's a NeeFee?"

No experience with dataflow
No experience with NiFi



"I can pick this up pretty quickly"

Some experience with dataflow
Some experience with NiFi



"I refactored the Ambari integration endpoint to allow for mutual authentication TLS during my coffee break"

Forgotten more about NiFi than most of us will ever know

HORTONWORKS

Agenda

- *What is dataflow and what are the challenges?*
- Apache NiFi
- Apache MiNiFi
- Apache NiFi Registry
- Security
- New Features
- Roadmap
- Q&A

All slides provided online, no need to transcribe

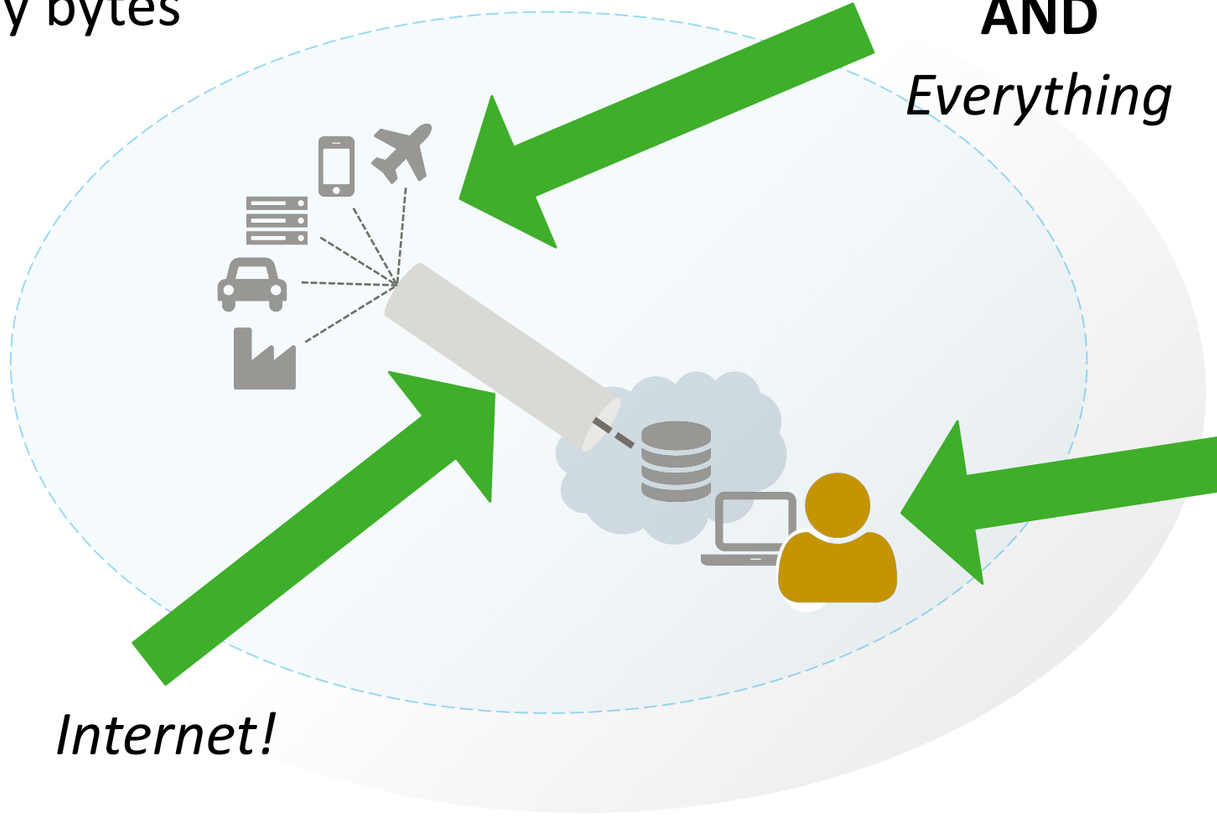
HORTONWORKS

What is dataflow?

HORTONWORKS®

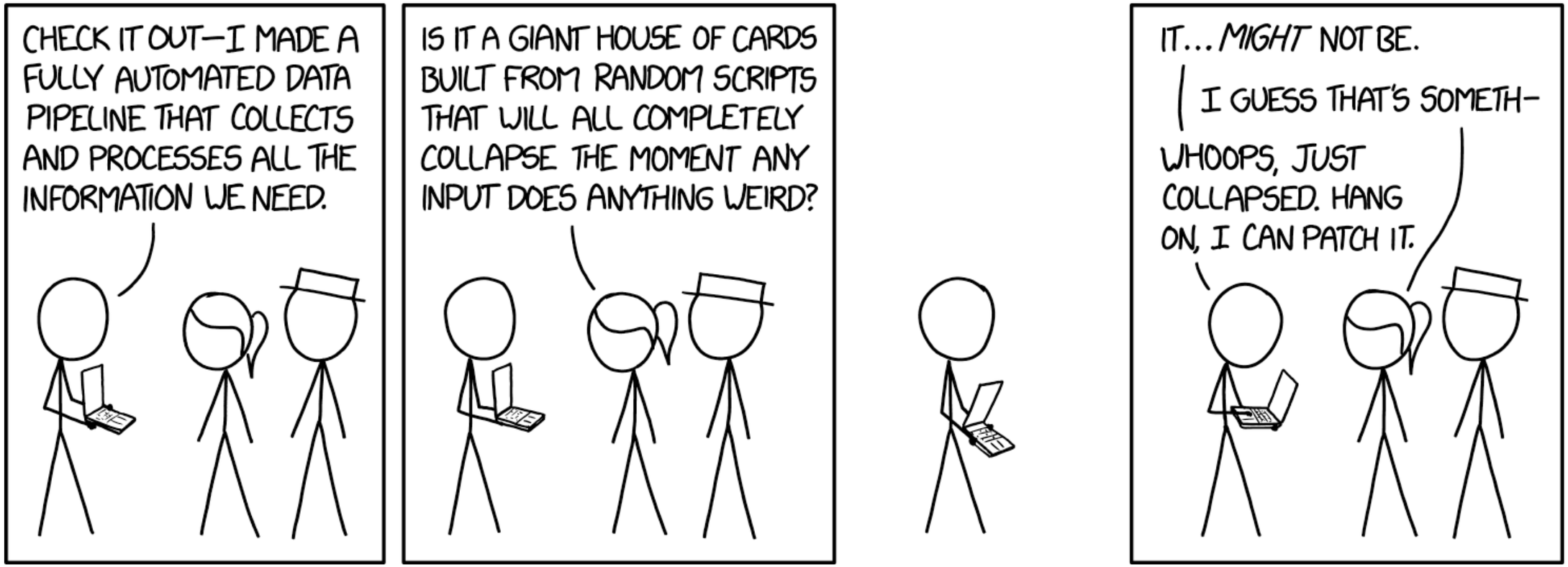
What is dataflow?

- Moving some content from A to B
- Content could be any bytes
  - Logs
  - HTTP
  - XML
  - CSV
  - Images
  - Video
  - Telemetry



HORTONWORKS

Moving data *effectively* is hard



"Data Pipeline" https://xkcd.com/2054/

HORTONWORKS

Dataflow Challenges In 3 Categories

Data

- Standards
- **Formats**
- Protocols
- Veracity
- Validity
- Schemas
- Partitioning/ Bundling

Infrastructure

- “Exactly Once” Delivery
- Ensuring Security
- **Overcoming** Security
- Credential Management
- Network

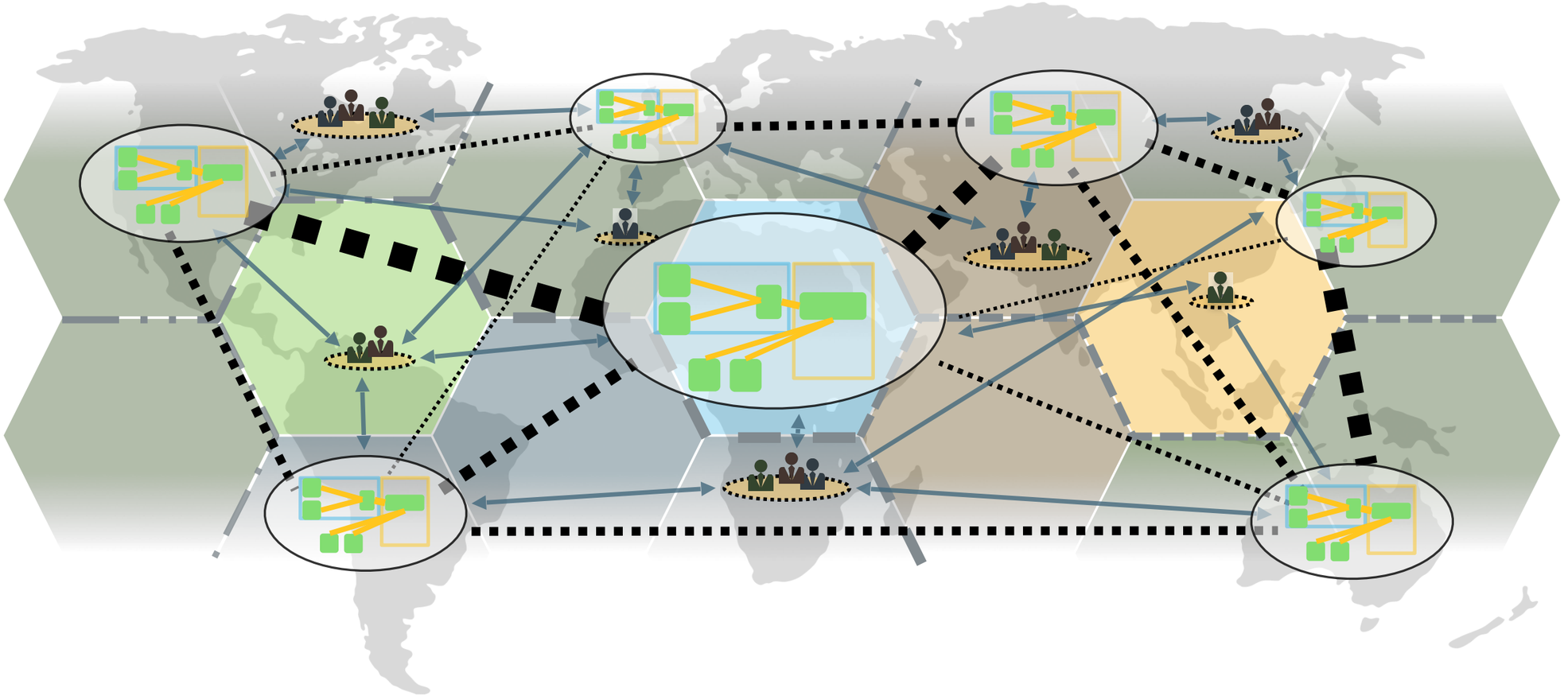
People

- Compliance
- “**That** [person|team|group]”
- **Consumers Change**
- **Requirements Change**
- “Exactly Once” Delivery

HORTONWORKS

Let’s Connect Lots of As to Bs to As to Cs to Bs to Δs to Cs to φs

Raise your hand if you want to maintain Python scripts for the rest of your life



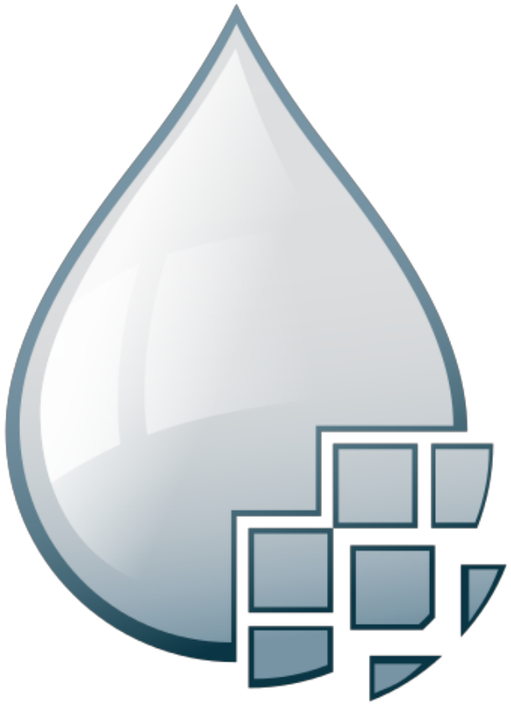
HORTONWORKS

Apache NiFi

HORTONWORKS

Apache NiFi

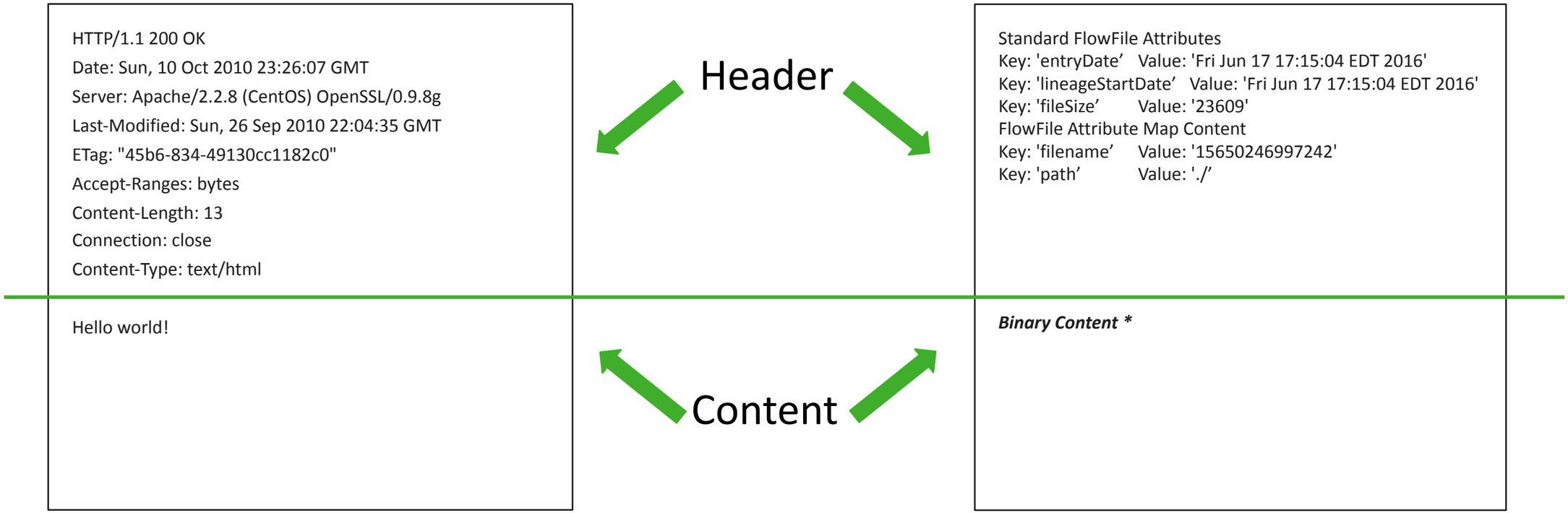
Key Features



- Guaranteed delivery
  - Data buffering
  - Backpressure
- Pressure release
- Prioritized queuing
- Flow specific QoS
  - Latency vs. throughput
  - Loss tolerance
- Data provenance
- Supports push and pull models
- Recovery/recording
  a rolling log of fine-grained history
- Visual command and control
- Flow templates
- Pluggable, multi-tenant security
- Designed for extension
- Clustering

HORTONWORKS

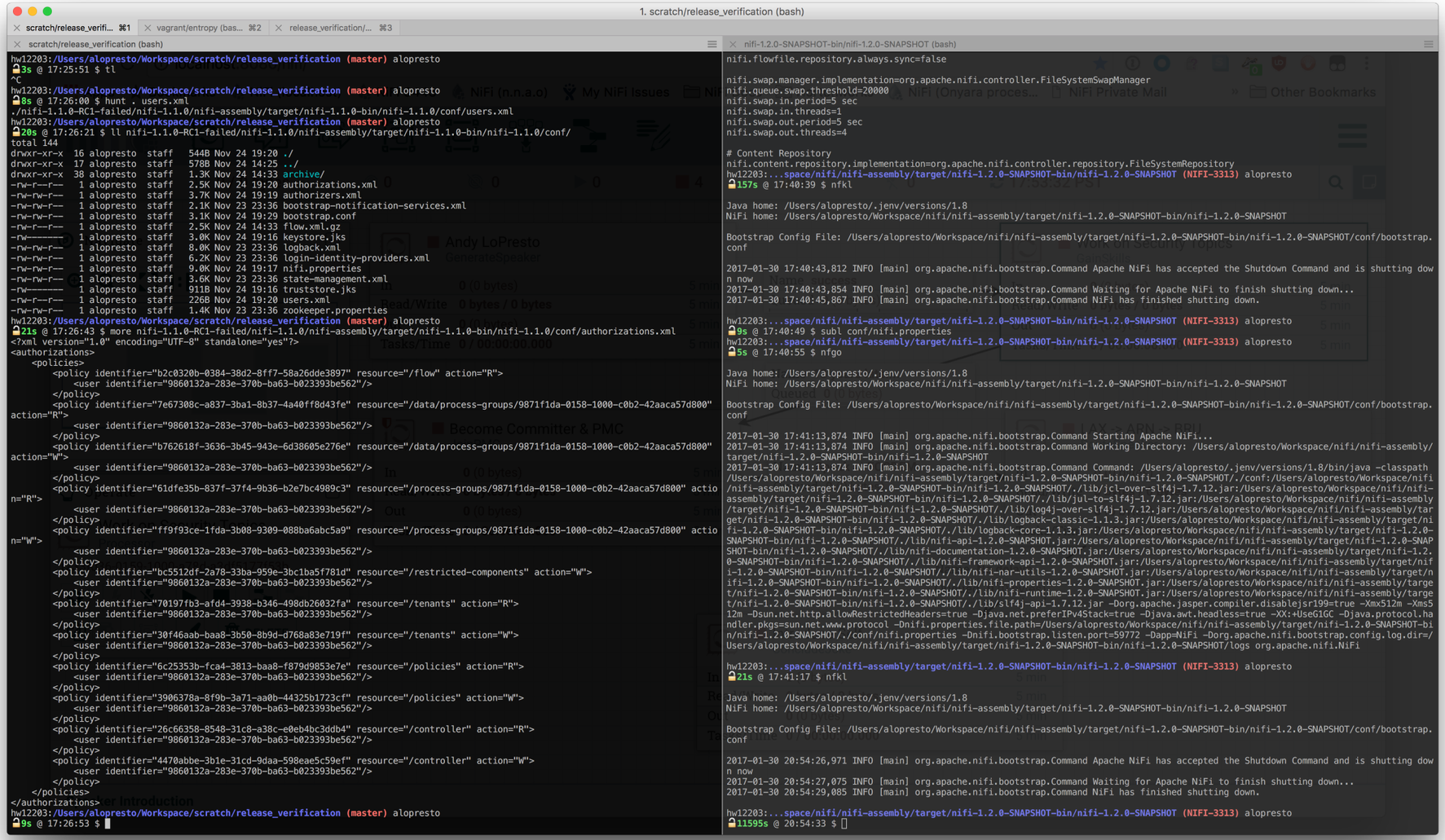
Flowfiles Are Like HTTP Data



HORTONWORKS

User Interface

Less of this… … more of this



HORTONWORKS

Deeper Ecosystem Integration: 286+ Processors, 61 Controller Services

- FTP
- SFTP
- HL7
- UDP
- XML
- HTTP
- WebSocket
- Email
- HTML
- Image
- Syslog
- AMQP

Hash	Encrypt	GeoEnrich
Merge	Tail	Scan
Extract	Evaluate	Replace
Duplicate	Execute	Translate
Split	Fetch	Convert

Parse Records	Convert Records
Route Text	Distribute Load
Route Content	Generate Table Fetch
Route Context	Jolt Transform JSON
Control Rate	Prioritized Delivery

HORTONWORKS

“Where did this chicken come from?”

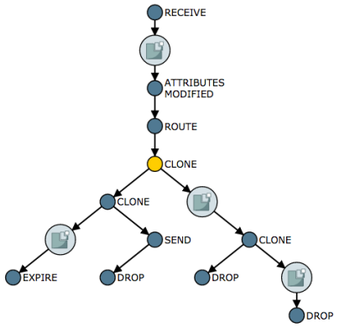
[Colin the Chicken | Portlandia | IFC]

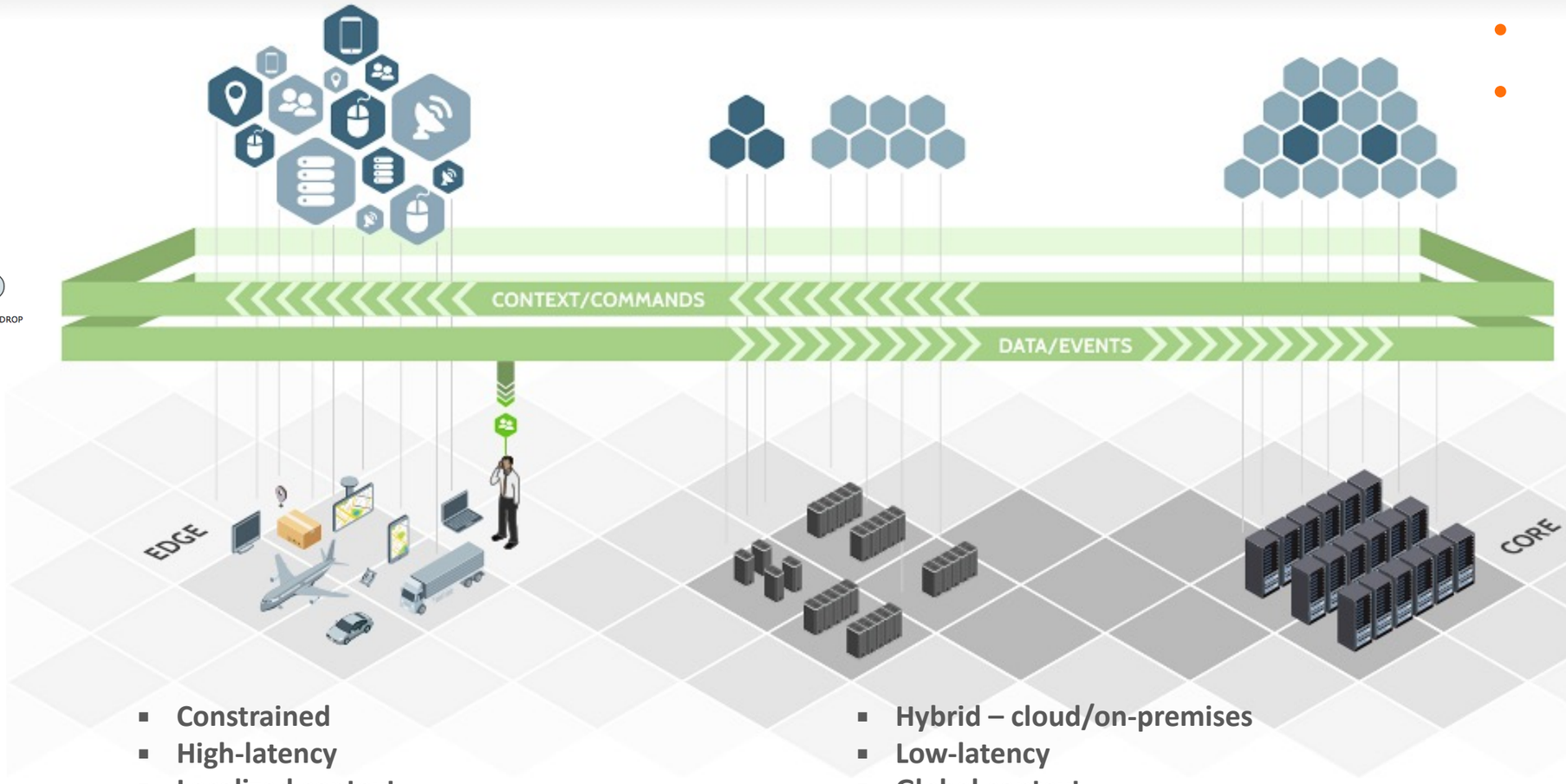
HORTONWORKS

Data Provenance

Origin – attribution
Replay – recovery

Evolution of topologies
Long retention





- **Constrained**
- **High-latency**
- **Localized context**

- **Hybrid – cloud/on-premises**
- **Low-latency**
- **Global context**

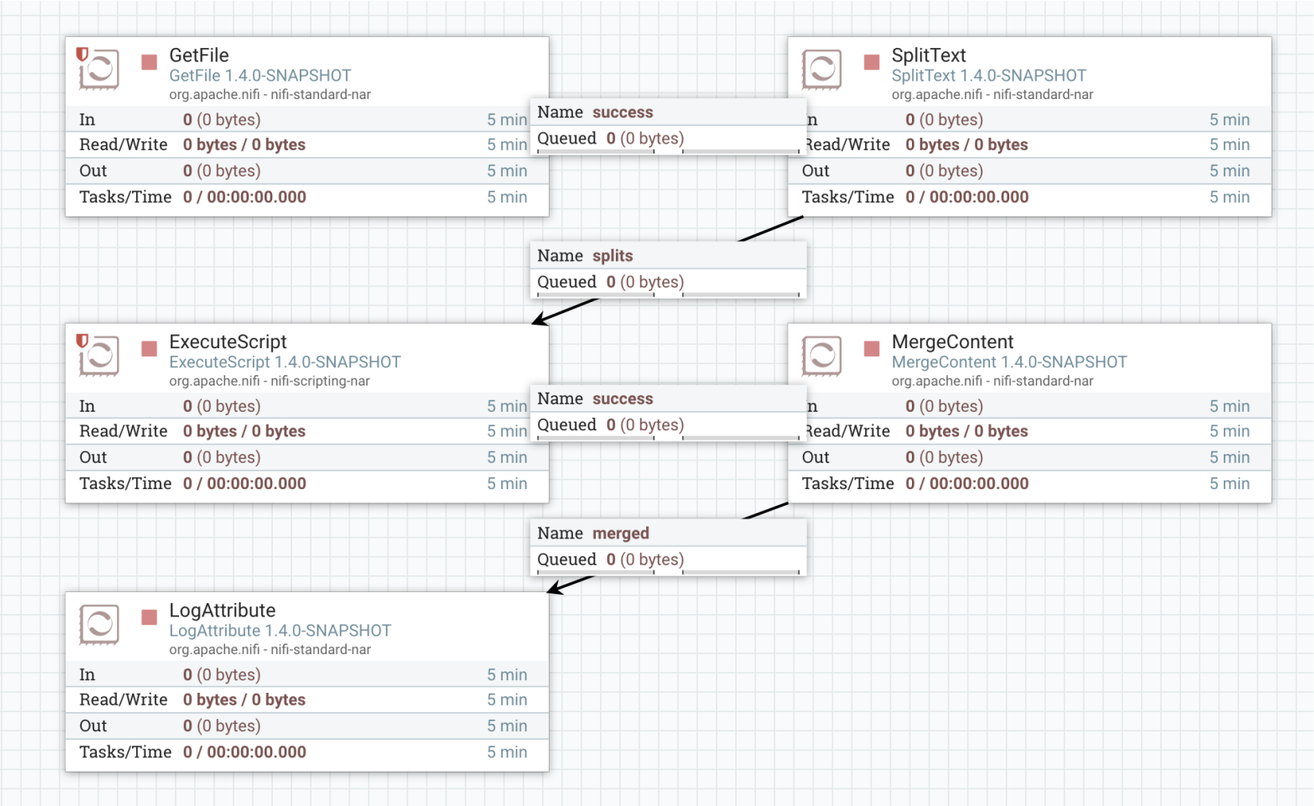
Types of Lineage

- Event
- Configuration

HORTONWORKS

Record Parsing

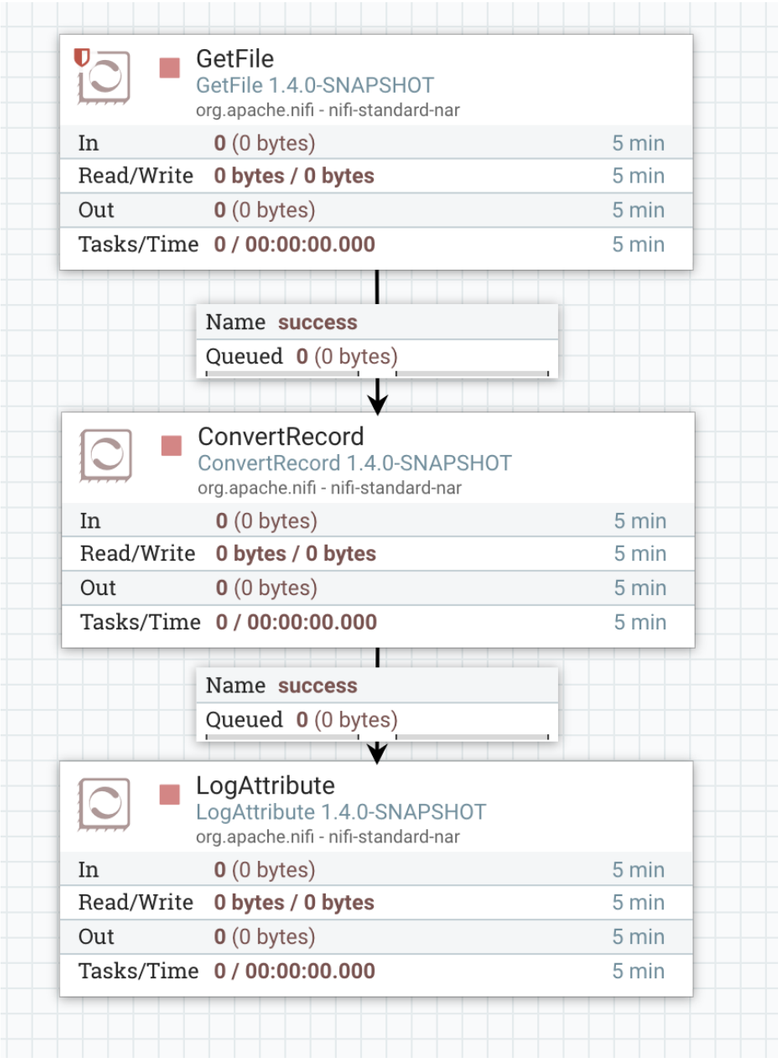
- Previously, data had to be divided into individual flowfiles to perform work
- CSV output with 50k lines would need to be split, operated on, re-merged
- 1 + 50k + 50k + 1 flowfiles = 100k flowfiles



HORTONWORKS

Record Parsing

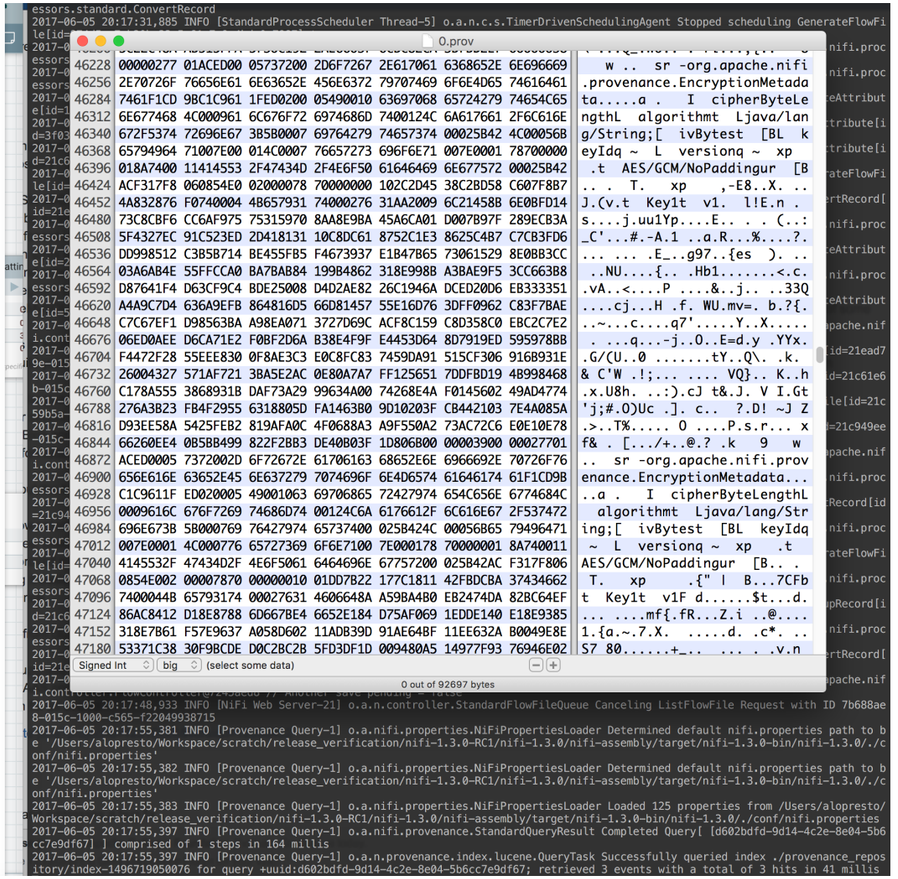
- Now flowfile content can contain many “record” elements
- Read and write with **Reader* and **Writer* Controller Services
- Perform lookups, routing, conversion, SQL queries, validation, and more…
- 1 + 1 flowfiles = 2 flowfiles



HORTONWORKS

Encrypted Provenance Repository

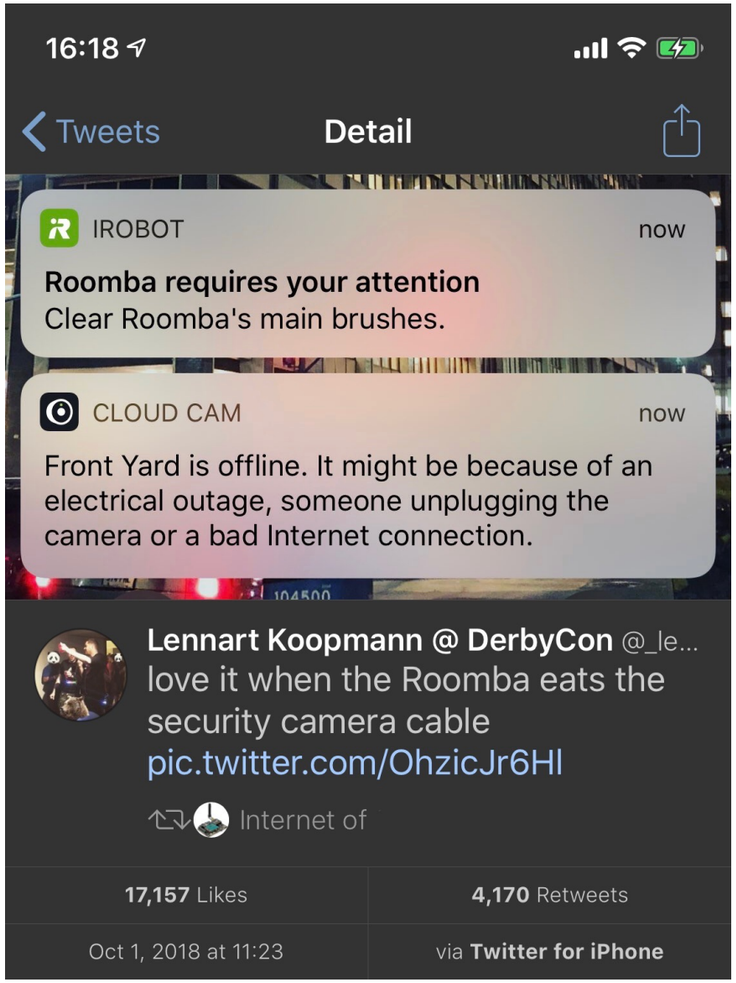
- Every provenance event record is encrypted with AES G/CM before being persisted to disk
- Decrypted on deserialization for retrieval/query
- Random access via offset seek
- Handles key migration & rotation



Apache MiNiFi

IoT Challenges

- Limited computing capability
- Limited power/network
- **Restricted software library/platform availability**
- **No UI**
- Physically inaccessible
- Not frequently updated
- **Competing standards/protocols**
- Scalability
- **Privacy & Security**

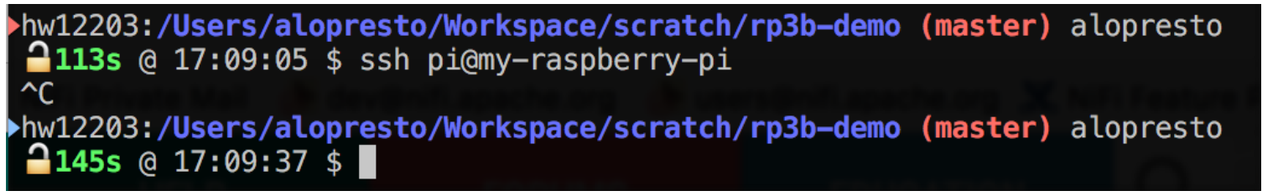


@_lennart

HORTONWORKS

So Why Do We Need A Different Solution?

- NiFi is designed to “own the box”
- NiFi 0.7.x started up in about 10-15 minutes on RP3 (593 MB)
- NiFi 1.x started up in about 30 minutes on RP3 (760 MB)
  - 33 new processors
  - Rewrite for multi tenant authorization
  - Complete UI overhaul



HORTONWORKS

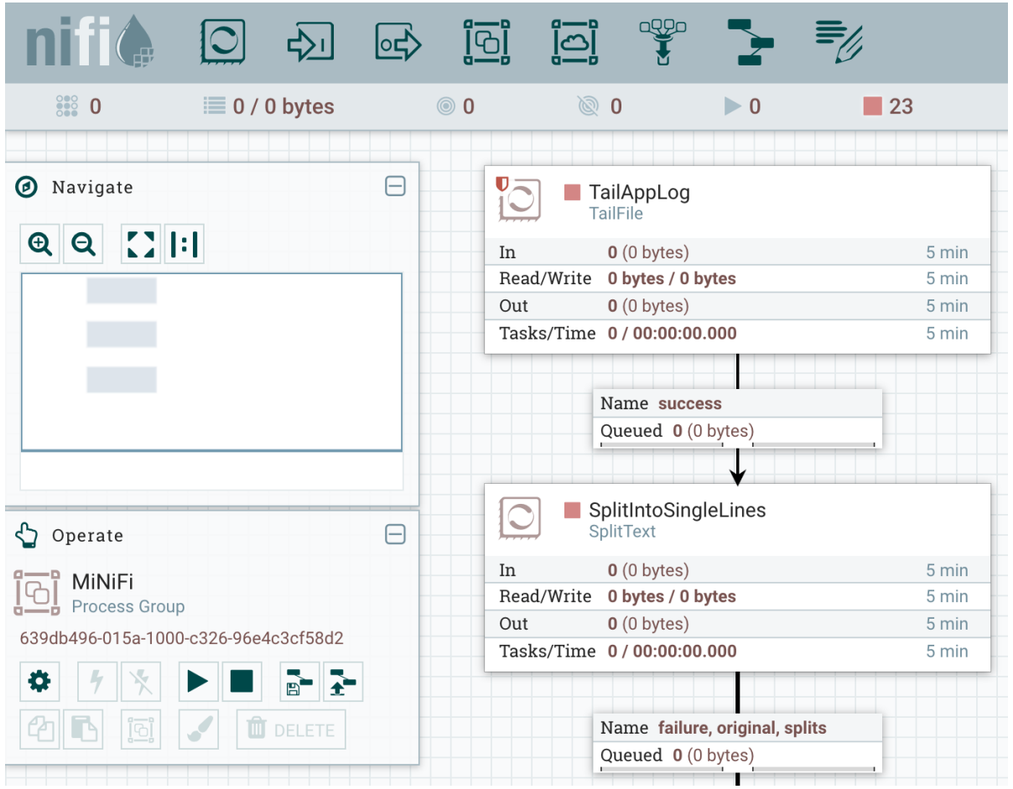
Apache NiFi Subproject: MiNiFi

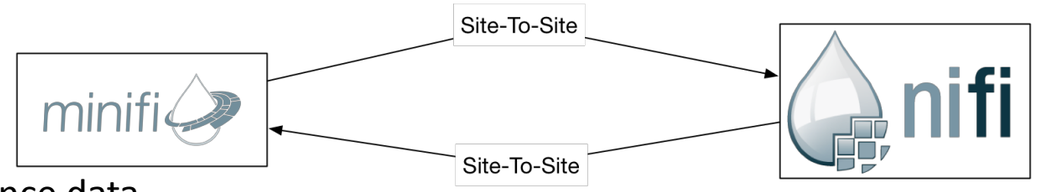
- Get the key parts of NiFi close to where data begins and provide bidirectional communication
- NiFi lives in the data center — give it an enterprise server or a cluster of them
- MiNiFi lives as close to where data is born and is a guest on that device or system
  - IoT
  - Connected car
  - Legacy hardware

HORTONWORKS

Flavors of MiNiFi

- MiNiFi Java (v0.5.0)
  - Modified version of NiFi
    - No UI
    - YAML configuration
  - Reduced processor count
    - 63+ by default, more available with additional NARs
- MiNiFi C++ (v0.5.0)
  - Written from scratch
  - 33 processors by default
  - Bi-directional site-to-site & provenance data



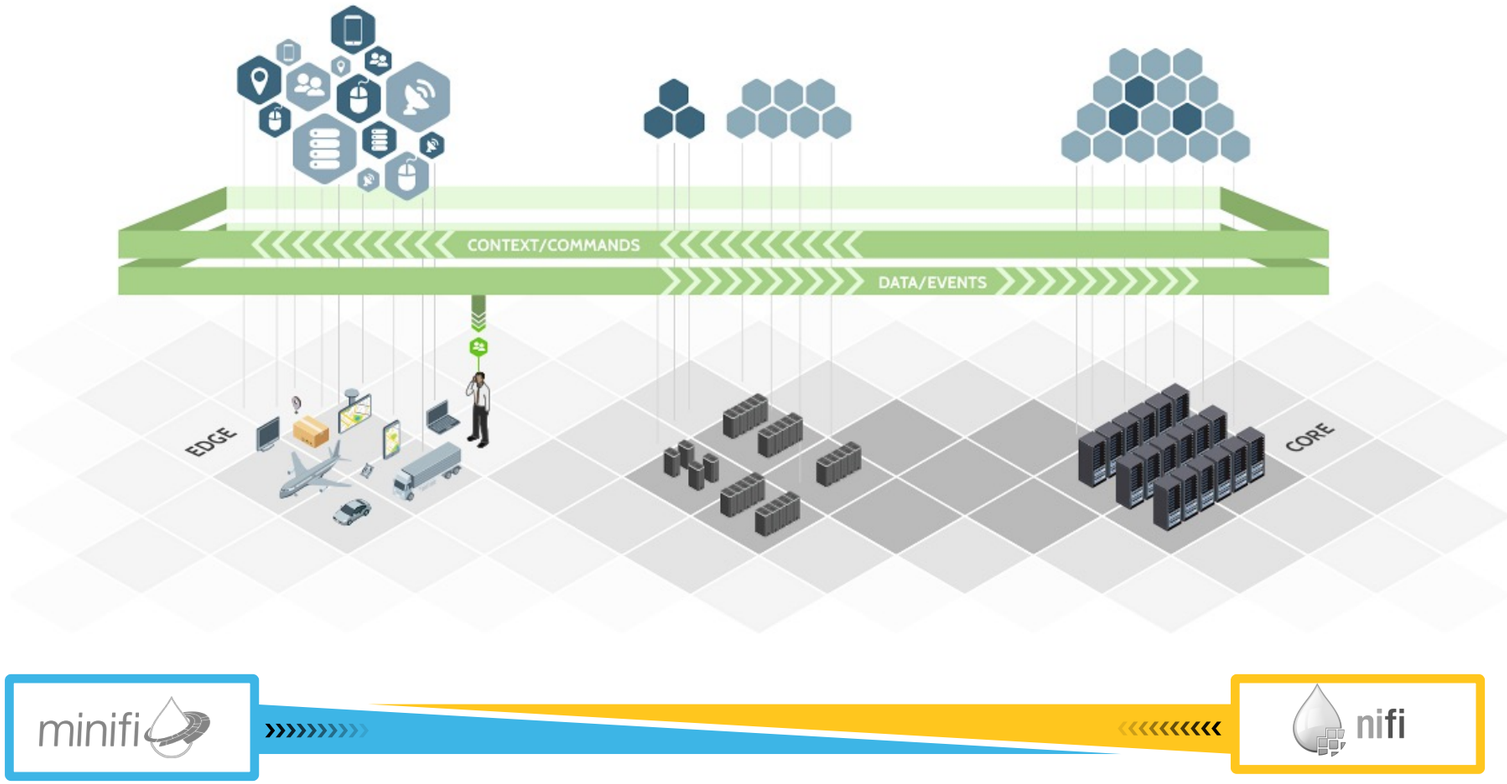



```
Security Properties:
    keystore: /tmp/ssl/localhost-ks.jks
    keystore type: JKS
    keystore password: localtest
    key password: localtest
    truststore: /tmp/ssl/localhost-ts.jks
    truststore type: JKS
    truststore password: localtest
    ssl protocol: TLS
    Sensitive Props:
        key:
        algorithm: PBEWITHMD5AND256BITAES-CBC-OPENSSL
        provider: BC

Processors:
    - name: TailAppLog
      class: org.apache.nifi.processors.standard.TailFile
      max concurrent tasks: 1
      scheduling strategy: TIMER_DRIVEN
      scheduling period: 10 sec
      penalization period: 30 sec
      yield period: 1 sec
      run duration nanos: 0
      auto-terminated relationships list:
      Properties:
          File to Tail: logs/minifi-app.log
          Rolling Filename Pattern: minifi-app*
          Initial Start Position: Beginning of File
    - name: SplitIntoSingleLines
      class: org.apache.nifi.processors.standard.SplitText
      max concurrent tasks: 1
      scheduling strategy: TIMER_DRIVEN
      scheduling period: 0 sec
      penalization period: 30 sec
      yield period: 1 sec
      run duration nanos: 0
      auto-terminated relationships list:
          - failure
          - original
      Properties:
          Line Split Count: 1
          Header Line Count: 0
          Remove Trailing Newlines: true
```


HORTONWORKS

How Does MiNiFi Interact With NiFi?

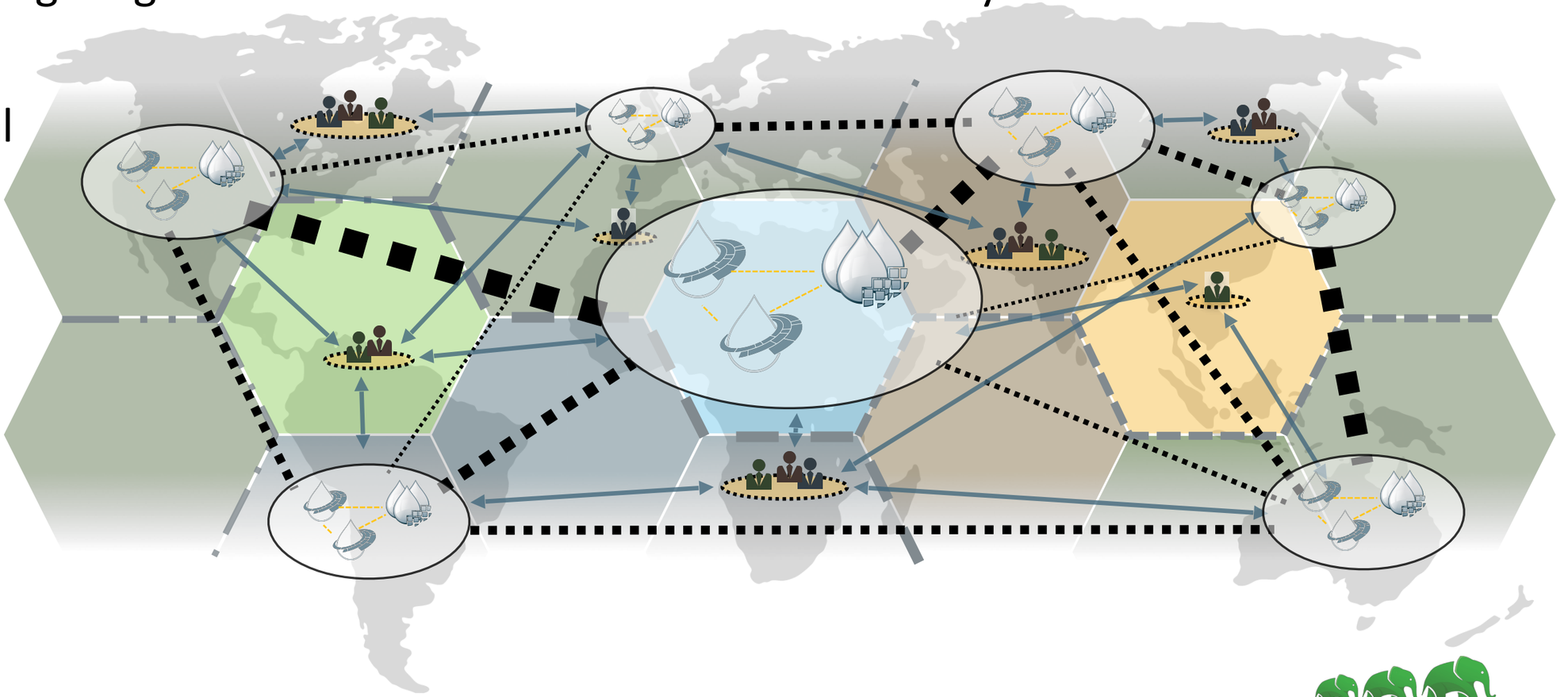
- NiFi
  - Design flows
  - Aggregate data from many sources
  - Perform routing/analysis/SEP
- MiNiFi
  - Receive flows
  - Collect data
  - Send for processing



HORTONWORKS

Let’s Add Dimensionality

- We’ve been imagining EDGE to CORE as a bi-directional linear system
- Let’s expand that to the real world

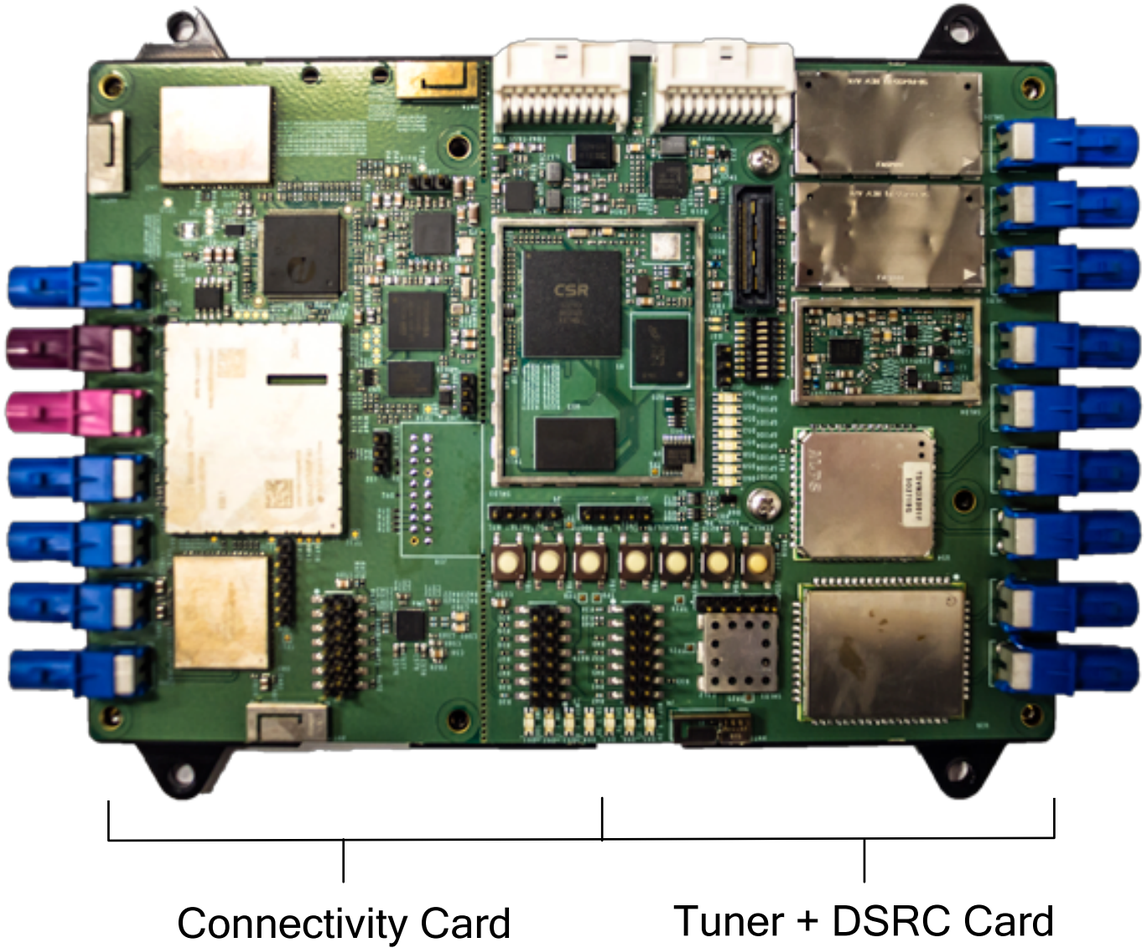


HORTONWORKS

What does MiNiFi provide?

- Data tagging/provenance
- Governance from edge (geopolitical restrictions)
- Security (encryption, certificate-based authentication)
- Low latency (immediate reactions & decision-making)

Connected Car Reference Platform Box

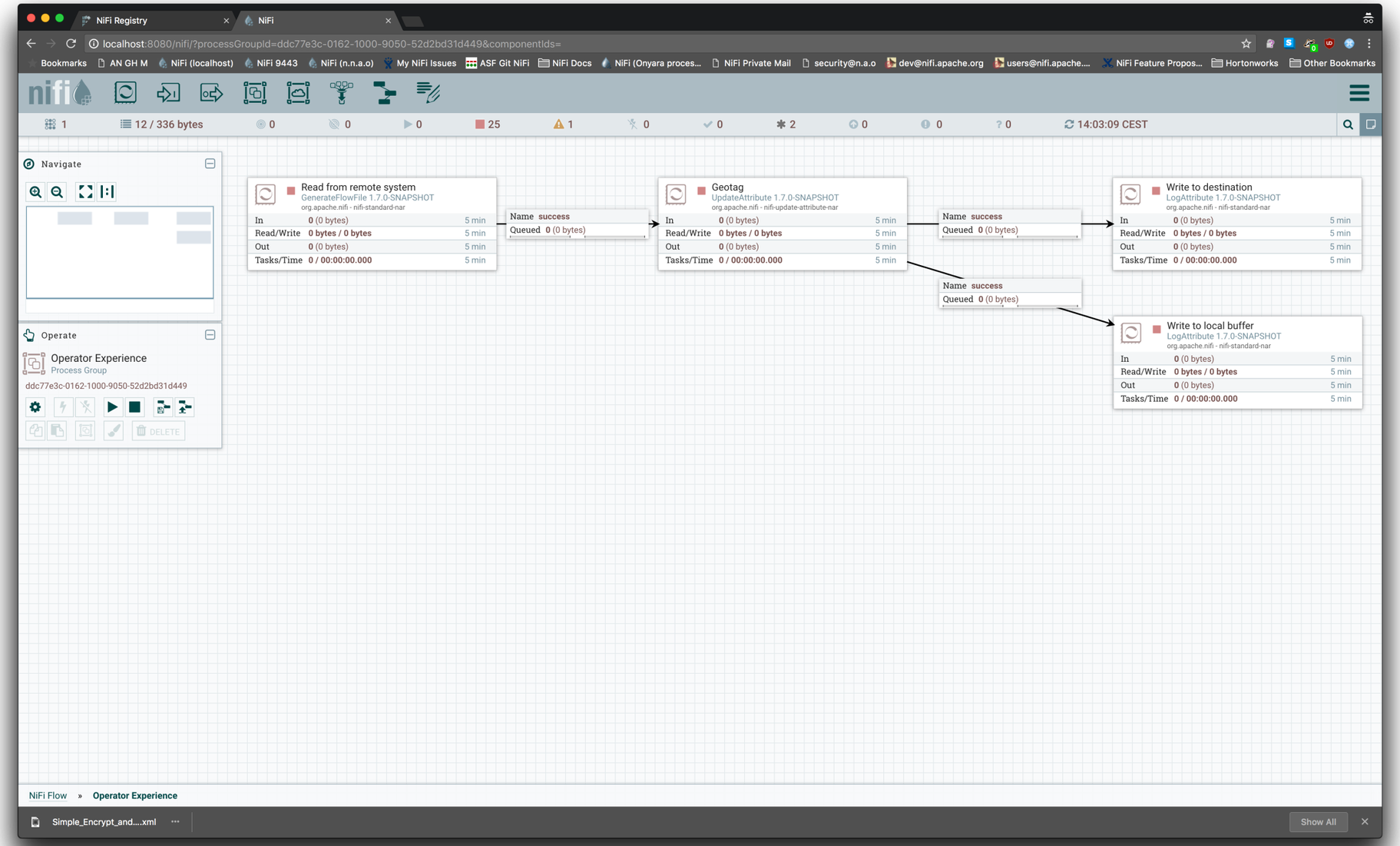


Apache NiFi Registry

Flow Development Lifecycle (FDLC)

- Origins of NiFi
- Operator Experience
  - MC data, don’t drop, mitigate temporarily
- Version Control
- Environment Promotion

Operator Experience



HORTONWORKS

Challenges

- Templates
  - Updates/replacement
  - Sensitive property replacement
- Flow.xml.gz migration
  - Key synchronization
- Environment promotion
  - Approval processes
  - Verifiability

HORTONWORKS

Template Replacement

- Export a new version of template
- Transfer (somehow)
- Verify?
- Import onto canvas side-by-side existing flow
- Stop processors
- Empty queues
- Reconnect queues
- Start
- Pray?

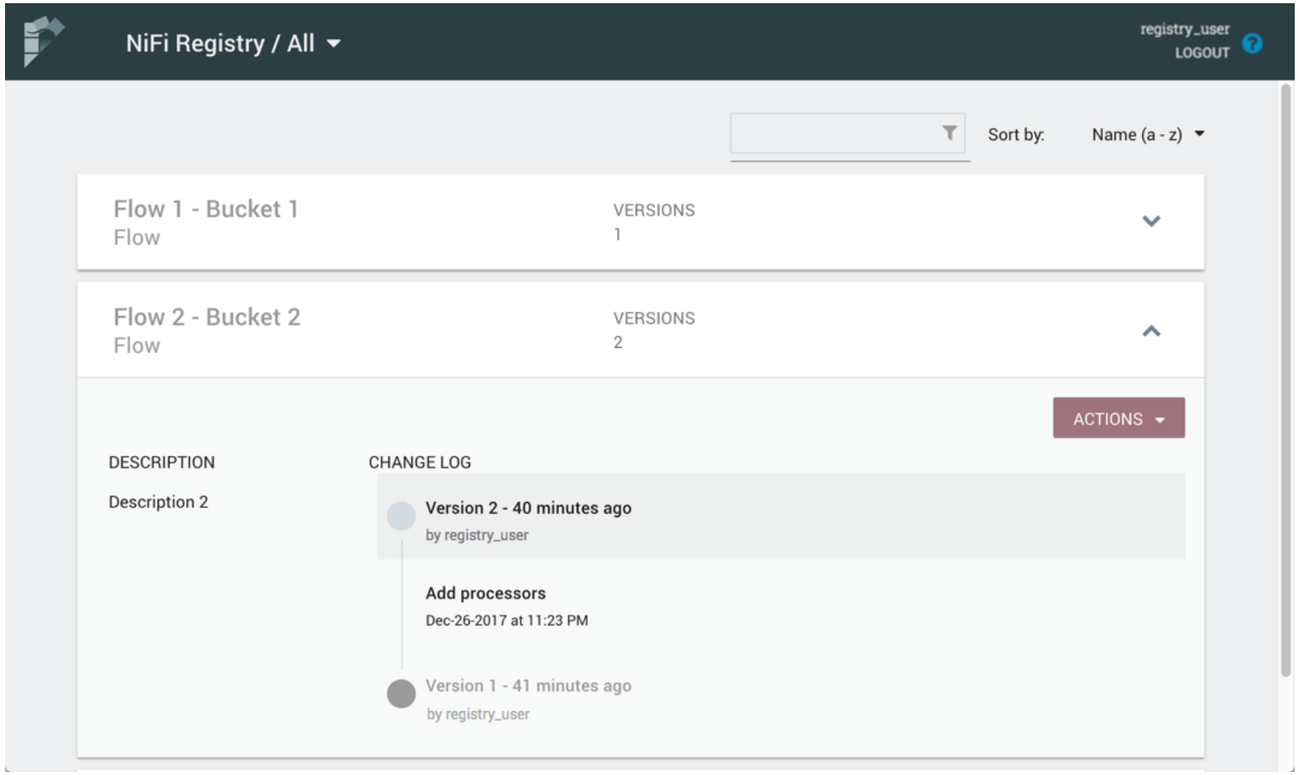
```
-rw-r--r--    1 alopresto  staff   381B Apr 19 15:02 flow_20170626-171416_label_description.xml
-rw-r--r--    1 alopresto  staff   1.4K Apr 19 15:02 flow_20170628-111620_xxe_transform_xml.xml
-rw-r--r--    1 alopresto  staff   1.4K Apr 19 15:02 flow_20170628-155452_xxe_with_patch.xml
-rw-r--r--    1 alopresto  staff   1.4K Apr 19 15:02 flow_20170629-103436_xxe_password_leak.xml
-rw-r--r--    1 alopresto  staff   1.9K Apr 19 15:02 flow_20170717-192339_trusted_hostname.xml
-rw-r--r--    1 alopresto  staff   2.7K Apr 19 15:02 flow_20170718-101751_evaluate_json_path.xml
-rw-r--r--    1 alopresto  staff   5.2K Apr 19 15:02 flow_20170721-175745_site_to_site_secure.xml
-rw-r--r--    1 alopresto  staff   5.2K Apr 19 15:02 flow_20170724-161539.xml
-rw-r--r--    1 alopresto  staff   5.7K Apr 19 15:02 flow_20170724-181056_listen_http_sslv3.xml
-rw-r--r--    1 alopresto  staff   2.3K Apr 19 15:02 flow_20170726-155905_encrypted_spk.xml
-rw-r--r--    1 alopresto  staff   5.2K Apr 19 15:02 flow_20170802-172619_simple_gen_and_log.xml
-rw-r--r--    1 alopresto  staff   5.2K Apr 19 15:02 flow_20170804-164011_simple_gen_and_log.xml
-rw-r--r--    1 alopresto  staff   1.7K Apr 19 15:02 flow_20170807-154118_groovy_json_uppercase.xml
-rw-r--r--    1 alopresto  staff   1.8K Apr 19 15:02 flow_20170807-220536_put_mongo_record.xml
-rw-r--r--    1 alopresto  staff   6.1K Apr 19 15:02 flow_20170809-105304_date_difference.xml
-rw-r--r--    1 alopresto  staff   6.4K Apr 19 15:02 flow_20170809-174400_json_record_path.xml
-rw-r--r--    1 alopresto  staff   1.3K Apr 19 15:02 flow_20170809-210516_validate_xml.xml
-rw-r--r--    1 alopresto  staff   1.4K Apr 19 15:02 flow_20170815-102522_json_to_sql.xml
-rw-r--r--    1 alopresto  staff   1.5K Apr 19 15:02 flow_20170822-104407_json_formatter.xml
-rw-r--r--    1 alopresto  staff   1.7K Apr 19 15:02 flow_20170823-123729_evaluatexpath.xml
-rw-r--r--    1 alopresto  staff   2.5K Apr 19 15:02 flow_20170825-115758_encrypt_and_decrypt.xml
-rw-r--r--    1 alopresto  staff   2.0K Apr 19 15:02 flow_20170825-121351_encrypt_decrypt_clean.xml
-rw-r--r--    1 alopresto  staff   2.0K Apr 19 15:02 flow_20170825-121656_encrypt_decrypt_aligned.xml
-rw-r--r--    1 alopresto  staff   2.3K Apr 19 15:02 flow_20170825-190225_listen_http_and_invoke_http_with_tls1_2.xml
-rw-r--r--    1 alopresto  staff   5.0K Apr 19 15:02 flow_20170831-153220_psaltis_meetup_template_local.xml
-rw-r--r--    1 alopresto  staff   4.0K Apr 19 15:02 flow_20170905-175343_xxe_template.xml
-rw-r--r--    1 alopresto  staff   4.3K Apr 19 15:02 flow_20170905-185907_funnel_bus.xml
-rw-r--r--    1 alopresto  staff   8.1K Apr 19 15:02 flow_20170906-111801_chained_get_html_element_NIFI-4356.xml
-rw-r--r--    1 alopresto  staff   2.5K Apr 19 15:02 flow_20170909-121848_sydney_demo.xml
-rw-r--r--    1 alopresto  staff   3.2K Apr 19 15:02 flow_20170919-202215_meetup_sydney.xml
-rw-r--r--    1 alopresto  staff   3.8K Apr 19 15:02 flow_20170920-190328_crash_course.xml
-rw-r--r--    1 alopresto  staff   2.9K Apr 19 15:02 flow_20170926-180548_handle_http_with_tls1_2.xml
-rw-r--r--    1 alopresto  staff   1.7K Apr 19 15:02 flow_20170929-172333_encrypt_decrypt_1_4_0_verification.xml
-rw-r--r--    1 alopresto  staff   6.3K Apr 19 15:02 flow_20171003-192928_update_record_and_replace_text.xml
-rw-r--r--    1 alopresto  staff   8.9K Apr 19 15:02 flow_20171004-180529_web_server_and_json.xml
-rw-r--r--    1 alopresto  staff   3.8K Apr 19 15:02 flow_20171103-183334_groovy_json_to_sql_srsw.xml
-rw-r--r--    1 alopresto  staff   1.5K Apr 19 15:02 flow_20171127-190651_local_site_to_site.xml
-rw-r--r--    1 alopresto  staff   2.1K Apr 19 15:02 flow_20180101-145059_registry_rc_verification.xml
-rw-r--r--    1 alopresto  staff   1.4K Apr 19 15:02 flow_20180104-104211_test_count_text.xml
-rw-r--r--    1 alopresto  staff   2.0K Apr 19 15:02 flow_20180104-112138_test_count_text_remote.xml
-rw-r--r--    1 alopresto  staff   1.7K Apr 19 15:02 flow_20180202-100218_consume_jms_test.xml
-rw-r--r--    1 alopresto  staff   1.7K Apr 19 15:02 flow_20180202-105514.xml
-rw-r--r--    1 alopresto  staff   1.7K Apr 19 15:02 flow_20180202-131629_consume_jms_10_threads.xml
-rw-r--r--    1 alopresto  staff   1.7K Apr 19 15:02 flow_20180202-160620_consume_jms_multithread.xml
-rw-r--r--    1 alopresto  staff   1.6K Apr 19 15:02 flow_20180208-200731_simple_demo_route_on_time_parity.xml
-rw-r--r--    1 alopresto  staff   5.2K Apr 19 15:02 flow_20180214-182110_jms_deserialization_test.xml
-rw-r--r--    1 alopresto  staff   2.6K Apr 19 15:02 flow_20180216-205336_attribute_evaluator.xml
-rw-r--r--    1 alopresto  staff   5.7K Apr 19 15:02 flow_20180228-130457_attribute_soq.xml
-rw-r--r--    1 alopresto  staff   8.3K Apr 19 15:02 flow_20180307-171740_NIFI-4928.xml
-rw-r--r--    1 alopresto  staff   1.5K Apr 19 15:02 flow_20180313-171310_python_process_attribute.xml
-rw-r--r--    1 alopresto  staff   2.4K Apr 19 15:02 flow_20180315-113634_NIFI-4246-OAuth.xml
-rw-r--r--    1 alopresto  staff   1.9K Apr 19 15:02 flow_20180328-115056.xml
-rw-r--r--    1 alopresto  staff   2.5K Apr 19 15:02 flow_20180405-111419_prioritizers.xml
```


HORTONWORKS

NiFi Registry for Dataflows

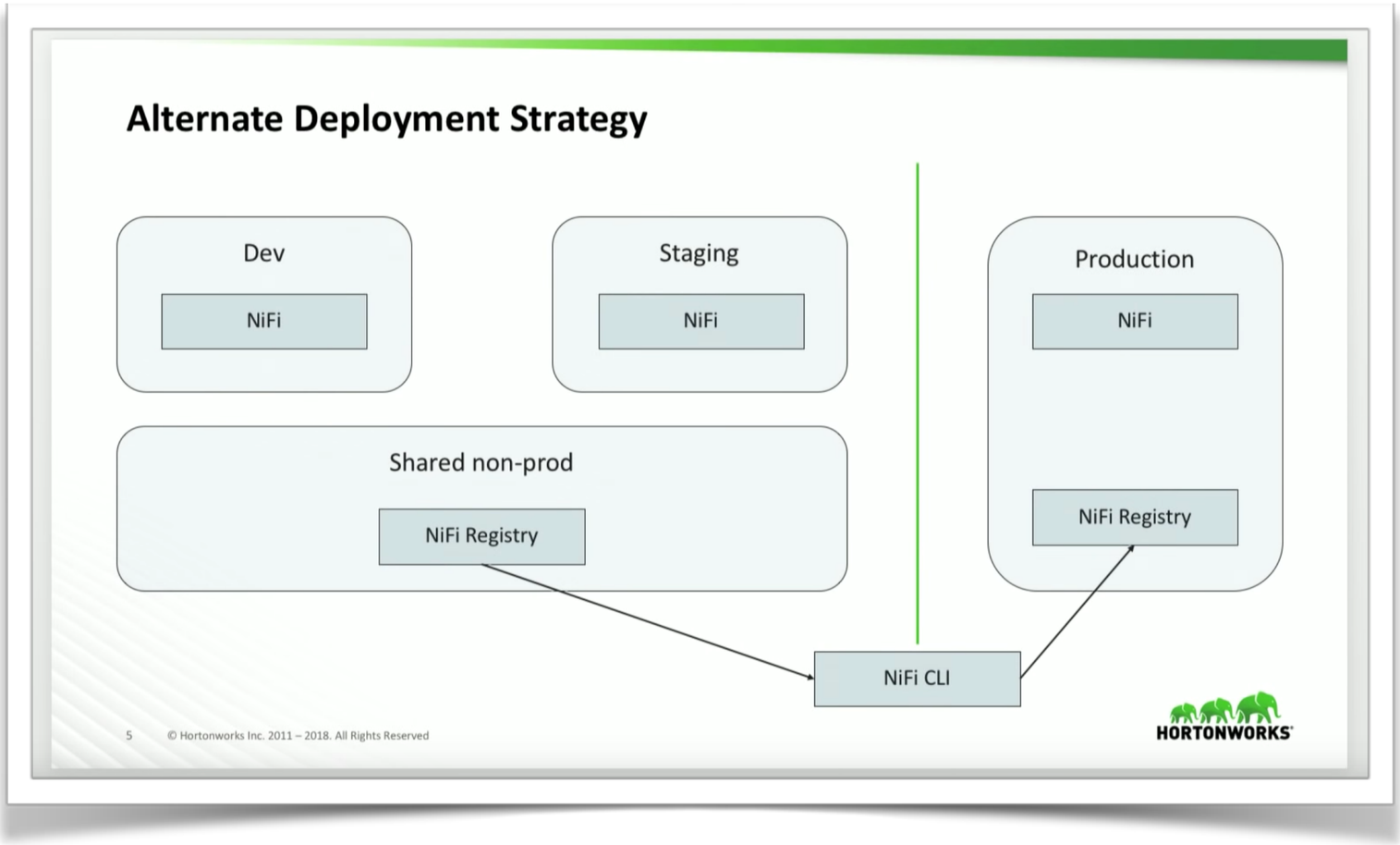
Introducing Apache NiFi Registry 0.3.0

- Previously, flows were exported via XML templates
  - Didn’t contain sensitive values
  - Couldn’t be updated in-place
  - No tracking system
- NiFi Registry brings asset management as first-class citizen to NiFi
- Flows can be versioned



Flows can be promoted between environments

- Connect multiple NiFi instances to a NiFi Registry instance
- Communicate between multiple NiFi Registry instances
  - via multiple Registry Clients
  - via *NiFi CLI*



HORTONWORKS

Extensibility

- Git-backed persistence
  - Share flows via GitHub, etc.
- Commit hooks
  - Register a hook & action
  - “When a new version of the flow is committed to QA Registry, email the QA team and post in the QA Deploy Slack channel”
- Pluggable DB implementations

Event Hooks

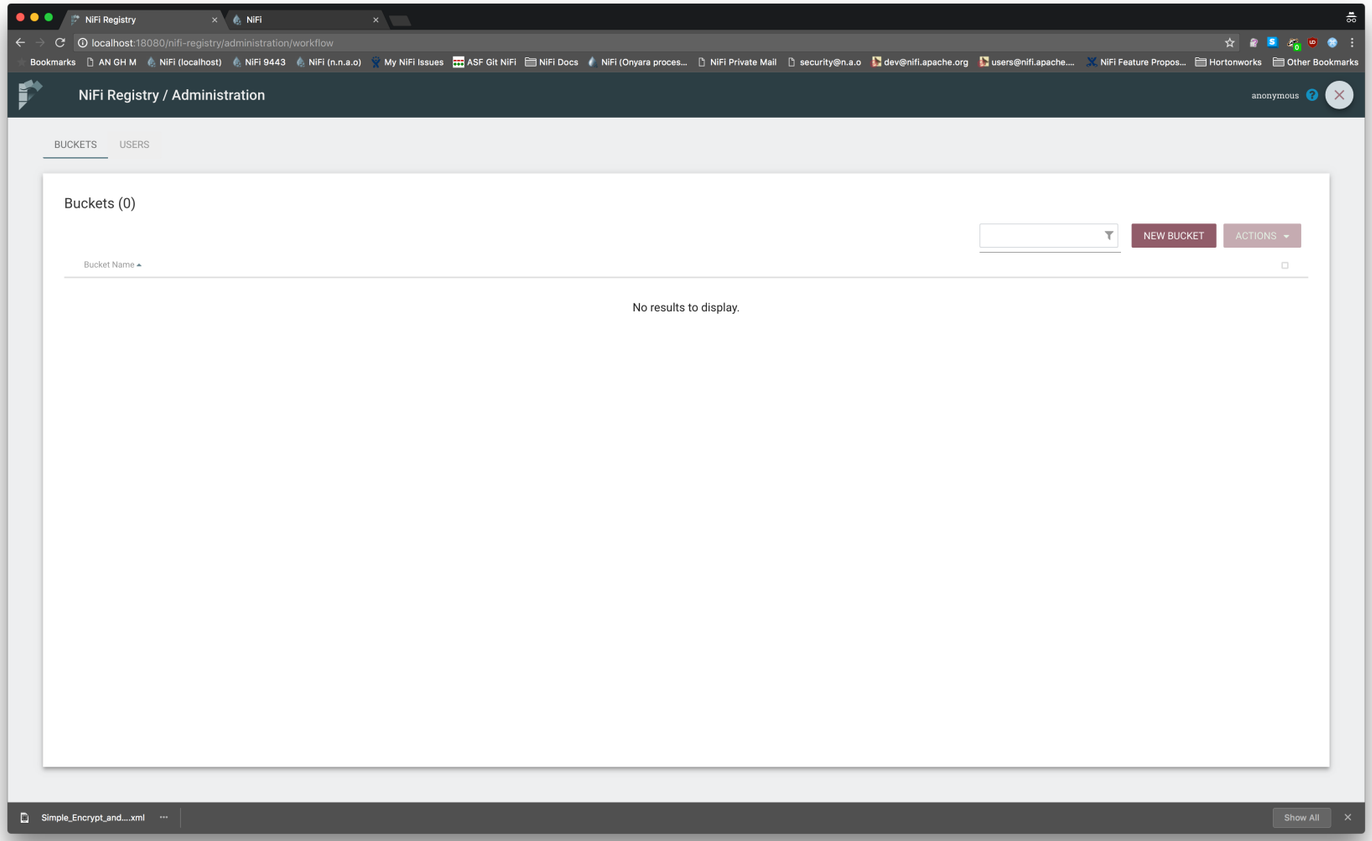
Event hooks are an integration point that allows for custom code to to be triggered when NiFi Registry application events occur.

Event Name	Description
CREATE_BUCKET	A new registry bucket is created.
CREATE_FLOW	A new flow is created in a specified bucket. Only triggered on first time creation of a flow with a given name.
CREATE_FLOW_VERSION	A new version for a flow has been saved in the registry.
UPDATE_BUCKET	A bucket has been updated.
UPDATE_FLOW	A flow that exist in a bucket has been updated.
DELETE_BUCKET	An existing bucket in the registry is deleted.
DELETE_FLOW	An existing flow in the registry is deleted.
REGISTRY_START	Invoked once the NiFi Registry application has been successfully started. This is only invoked after a complete and successful start.

HORTONWORKS

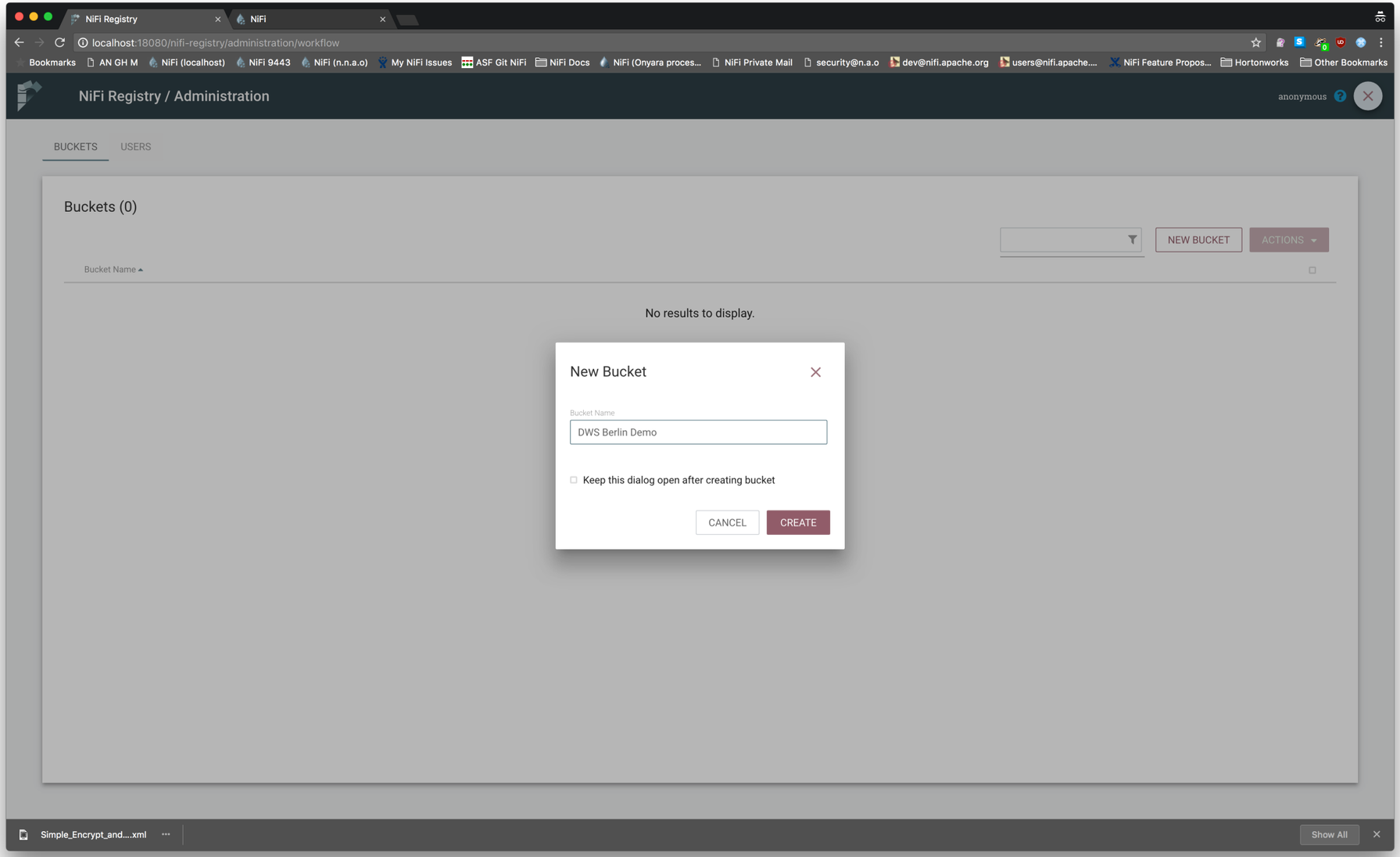
Create Registry

- Install nifi-registry
- `$ mvn clean install`
- `$ ./bin/nifi-registry.sh start`
- Browse to http://localhost:18080

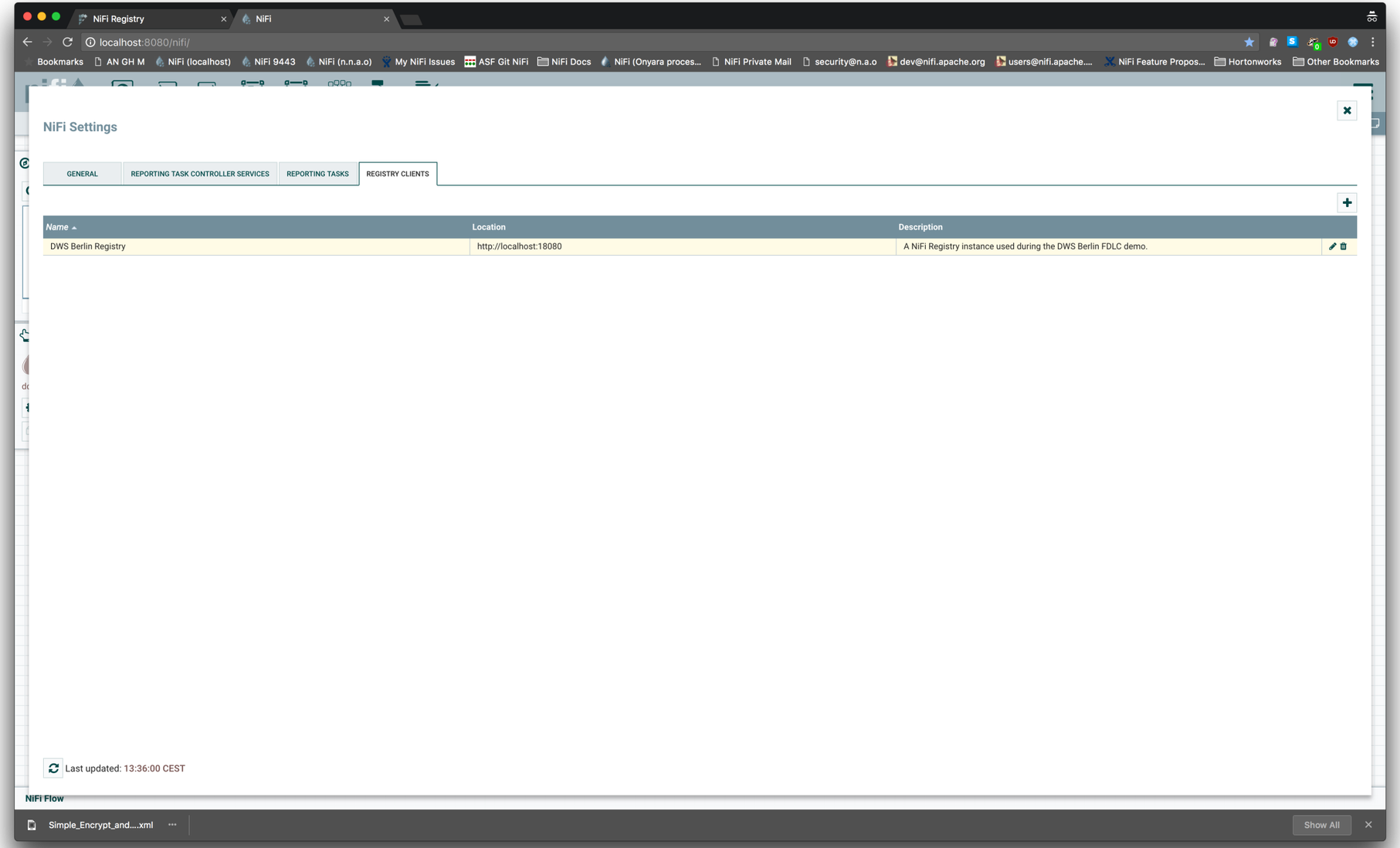


HORTONWORKS

Create Bucket

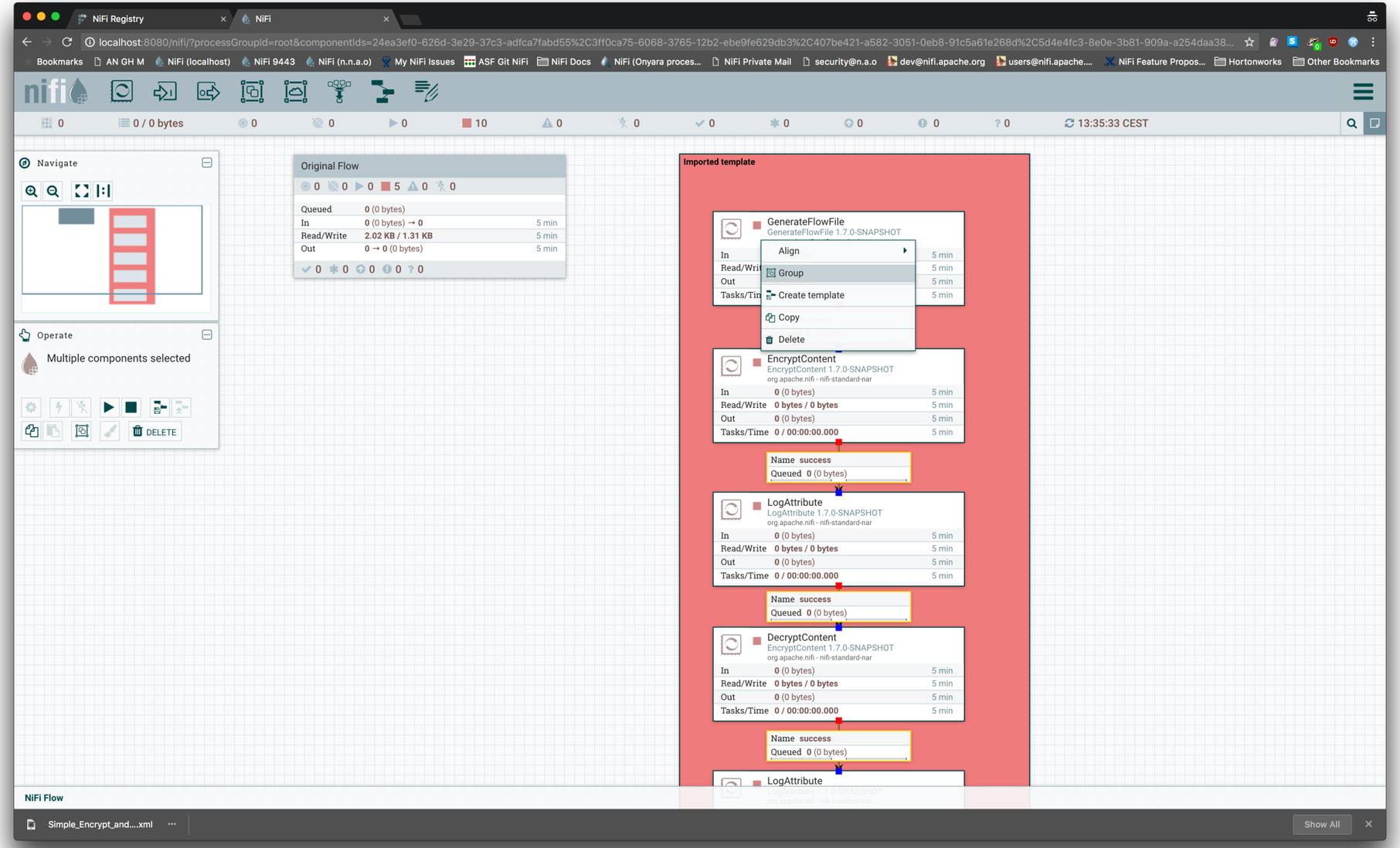


Connect to NiFi

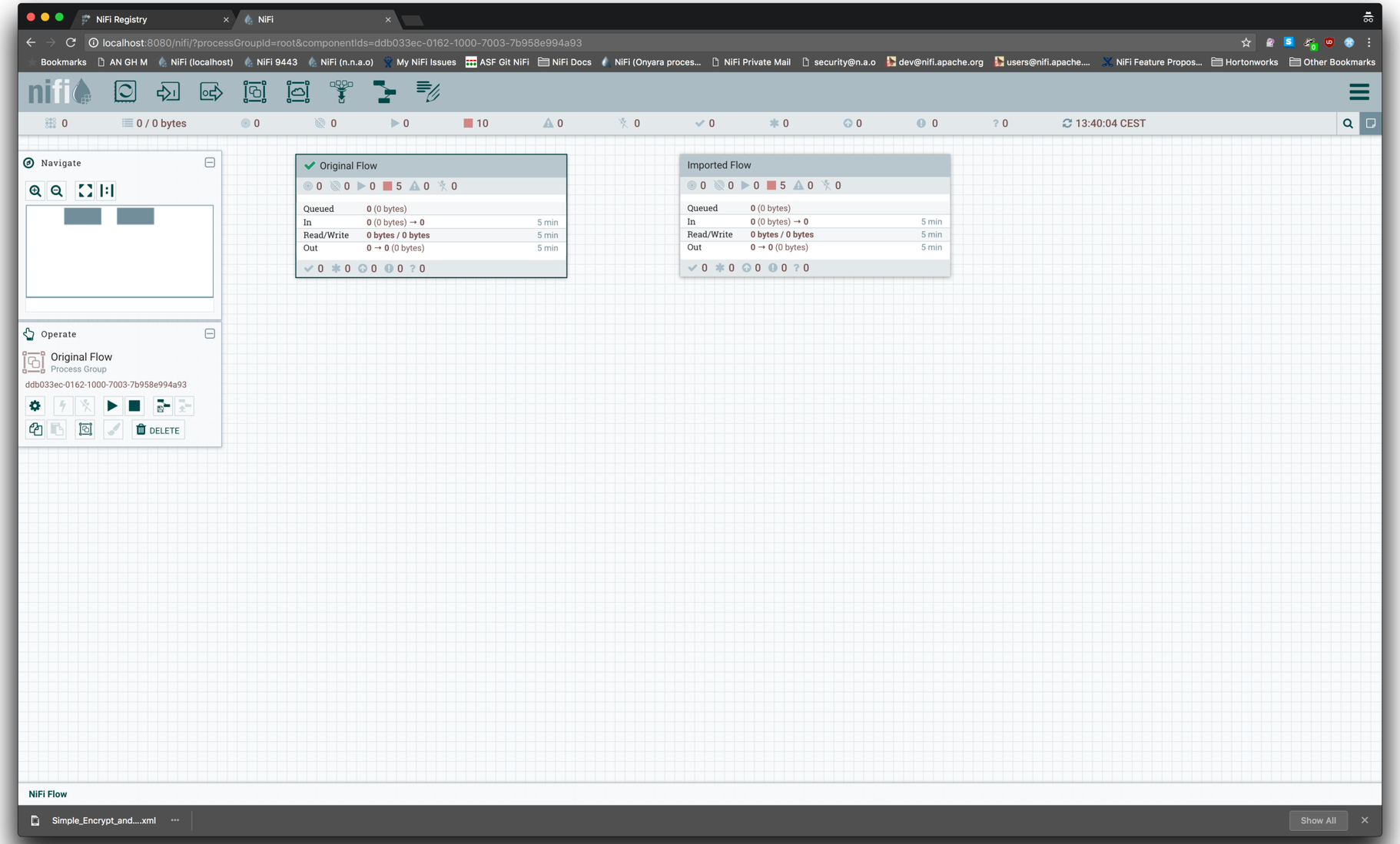


HORTONWORKS

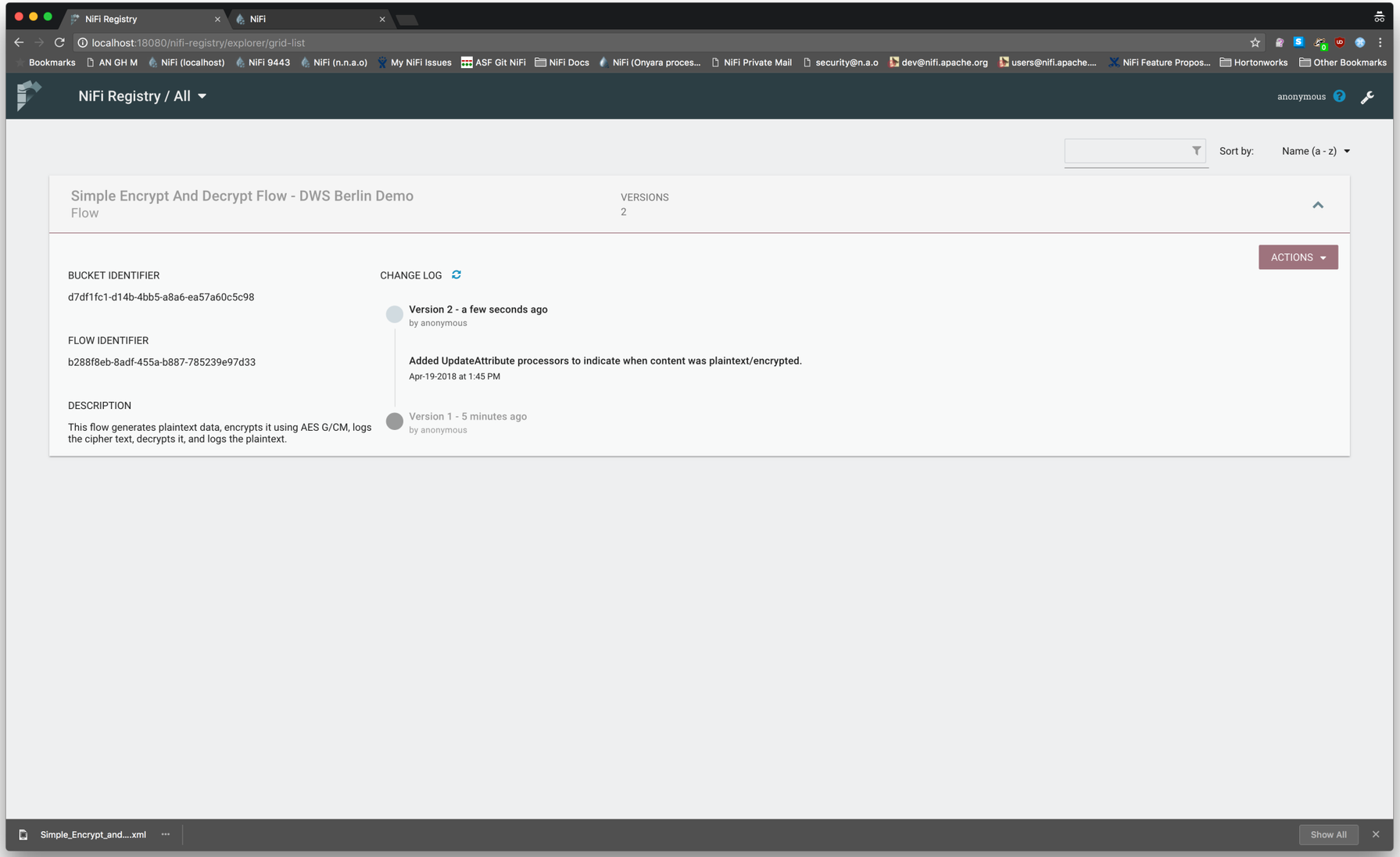
Create Process Group



Commit Version

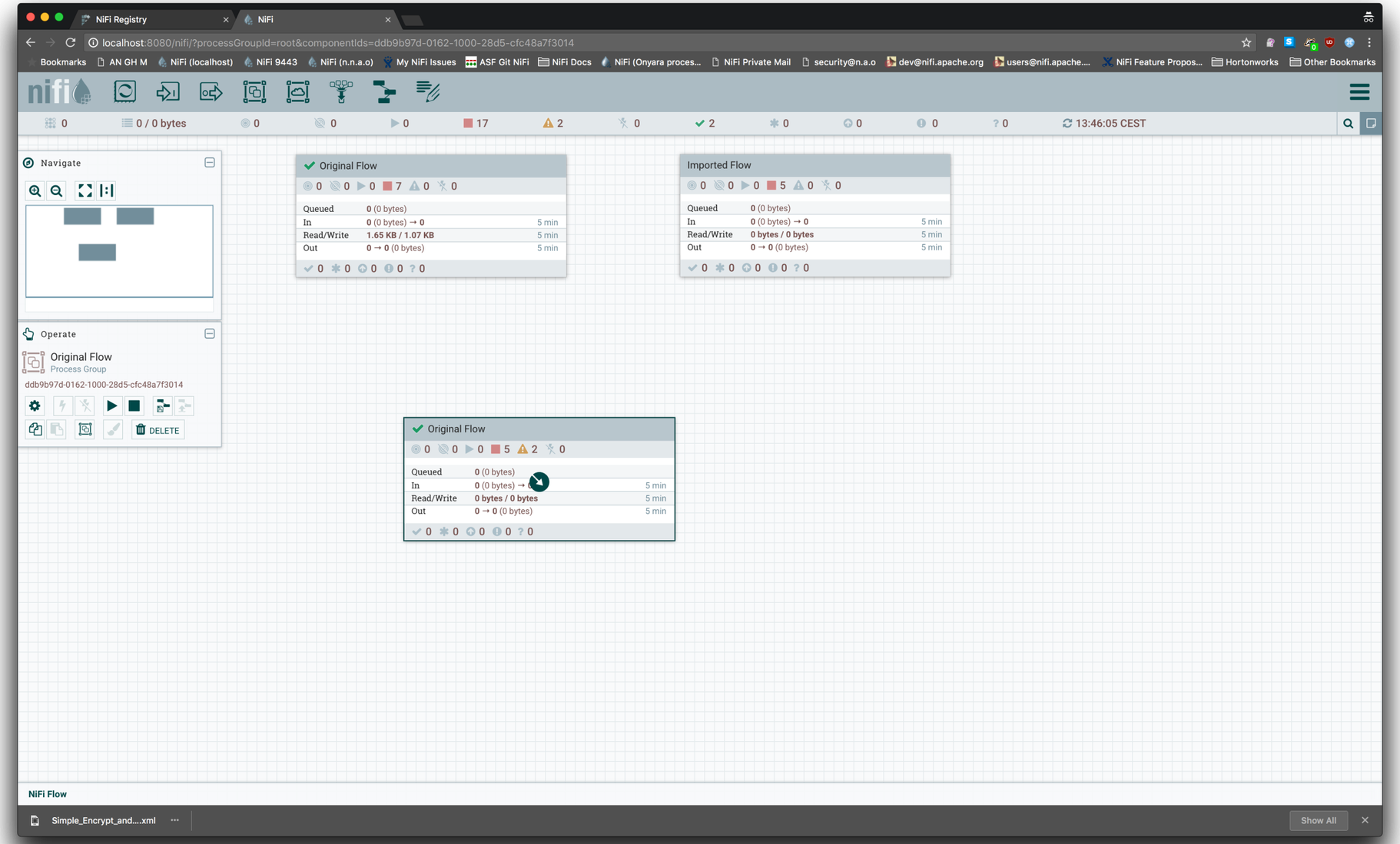


View flow in Registry

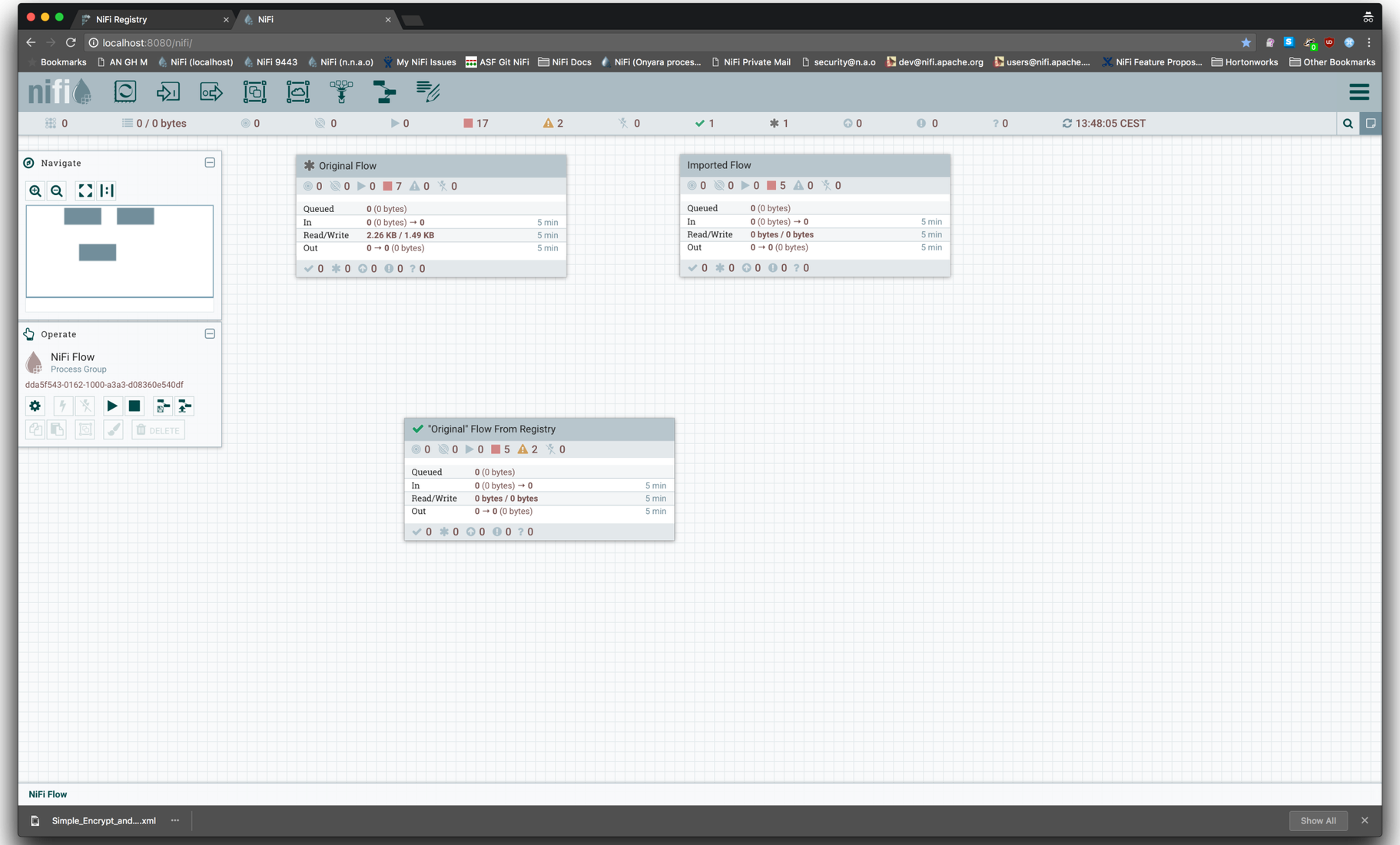


HORTONWORKS

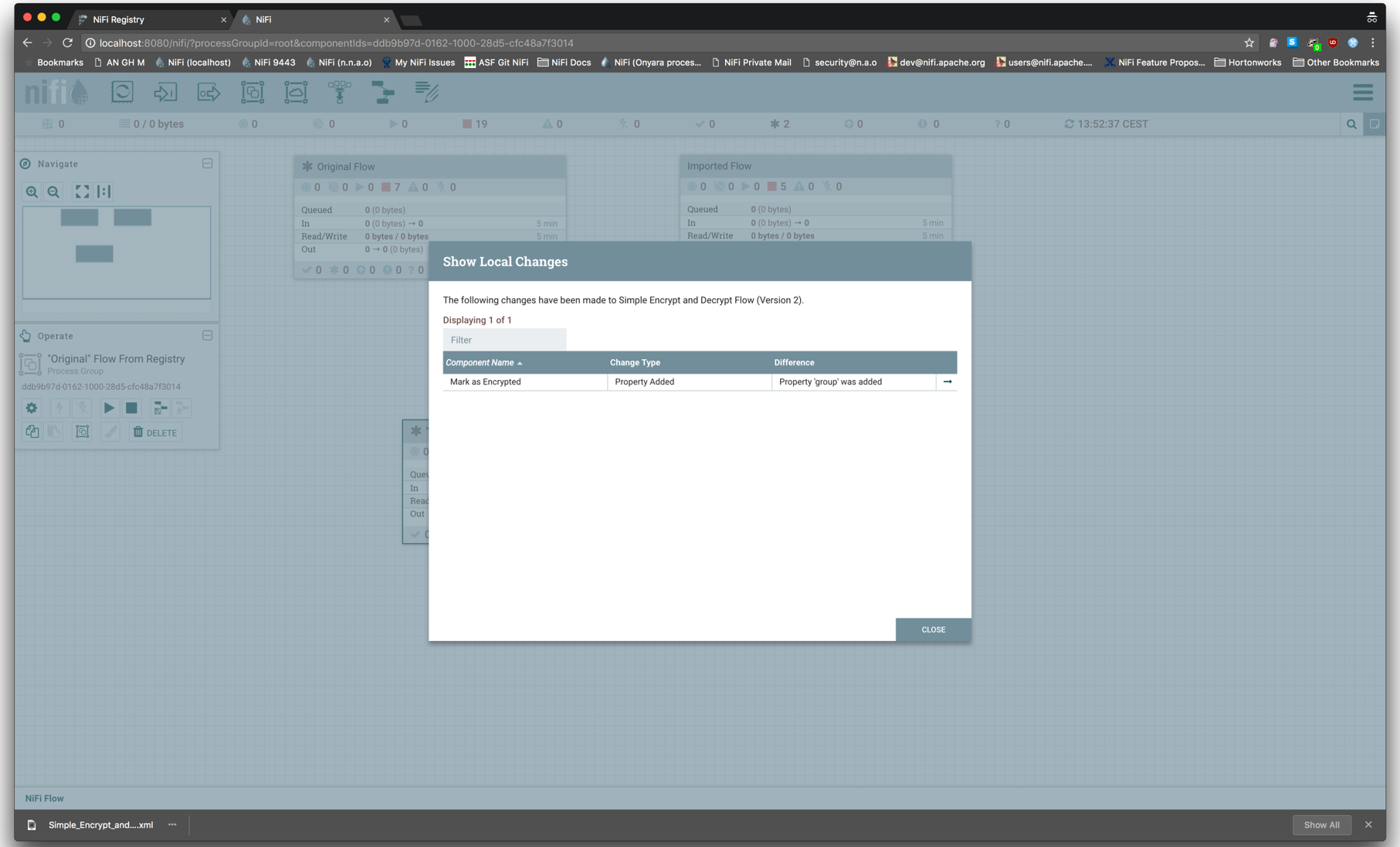
Import new instance into NiFi



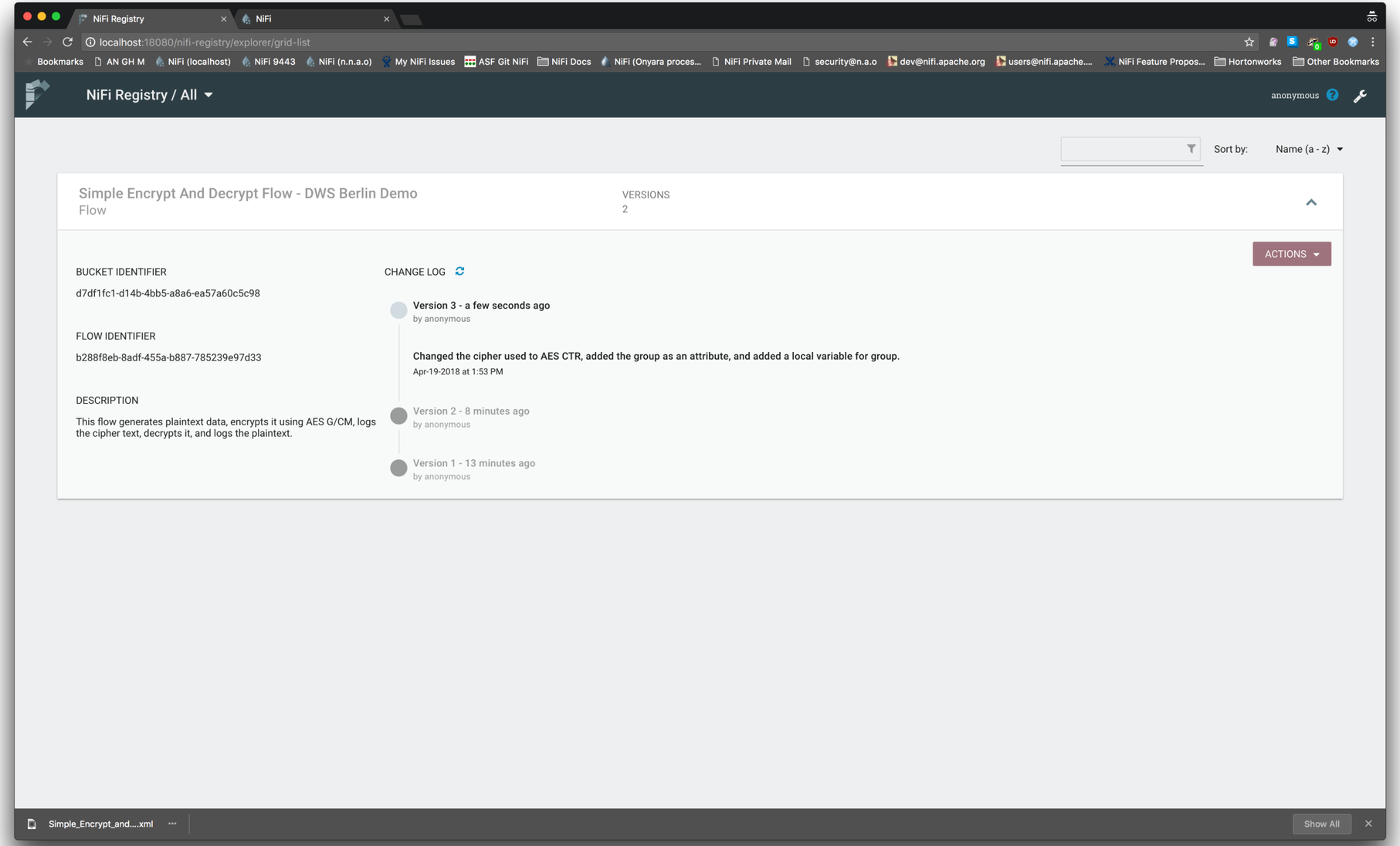
Modify the original flow



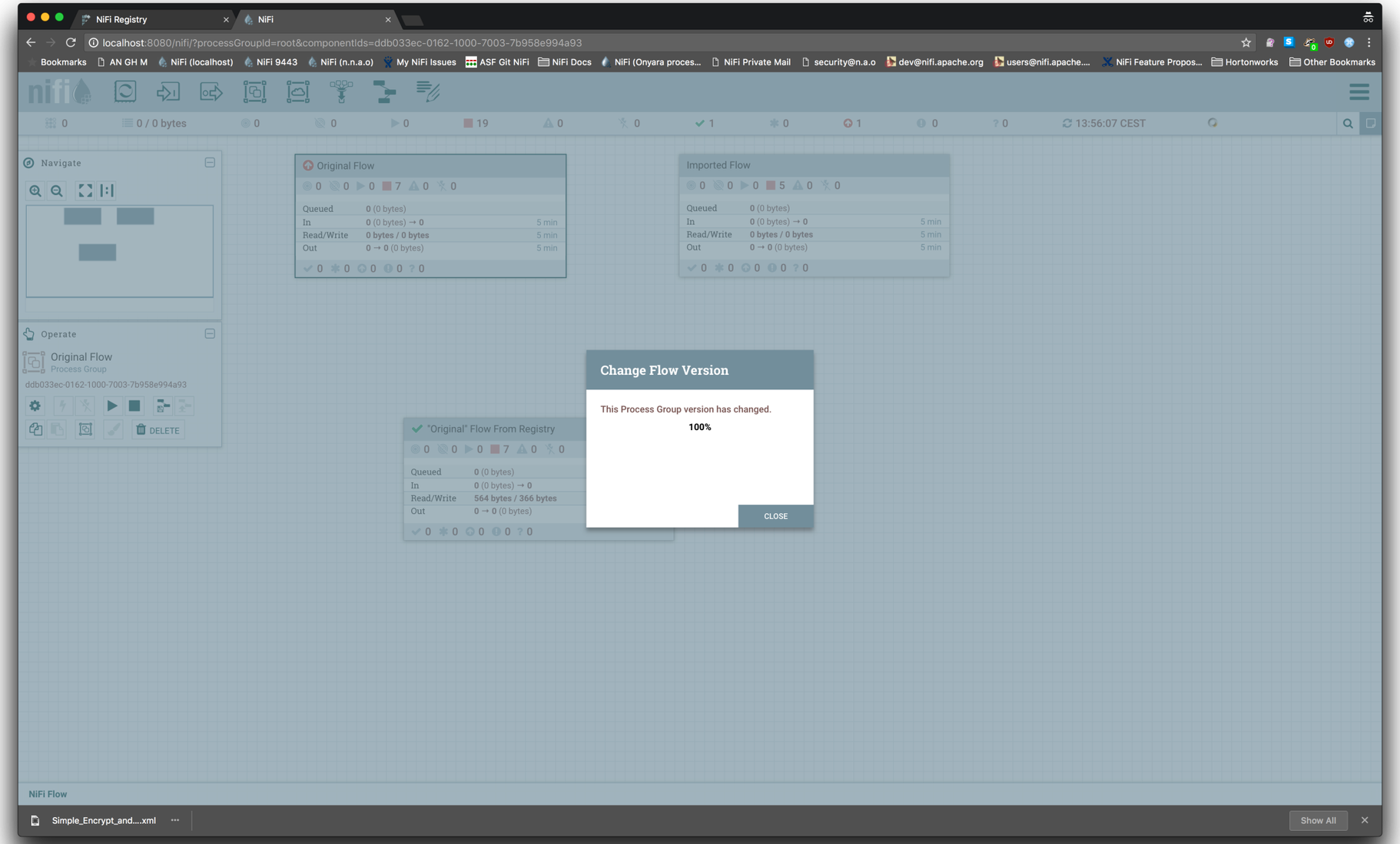
See local changes before committing



Commit



Update new instance from Registry



Complementary Tools

HORTONWORKS

NiFi Toolkit

- TLS Toolkit
  - Generates, signs, and packages keys and certificates for NiFi services (node/cluster, clients)
- Encrypt Config
  - Protects sensitive configuration values like passwords
- **CLI**
  - Interacts with NiFi & NiFi Registry to operate on flows

The default output type in interactive mode is `simple`, and the default output type in standalone mode is `json`.

Example of simple output for list-buckets:

```
#> registry list-buckets -ot simple

My Bucket - 3c7b7467-0012-4d8f-a918-6aa42b6b9d39
```

Example of json output for list-buckets:

```
#> registry list-buckets -ot json
[ {
  "identifier" : "3c7b7467-0012-4d8f-a918-6aa42b6b9d39",
  "name" : "My Bucket",
  "createdTimestamp" : 1516718733854,
  "permissions" : {
    "canRead" : true,
    "canWrite" : true,
    "canDelete" : true
  },
  "link" : {
    "params" : {
      "rel" : "self"
    },
    "href" : "buckets/3c7b7467-0012-4d8f-a918-6aa42b6b9d39"
  }
} ]
```

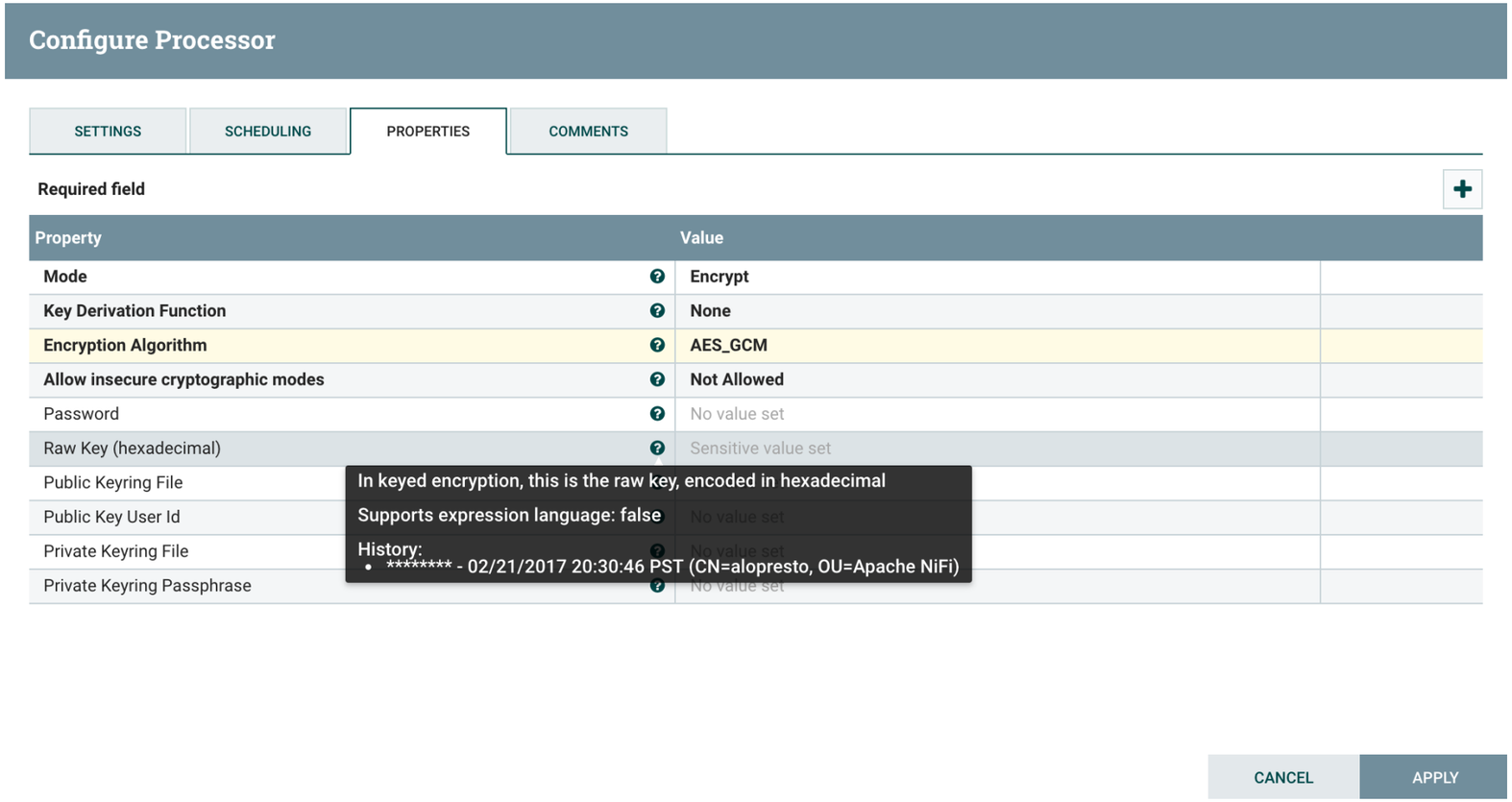

HORTONWORKS

Security

HORTONWORKS®

Security

- Secure the instance
  - Sensitive properties
  - HTTPS
  - Authentication & Authorization
- Secure the configuration
  - Encrypt the configs
- Secure the data
  - `EncryptContent`
  - Encrypted repositories



Sensitive Component Properties

- NiFi encrypts all sensitive component properties (*database password, FTP password, etc.*) with a configurable algorithm
  - Default is `PBEWITHMD5AND256BITAES-CBC-OPENSSL`
- If no key material is provided by the admin, a default is used
  - TBC in 2.0.0
- **Populate this with a random, unique value when deploying NiFi** `nifi.sensitive.props.key=thisIsBadButWayBetterThanNothing`

```
148  # security properties #
149  nifi.sensitive.props.key=
150  nifi.sensitive.props.key.protected=
151  nifi.sensitive.props.algorithm=PBEWITHMD5AND256BITAES-CBC-OPENSSL
152  nifi.sensitive.props.provider=BC
153  nifi.sensitive.props.additional.keys=
```

```
/**
 * This constructor creates an encryptor using <em>Password-Based Encryption</em> (PBE). The <em>key</em> value is
  the direct value provided in <code>nifi.sensitive.props.key</code> in
 * <code>nifi.properties</code>, which is a <em>PASSWORD</em> rather than a <em>KEY</em>, but is named such for
  backward/legacy logical compatibility throughout the rest of the codebase.
 * <p>
 * For actual raw key provision, see {@link #StringEncryptor(String, String, byte[])}.
 *
 * @param algorithm the PBE cipher algorithm ({@link EncryptionMethod#algorithm})
 * @param provider  the JCA Security provider ({@link EncryptionMethod#provider})
 * @param key       the UTF-8 characters from nifi.properties -- nifi.sensitive.props.key
 */
public StringEncryptor(final String algorithm, final String provider, final String key) {
    this.algorithm = algorithm;
    this.provider = provider;
    this.key = null;
    this.password = new PBEKeySpec(key == null
            ? DEFAULT_SENSITIVE_PROPS_KEY.toCharArray()
            : key.toCharArray());
    initialize();
}
```


HORTONWORKS

First Step After Uncompressing NiFi

- Change the default key material
- Input to KDF to generate the symmetric key used for sensitive processor property value encryption

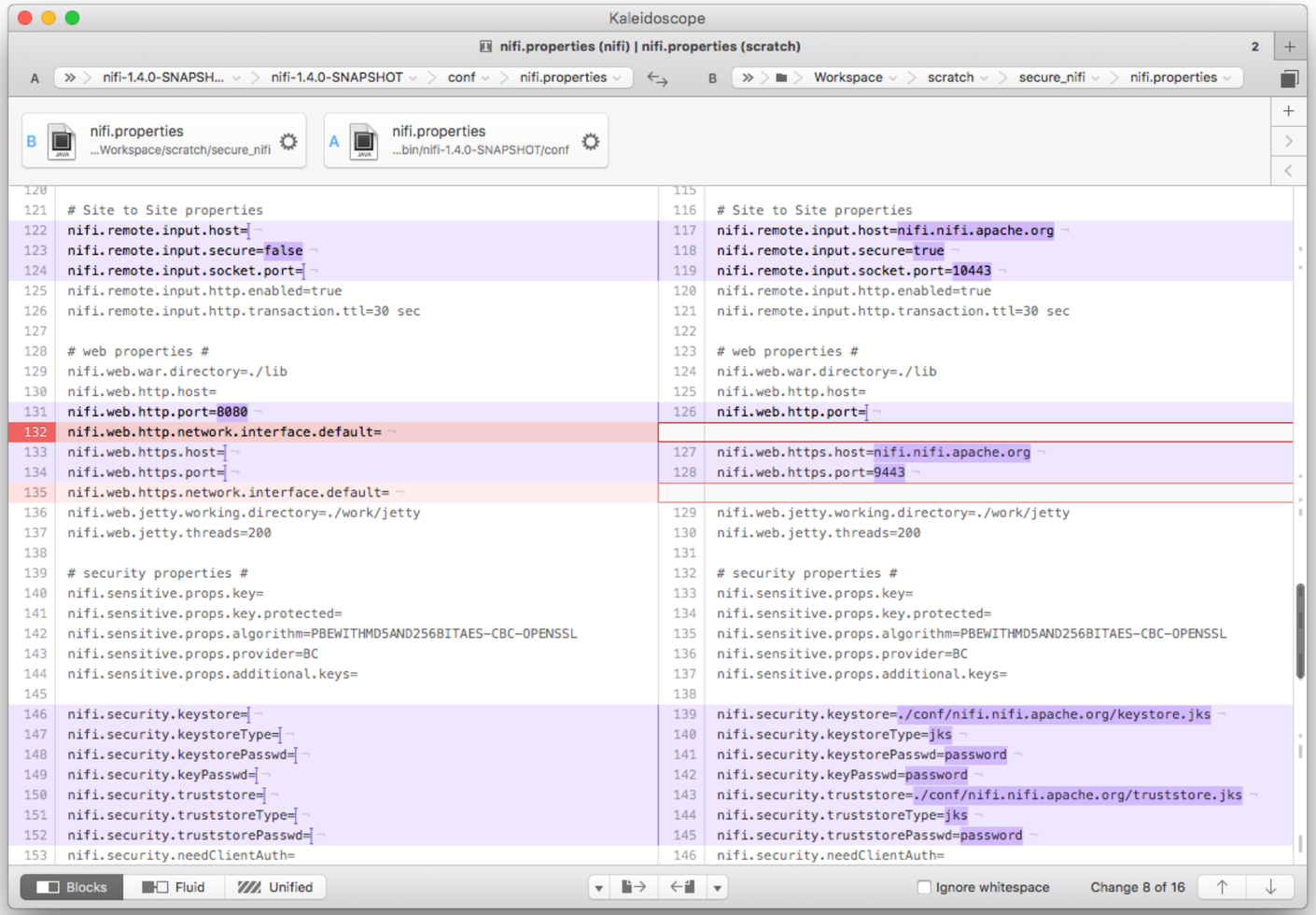
```
139  # security properties #
140  nifi.sensitive.props.key=This is not the default NiFi key material.
141  nifi.sensitive.props.key.protected=
142  nifi.sensitive.props.algorithm=PBEWITHMD5AND256BITAES-CBC-OPENSSL
143  nifi.sensitive.props.provider=BC
144  nifi.sensitive.props.additional.keys=
145
```

```
125          <name>Encryption Algorithm</name>
126          <value>AES_GCM</value>
127        </property>
128        <property>
129          <name>allow-weak-crypto</name>
130          <value>not-allowed</value>
131        </property>
132        <property>
133          <name>Password</name>
134        </property>
135        <property>
136          <name>raw-key-hex</name>
137          <value>enc{9D6F3EE98D4D04CD7875ED77E9EDBC30526820CDD2C764C45DCBBC0CBBC839FB2D80B
             B0477779BFB49DD8FC262E05C9F2EB6BDED343BA8D116395BC4C34B4DA41C3978DEC4CA939D08552
             A8DE9E089481959180E3DEAE537BE1BA278FA8F90C9}</value>
138        </property>
```


HORTONWORKS

Configure HTTPS

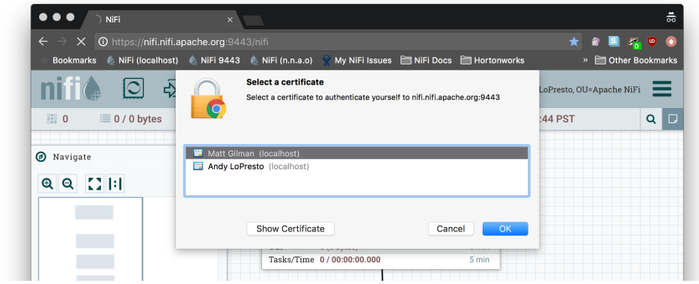
- Deploying a keystore & truststore secures the network connections and enables authentication
  - Keystore = “who am I?”
  - Truststore = “who should I listen to?”

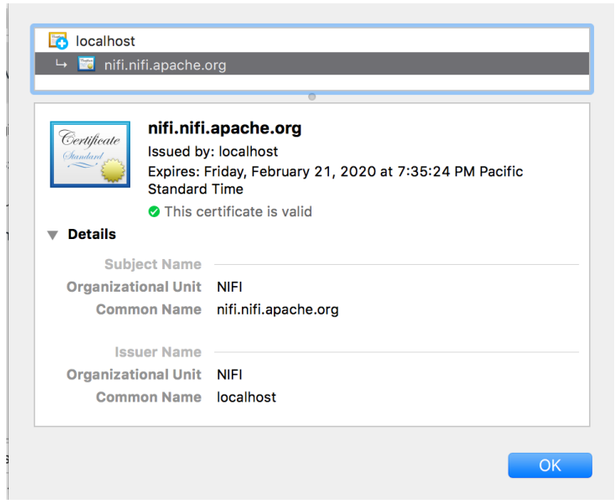


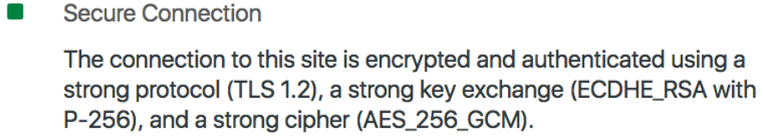
HORTONWORKS

Enable TLS (SSL)

- Three ways to do this
  - Self-signed certificate generation (**manual / hard**)
  - Existing enterprise certificates (**manual / hard**)
  - NiFi TLS Toolkit (**automatic / easy**)







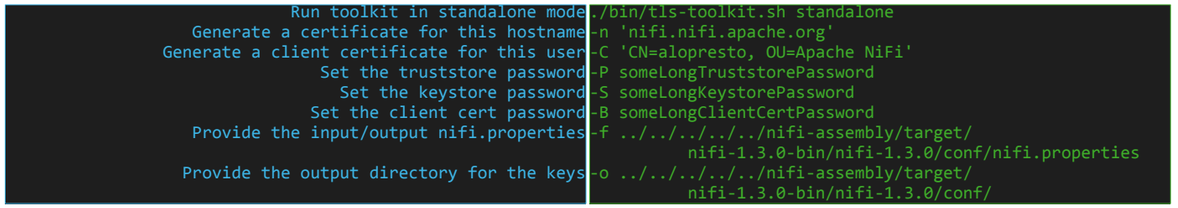
NiFi TLS Settings

- HDF 3.0 (NiFi 1.2.0+) API/UI supports only TLSv1.2+
  - Strong cipher suites
  - No SHA-1*
  - Prevents downgrade attacks
- Can still support legacy protocols for external service ingest

HORTONWORKS

NiFi TLS Toolkit

- Can run in *standalone* or *client/server* mode
- Automatically generates NiFi Certificate Authority (CA)
- Generates certificates for *n* nodes
- Signed by CA via CSR with shared-secret token
- Builds keystore & truststore
- Populates passwords in `nifi.properties`
- Generates client certificate for authentication
- Strong random passwords generated if not provided




```
hw12203:nifi-toolkit-1.3.0 (master) alopresto
🔓 15s @ 16:51:23 $ ./bin/tls-toolkit.sh standalone -n 'nifi.nifi.apache.org'
-C 'CN=alopresto, OU=Apache NiFi' -P someLongTruststorePassword -S
someLongKeystorePassword -B someLongClientCertPassword -f .../conf/
nifi.properties -o .../conf/
o.a.n.t: CommandLine: Using .../conf/nifi.properties as template.
o.a.n.t: Running standalone certificate generation with output directory .../
conf
o.a.n.t: Using existing CA certificate .../conf/nifi-cert.pem and key .../conf/
nifi-key.key
o.a.n.t: Writing new ssl configuration to .../conf/nifi.nifi.apache.org
o.a.n.t: Successfully generated TLS configuration for nifi.nifi.apache.org 1 in
.../conf/nifi.nifi.apache.org
o.a.n.t: Generating new client certificate .../conf/
CN=alopresto_OU=Apache_NiFi.p12
o.a.n.t: Successfully generated client certificate .../conf/
CN=alopresto_OU=Apache_NiFi.p12
o.a.n.t: tls-toolkit standalone completed successfully
hw12203:nifi-toolkit-1.3.0 (master) alopresto
🔓 6s @ 16:51:29 $
```


HORTONWORKS

Standalone Examples

Create 4 sets of keystore, truststore, nifi.properties for localhost along with a client certificate with the given DN:

```
bin/tls-toolkit.sh standalone -n 'localhost(4)' -C 'CN=username,OU=NIFI'
```

Create keystore, truststore, nifi.properties for 10 NiFi hostnames in each of 4 subdomains:

```
bin/tls-toolkit.sh standalone -n 'nifi[01-10].subdomain[1-4].domain'
```

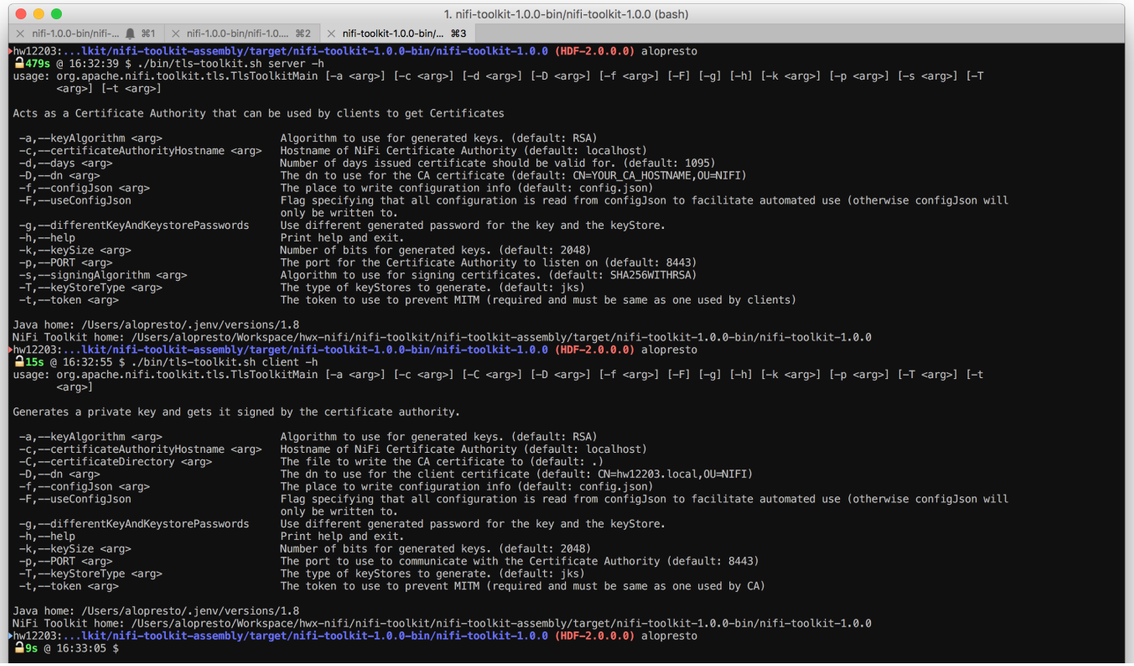
Create 2 sets of keystore, truststore, nifi.properties for 10 NiFi hostnames in each of 4 subdomains along with a client certificate with the given DN:

```
bin/tls-toolkit.sh standalone -n 'nifi[01-10].subdomain[1-4].domain(2)' -C 'CN=username,OU=NIFI'
```


HORTONWORKS

Client/Server Mode

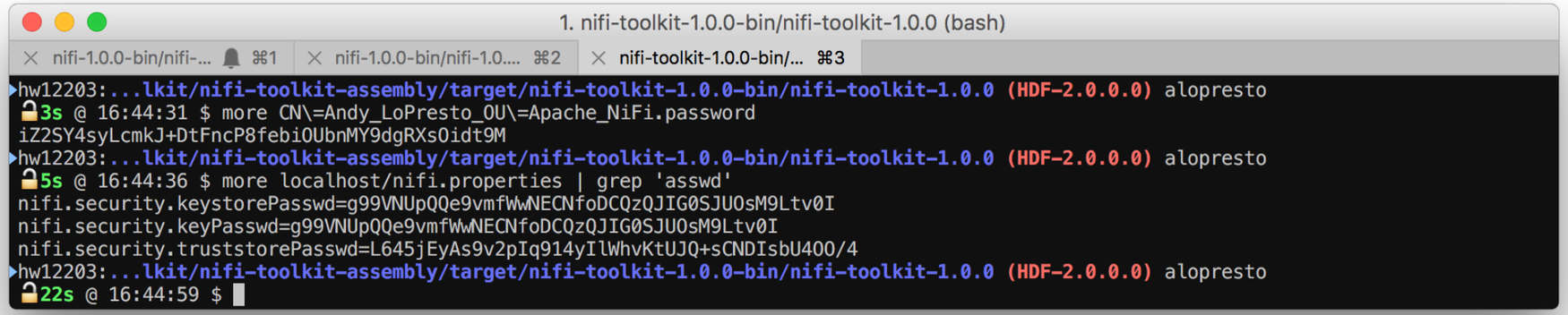
- Long running CA
- Multiple clients can generate a certificate locally and send CSR to CA to be signed



HORTONWORKS

Security Mechanisms

- Private key is not communicated from client to CA; CSR is issued and signed
- Token and HMAC/SHA-256 is used to prevent MitM
- Default passwords generated for keystores, certificates are 32 bytes of secure random, Base64-encoded (~43 chars)



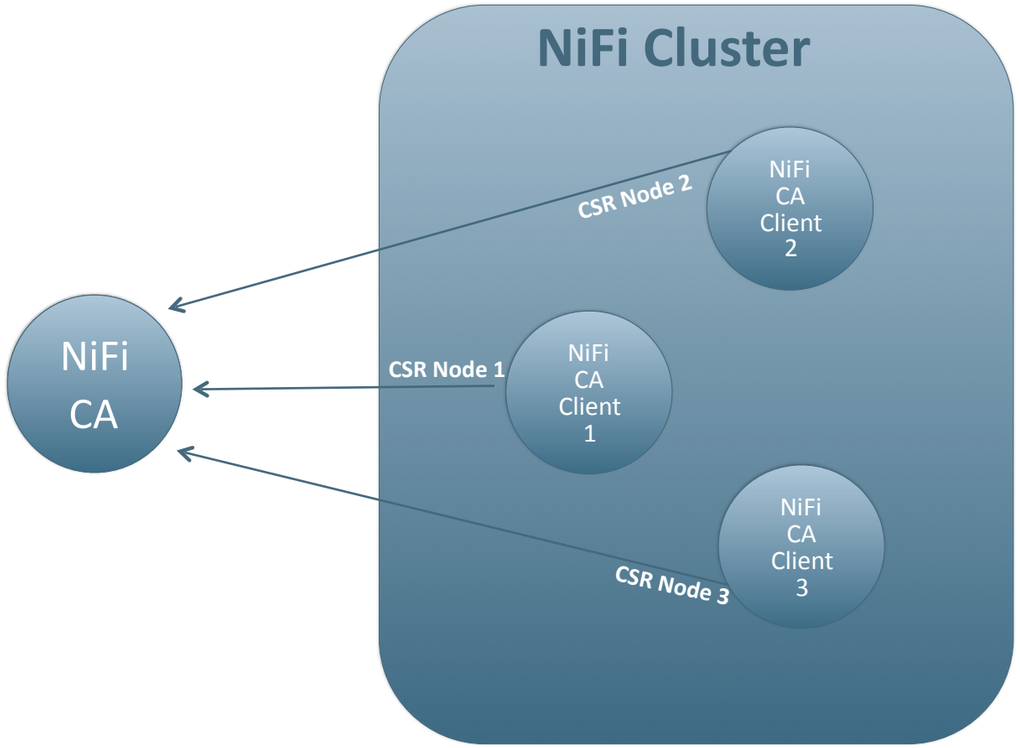
HORTONWORKS

NiFi Certificate Authority - Background

- Driving Factors
  - NiFi nodes in a secured cluster need a way to trust and securely communicate with each other (in addition to the user)
  - Generating and deploying trusted certificates manually for nodes can be very time consuming (especially in large clusters)
  - Ambari did not previously have a method to centralize certificate generation & distribution
- Quick Facts
  - Uses NiFi TLS Toolkit – Client / Server Mode
  - Uses a “Shared Secret” scheme to establish trust
  - Installation is NOT required
  - Can be managed similarly to other components (offered as additional option when adding NiFi service)
  - Can be installed on separate host from NiFi nodes
  - **1.8.0+** can use external CA certificate to sign certs (*rather than NiFi-generated only*)

NiFi Certificate Authority – Communication Workflow

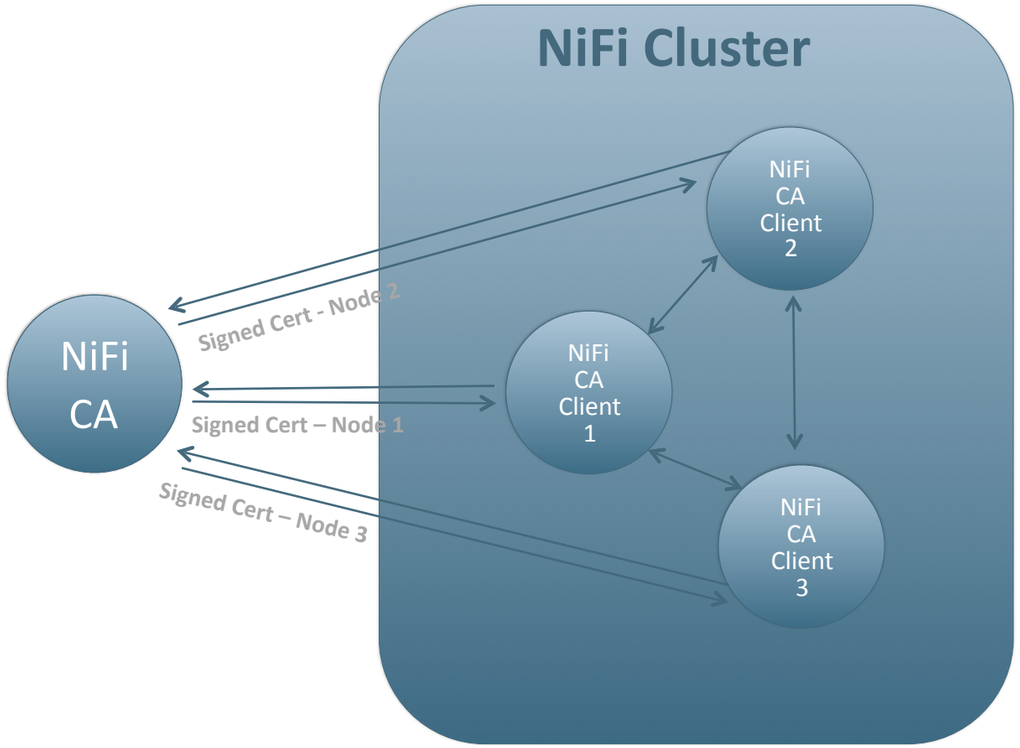
1. CA generates a self signed certificate on startup (default port 10443)
2. NiFi CA clients (running on NiFi node hosts), on startup, generate a key pair and a Certificate Signing Request (CSR). The CSR contains the identity of the client, the public key (from the key pair), and a hash generated using the **NiFi CA Token*** and the public key's signature
3. CA clients send the CSR to the CA
4. CA verifies that the CSR is from a trusted NiFi CA Client by generating a hash using its copy of the token with the public key signature in the CSR and compares with the hash included in the request



*Token is provided during NiFi configuration and should be **strong and protected**

NiFi Certificate Authority – Communication Workflow

5. When CA has confirmed node is trusted, it will sign the CSR and generate a hash of its public key fingerprint using the same token. Both are sent back to the node
6. Once the client receives the response from the CA, to verify it is the trusted CA, it will obtain the CA's public key (from the TLS connection) and generate a hash using the secret token and the CA's public key signature
7. Upon validation the client will accept the signed certificate. Ambari will then update NiFi properties settings for the newly signed cert to start the NiFi node in SSL mode



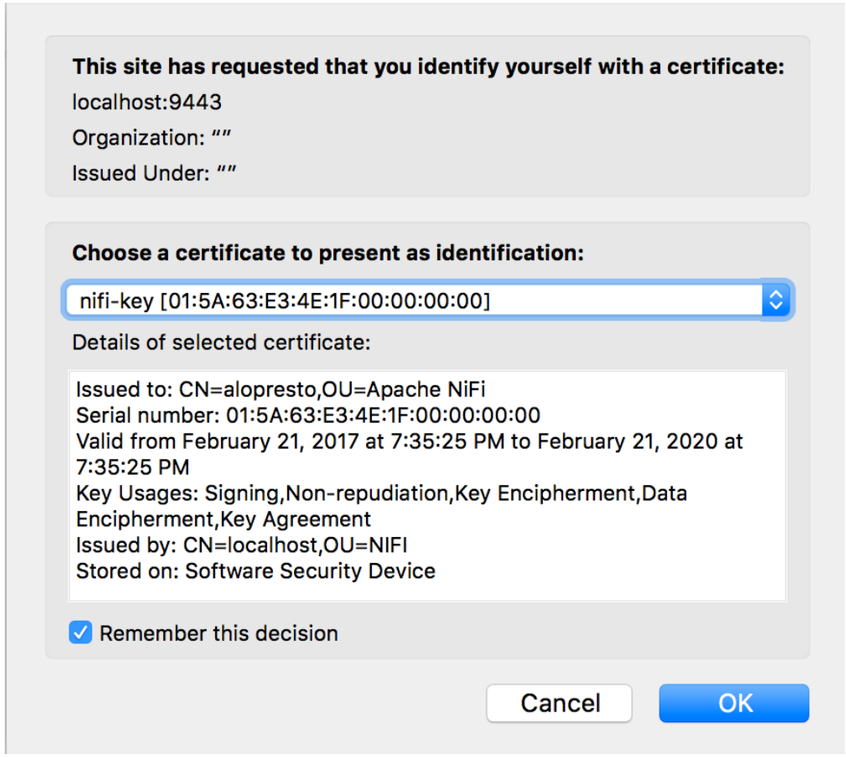
Authentication

- Supported authentication mechanisms
  - Client certificate
  - LDAP
  - Kerberos
  - OpenID Connect
  - KnoxSSO
  - OAuth 2.0 *under review

HORTONWORKS

Client Certificate Authentication

- Always available (original auth mechanism for NiFi)
- Client certificate signed by a CA in the server truststore
- Issuer DN becomes username
- Authenticated on every request



HORTONWORKS

LDAP Authentication

- Configured via `login-identity-providers.xml`
- Supports LDAP w/ StartTLS and LDAPS
- After initial authentication, encrypted JWT is issued via response and placed in local storage
- JWT substituted for LDAP credentials on following requests

```
nifi.security.keystore=./conf/keystore.jks
nifi.security.keystoreType=jks
nifi.security.keystorePasswd=FcD7b/eywqwicpl4ibC2PvvFm2vlz0OBZf00blrBd9c
nifi.security.keyPasswd=FcD7b/eywqwicpl4ibC2PvvFm2vlz0OBZf00blrBd9c
nifi.security.truststore=./conf/truststore.jks
nifi.security.truststoreType=jks
nifi.security.truststorePasswd=iQ5ZqLZtH2fskrUfXDcpFc/IbVRgidbbvaEvHz5Q3BE
nifi.security.needClientAuth=
nifi.security.user.authorizer=file-provider
nifi.security.user.login.identity.provider=ldap-provider
nifi.security.ocsp.responder.url=
nifi.security.ocsp.responder.certificate=
```

```
<provider>
    <identifier>ldap-provider</identifier>
    <class>org.apache.nifi.ldap.LdapProvider</class>
    <property name="Authentication Strategy">START_TLS</property>

    <property name="Manager DN”>CN=admin, OU=NiFi</property>
    <property name="Manager Password”>jkhsd7s0bdl1dk3asdk113</property>

    <property name="TLS - Keystore”>./conf/keystore.jks</property>
    <property name="TLS - Keystore Password">FcD7b/
eywqwicpl4ibC2PvvFm2vlz0OBZf00blrBd9c</property>
    <property name="TLS - Keystore Type">JKS</property>
    <property name="TLS - Truststore”>./conf/truststore.jks</property>
    <property name="TLS - Truststore Password">iQ5ZqLZtH2fskrUfXDcpFc/
IbVRgidbbvaEvHz5Q3BE</property>
    <property name="TLS - Truststore Type">JKS</property>
    <property name="TLS - Client Auth">WANT</property>
    <property name="TLS - Protocol">TLS</property>
    <property name="TLS - Shutdown Gracefully"></property>

    <property name=“Url”>ldap://ldap.nifi.apache.org:389</property>
    <property name="User Search Base”>dc=nifi,dc=apache,dc=org</property>
    <property name="User Search Filter”>cn={0}</property>
...
</provider>
```

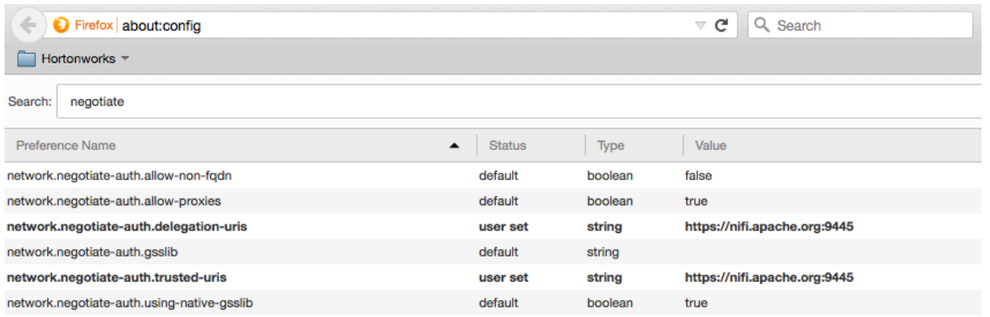

HORTONWORKS

Kerberos Authentication

- Configured via `login-identity-providers.xml`
- Supports SPNEGO via GSS-API and form login as backup
  - *May be removed in future release*
- After initial authentication, encrypted JWT is issued via response and placed in local storage
- JWT substituted for Kerberos credentials on following requests

```
klist
Credentials cache: API:5D669ADD-9028-441F-A6E0-C2C78DA04CC5
        Principal: alopresto@NIFI.APACHE.ORG

  Issued                Expires               Principal
Jun 2 10:12:29 2017  Jun 22 20:12:27 2017  krbtgt/NIFI.APACHE.ORG@NIFI.APACHE.ORG
```

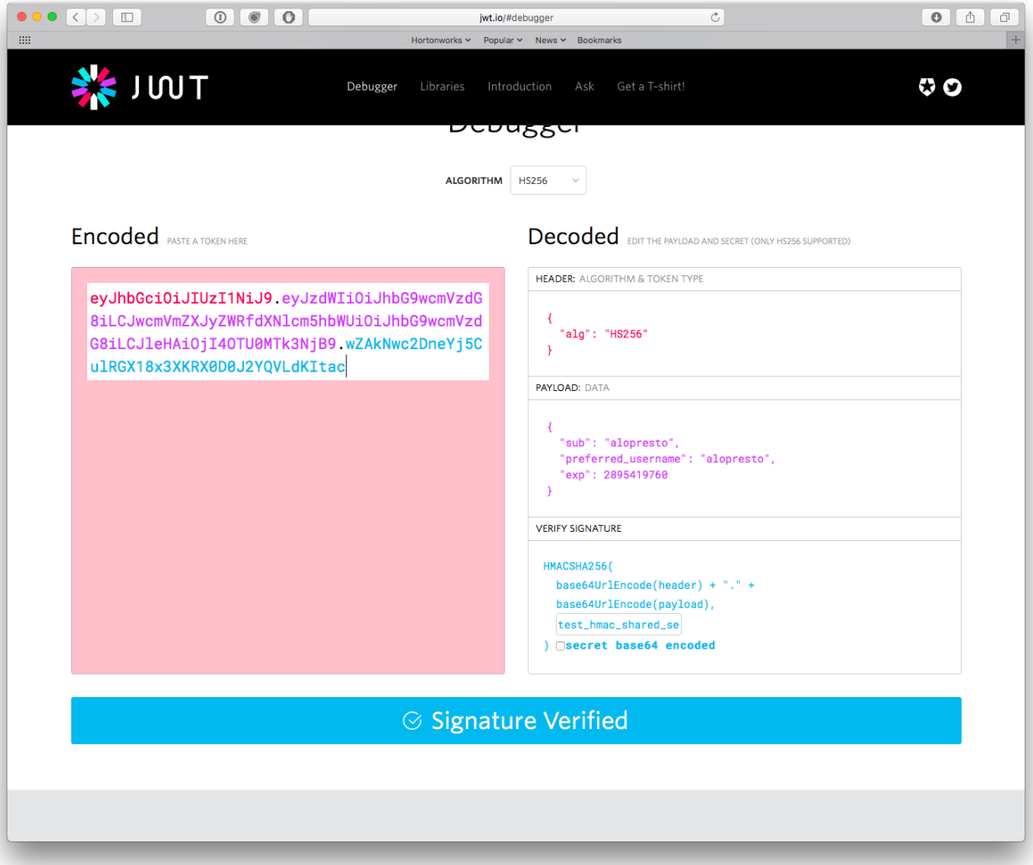
```
nifi.security.user.login.identity.provider=kerberos-provider

nifi.kerberos.krb5.file=/path/to/docker-kdc/krb5.conf

nifi.kerberos.spnego.principal=HTTP/nifi.apache.org@NIFI.APACHE.ORG
nifi.kerberos.spnego.keytab.location=/path/to/docker-kdc/krb5.keytab
nifi.kerberos.spnego.authentication.expiration=12 hours
```



HORTONWORKS

JWT - JSON Web Tokens

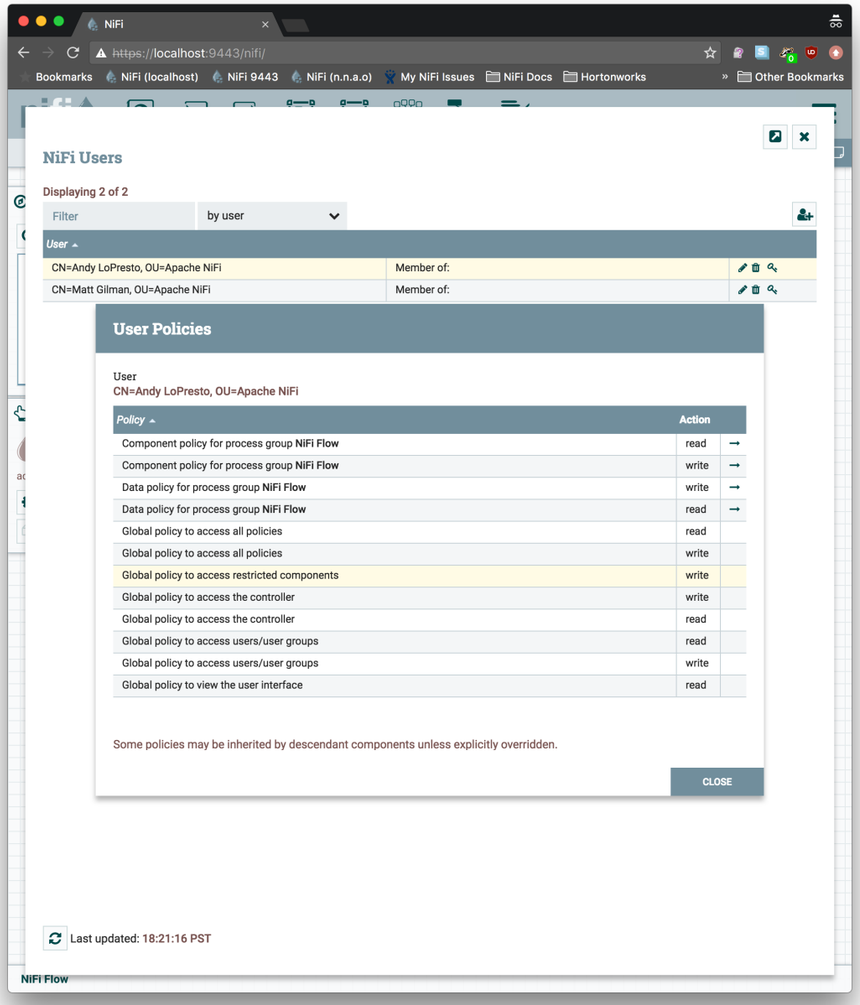
- Encrypted & signed token containing data passed as Base64-encoded `Authorization` header in HTTP request
- Allows for stateless web applications
- Encrypted with per-user key stored by NiFi



HORTONWORKS

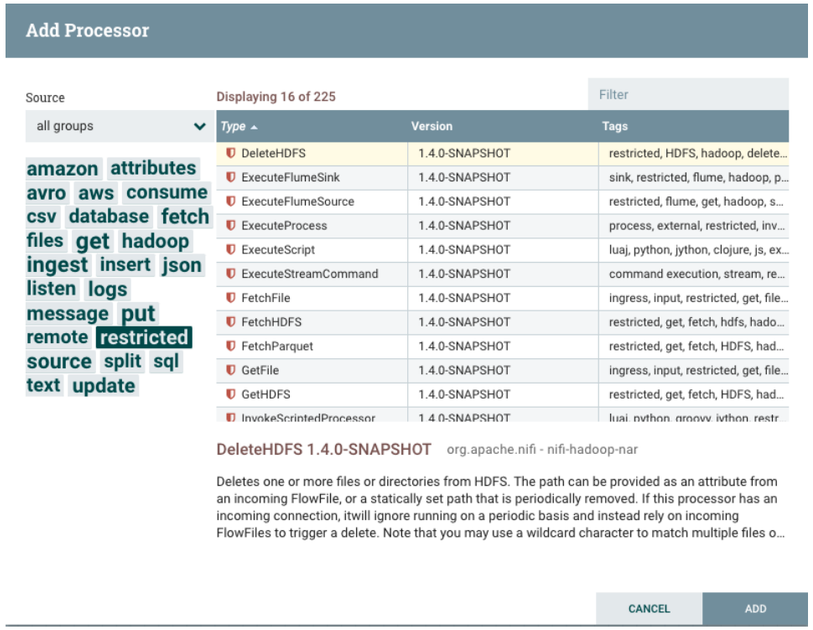
Access Policies Enhancements

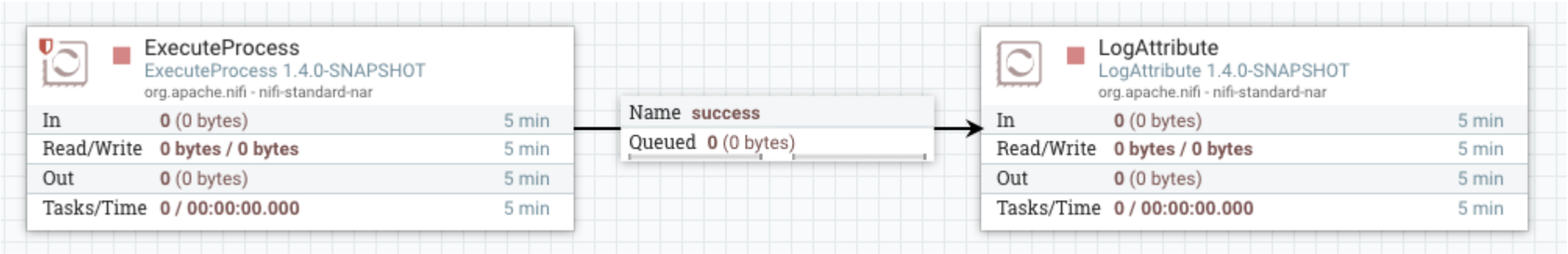
- Cleaner UI and more convenient for common tasks
  - Collect all user policies in single view (policy-centric vs. user-centric)
  - Fewer clicks to add policies



Restricted Components

- Controller services, reporting tasks, and processors which could maliciously affect the host system are now **RESTRICTED**
- Special annotation and access policy required





Secure Configuration

- Previously, sensitive configuration values were exposed in **plaintext**
  - Recommendation for POSIX file permissions to prevent unauthorized access
- Pluggable architecture which allows `NiFiProperties` to be secured until application load
- Passwords protected by default; arbitrary values can be protected via `nifi.sensitive.props.additional.keys`


```
# security properties #
nifi.sensitive.props.key=n2z+tTTbHuZ4V4V2||uWhdasyDXD4ZG2lMAes/vqh6u4vaz4xgL4aEbF4Y/dXevqk3ulRcOwf1vc4RDQ
nifi.sensitive.props.key.protected=aes/gcm/256
nifi.sensitive.props.algorithm=PBEWITHMD5AND256BITAES-CBC-OPENSSL
nifi.sensitive.props.provider=BC
nifi.sensitive.props.additional.keys=

nifi.security.keystore=/path/to/keystore.jks
nifi.security.keystoreType=JKS
nifi.security.keystorePasswd=oBjT92hIGRElIGOh||MZ6uYuWNBrOA6usq/Jt3DaD2e4otNirZDytac/w/KFe0HOkrJR03ycbo
nifi.security.keystorePasswd.protected=aes/gcm/256
nifi.security.keyPasswd=ac/BaE35SL/esLiJ||+ULRvRLYdIDA2VqpE0eQXDEMjaLBMG2kbKOdOwBk/hGebDKlVg
nifi.security.keyPasswd.protected=aes/gcm/256
nifi.security.truststore=
nifi.security.truststoreType=
nifi.security.truststorePasswd=/X/RSlNr2QCJ1Kwe||dENJevX5P61ix+97airrtoBQovasMFS6DG6fHbX+SZtw2VAMllSSnDeT97Q
nifi.security.truststorePasswd.protected=aes/gcm/256
nifi.security.needClientAuth=
nifi.security.user.authorizer=
```


HORTONWORKS

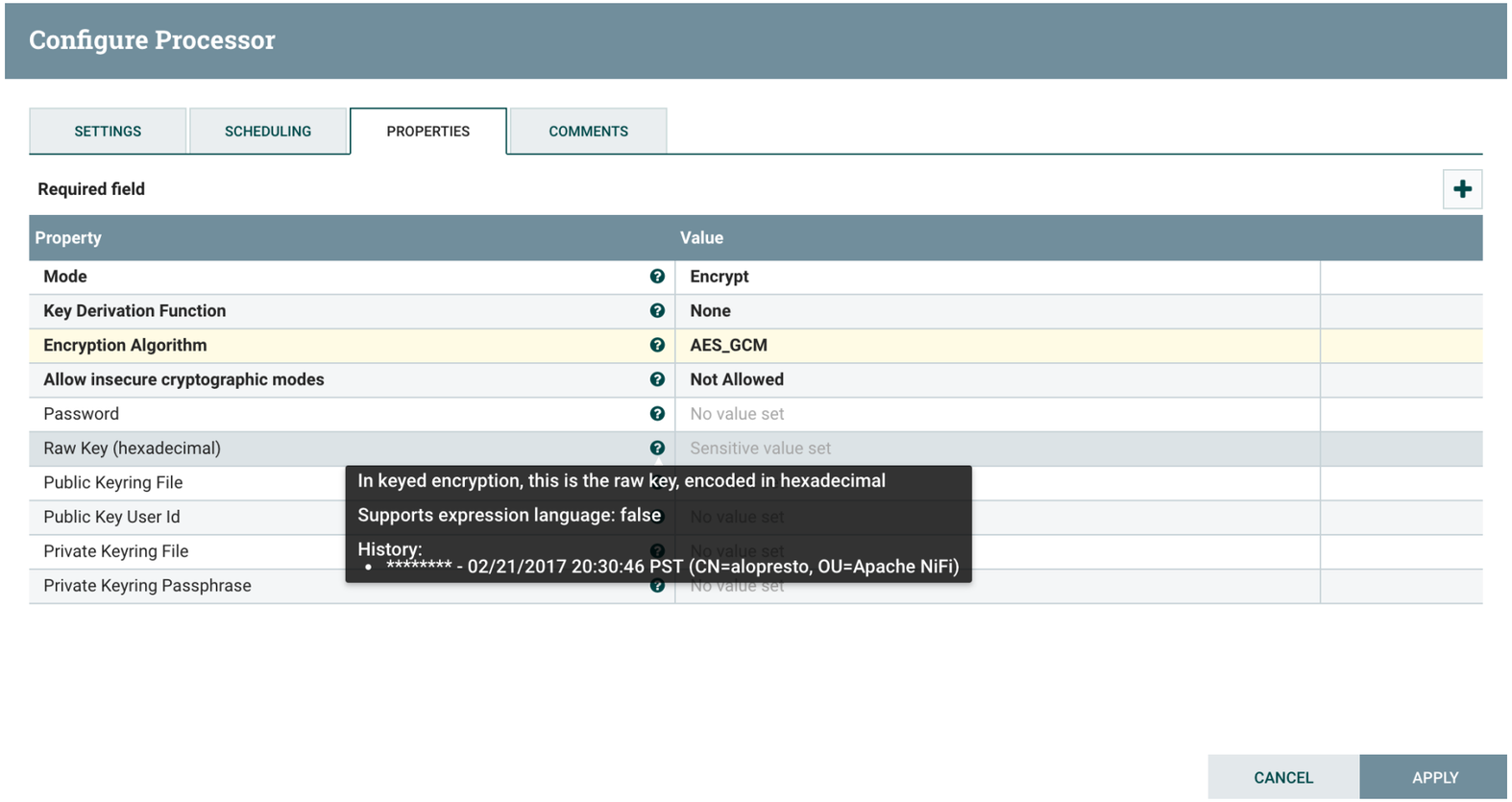
Config Encryption Tool

- First implementation
  - Future options: HSM integration, Vault, KeyWhiz, etc.
- Command-line tool which accepts plaintext `nifi.properties` file and encrypts values using AES/GCM
  - 128 or 256-bit key depending on system JCE unlimited strength cryptographic jurisdiction policies
- Persists master key in `bootstrap.conf`
- Can also handle `login-identity-providers.xml`, `authorizers.xml`, and `flow.xml.gz`

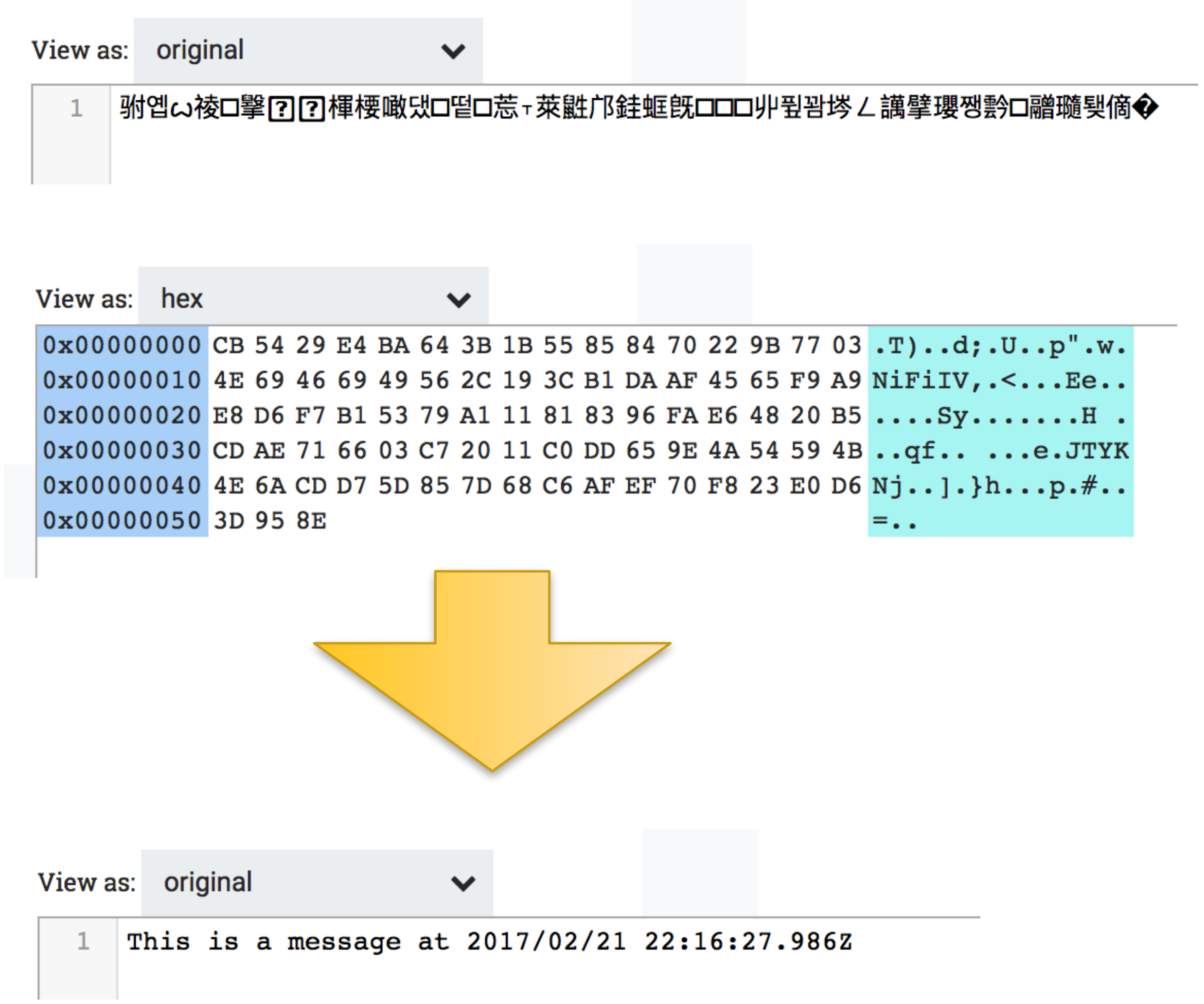
HORTONWORKS

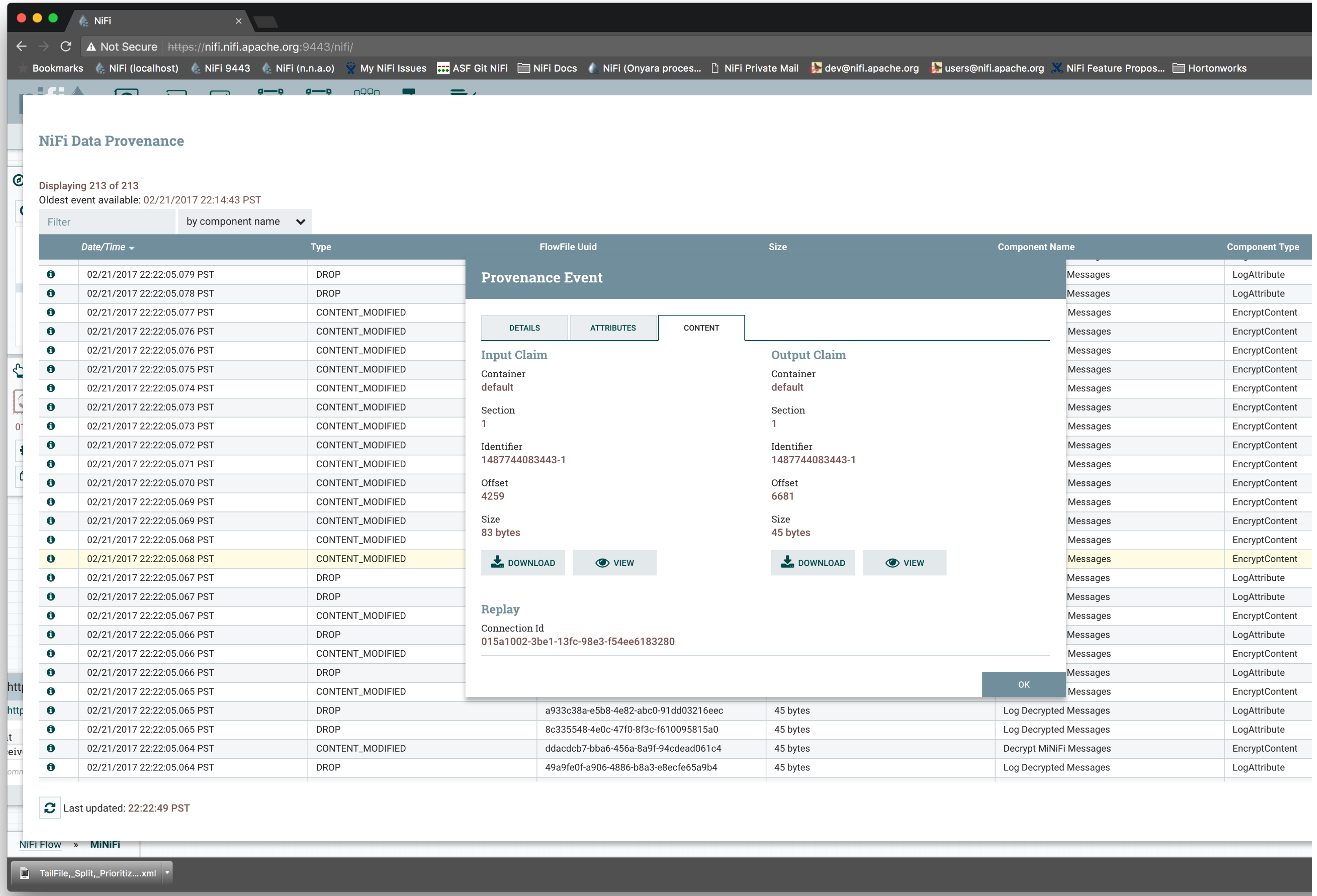
If We Really Have TLS, Why Encrypt Data?

- All data transmitted over TLS is encrypted
- On NiFi, automatically decrypted
- Attributes visible
- Content still encrypted because of `EncryptContent` processor
- Can serve as secure route for follow-on systems



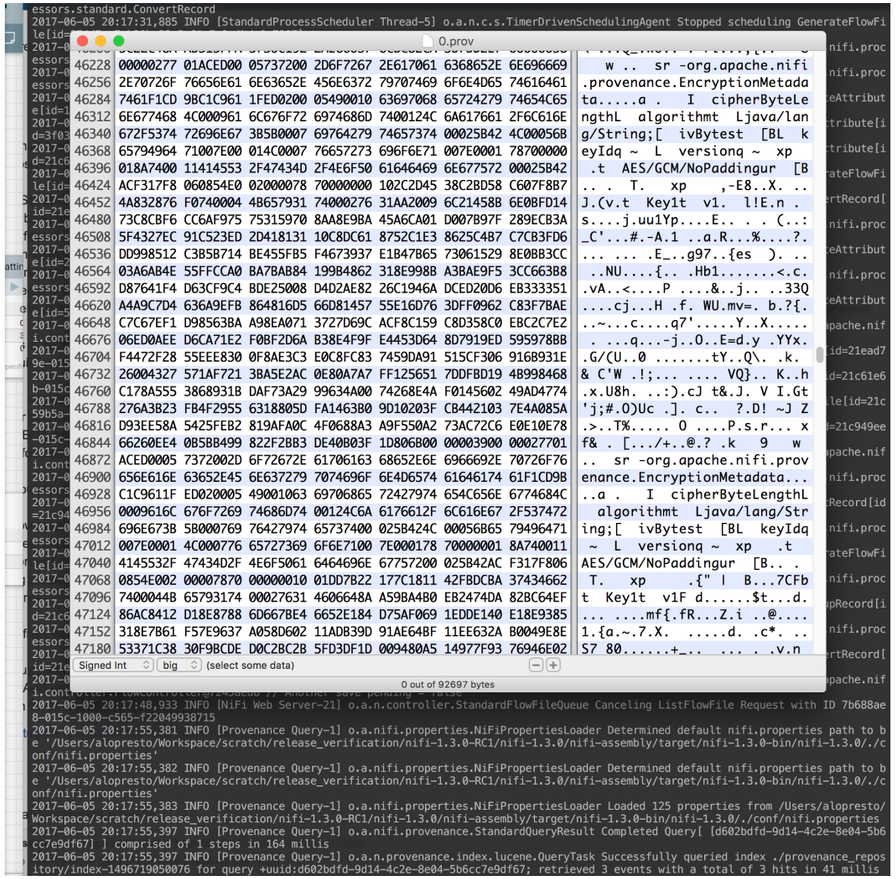
Does It Work?





Encrypted Provenance Repository

- Every provenance event record is encrypted with AES G/CM before being persisted to disk
- Decrypted on deserialization for retrieval/query
- Random access via offset seek
- Handles key migration & rotation



New Features

HORTONWORKS

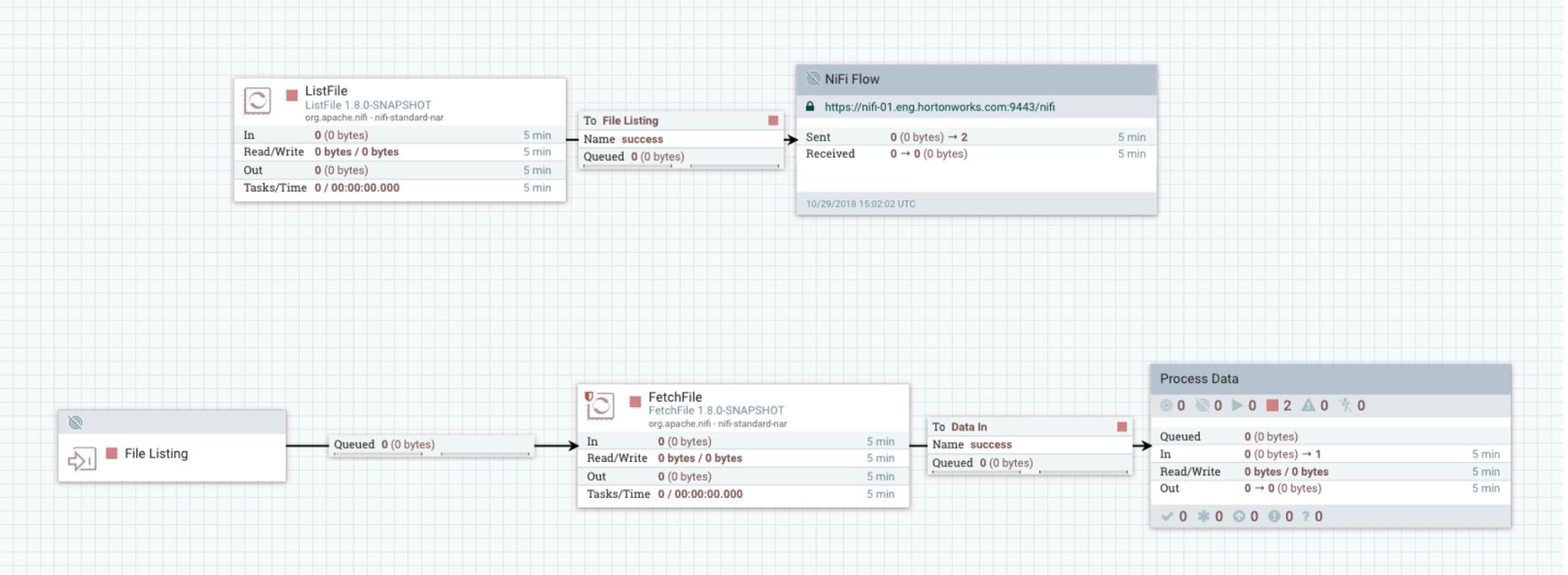
New Features in 1.8.0

- Cluster Data Management
  - Load-Balanced Connections
  - Node Decomissioning
- SQL results to record format
- Elasticsearch lookup service
- Docker improvements
- TLS Toolkit signing w/ external CA (standalone only)

HORTONWORKS

Previously, on *NiFi Clusters*…

- Ingest activity (FTP, file read, etc.) performed on single node
- Routed via RPG to Input Port in cluster
- Flowfiles distributed across all nodes

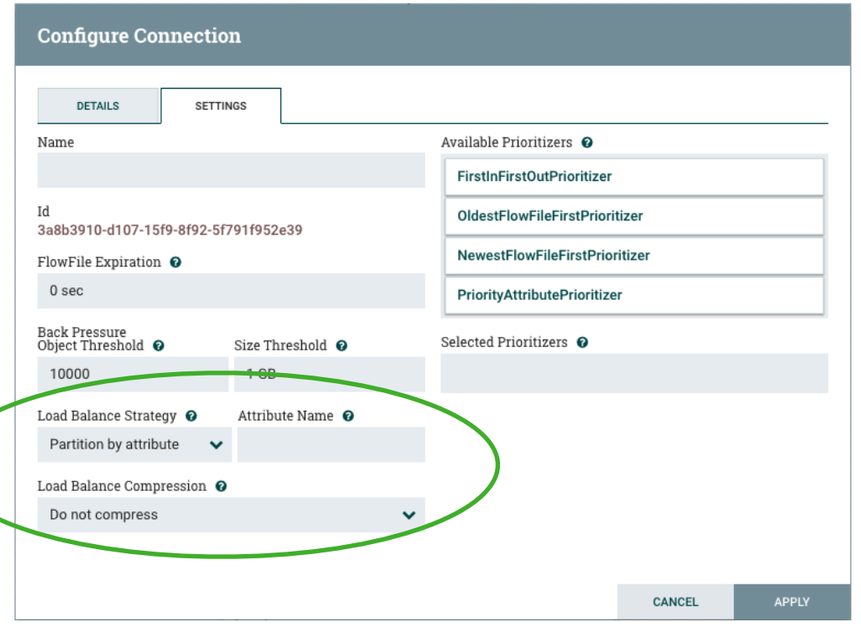


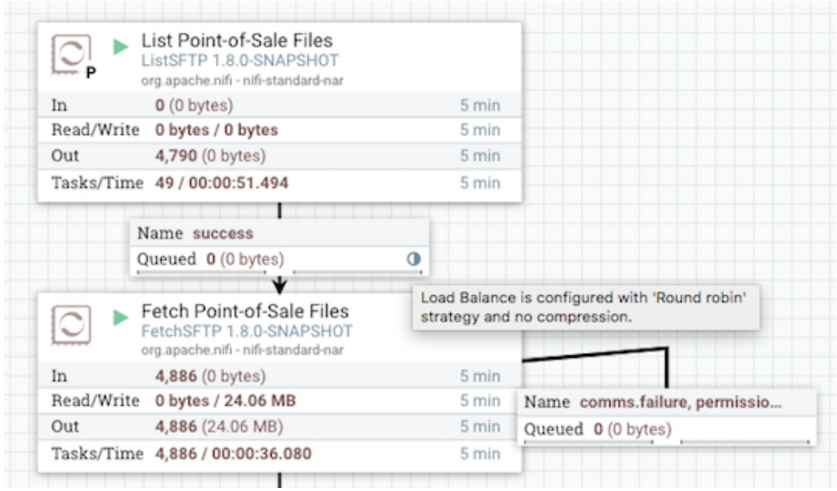
[Mark Payne, Load Balancing Data Across a NiFi Cluster]

HORTONWORKS

Load-Balanced Connections

- Individual connections can be load-balanced
  - None
  - Round Robin
  - Attribute node affinity
  - Single node





[Mark Payne, Load Balancing Data Across a NiFi Cluster]

HORTONWORKS

Roadmap

Roadmap

- Encrypted content and flowfile repositories
- Encryption component enhancements
- Cryptographic proof on sequences
- Sensitive properties & variables
- Sensitive Attributes
- TLS Improvements
- Encrypted Logs

HORTONWORKS

Encrypted Content & Flowfile Repositories

- Similar to encrypted provenance repository
- Data stored on disk encrypted with referenceable keys & metadata
- Still allow random access (offset seek vs. linear read)
- Use framework-level `KeyProvider` interface
- HDF/NiFi ready for deployment on untrusted hardware w/o writing sensitive data* to disk
  - TLS certs and encryption keys still persisted

HORTONWORKS

Encryption Component Improvements

- `EncryptContent`
  - Improvements to key management
  - Enhanced metadata in attributes
- `EncryptAttribute`
- `EncryptRecord`
- Use framework-level `KeyProvider` controller service

HORTONWORKS

Cryptographic Proof on Sequences

- Tired of hearing “blockchain” - signatures vs. distributed ledger
- Providing cascading cryptographic signatures that cover the previous event allows for provable provenance/chain of custody
- Can be applied to:
  - Provenance
  - Flow versioning / `flow.xml.gz` changes
  - Replicated requests in cluster operations
  - User/group authorization/authentication changes

HORTONWORKS

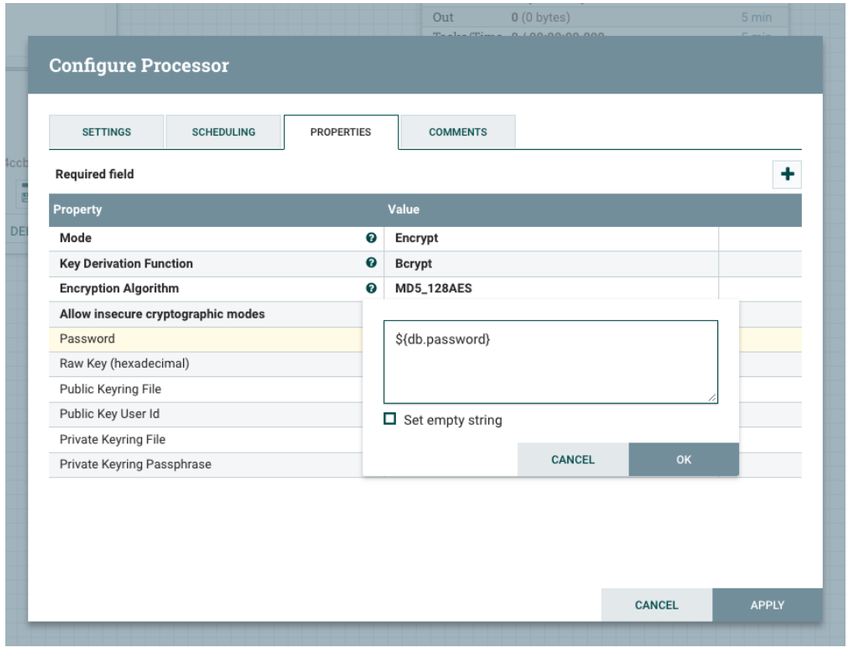
Sensitive Properties & Variables

- Users want to promote flows between environments
- NiFi Registry enables this, but templates and versioned flows do not export sensitive properties
- Not all properties (and no passwords) support Expression Language
  - If `def` = 123, is `abc${def}` a 9 char password, or "`abc`" + value(`def`) = `abc123` ?
- Should enable:
  - Secure storage of sensitive variables on disk (i.e. encryption)
  - Secure export of sensitive variables to shared storage (i.e. encryption via Registry)
  - Key management (i.e. localized, derived, unique keys & encryption translation)
  - Access control for sensitive variables
  - Enable on-demand sensitive protection for dynamic properties

HORTONWORKS

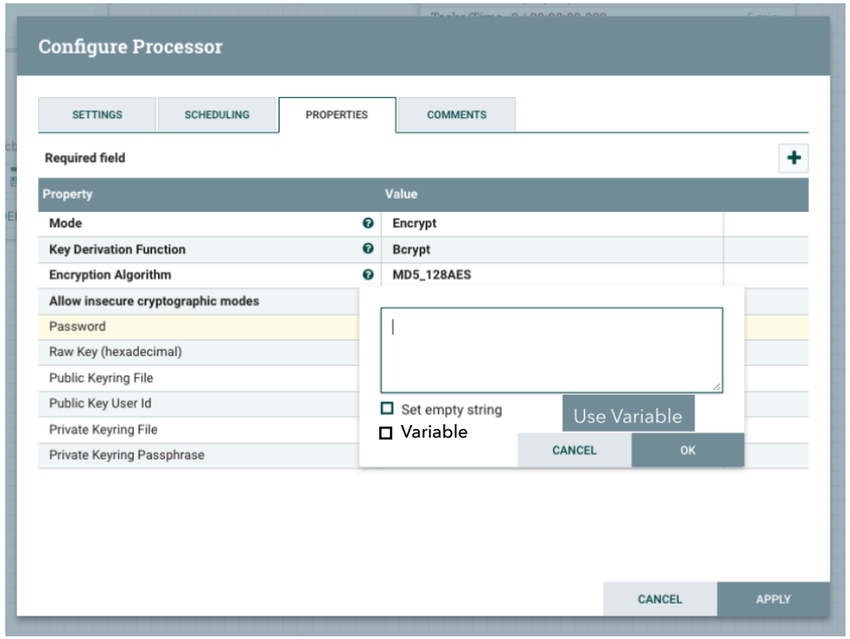
Current Situation

- No way to determine EL evaluation
- No way to protect dynamic properties



Proposed Changes

- Add “Variable” checkbox
  - Checked by default if property descriptor supports EL evaluation
  - If user un-checks, PD does not interpret EL (used for literal values containing ${…}
- Add “Use Variable” button (auto-complete)



HORTONWORKS

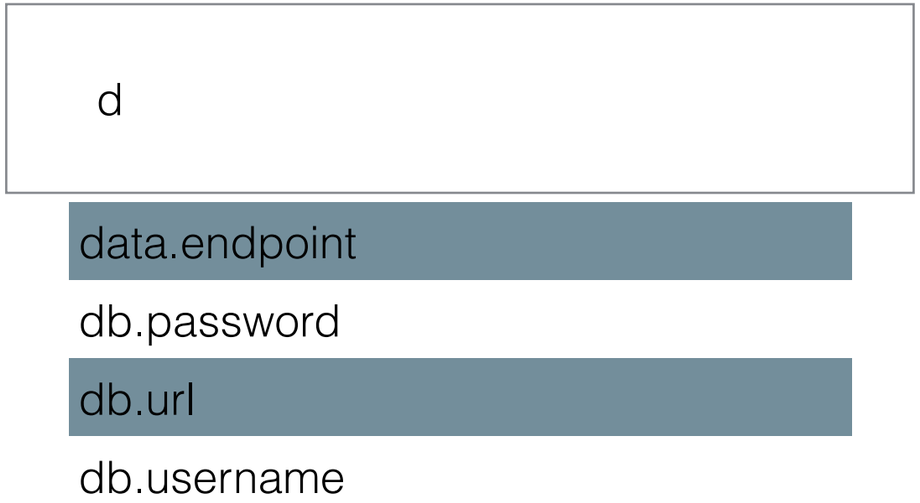
Use Variable

- Opens text field
- Allows free typing

HORTONWORKS

Use Variable

- Typing triggers AJAX request to return variables
- New API endpoint
  - `GET /process-groups/abcd…/variables?q=d`
  - Text query filter
  - Permissions applied on server
  - Scope applied on server



HORTONWORKS

Variable Export

- Options
  - Never allow
  - Allow based on Registry admin setting (per-bucket?)
- Key management policies
  - Export in plaintext (**not secure**)
  - Export encrypted with original key (can't be deployed to other NiFi instances/rolled-over instances)
  - Export encrypted with per-Registry key (maintains independence of NiFi instances; preferred)
    - Import operation would then re-encrypt with destination NiFi's `nifi.sensitive.props.key`
    - Also possible to export with derived key to isolate buckets

HORTONWORKS

Sensitive Attributes

- Some attributes contain sensitive data
- While components currently have access control policies, flowfiles/attributes do not
- Restricting access to sensitive attributes (PII, confidential, etc.) allows DFM/domain specialists to build and monitor flows without viewing data
- Prevents examination via provenance viewing as well

HORTONWORKS

TLS Improvements

- Handle keystores with multiple certificates to allow for varied identification by role
- Automatic Protocol/Cipher Suite Upgrades
  - Custom list of cipher suites and protocol versions loaded into Jetty at startup (work done; PR to be opened)
  - Integration with Mozilla TLS Labs tools
  - Plug n' Play Simplicity - "Set it and forget it"
    - Automatically sets and continuously analyzes supported cipher suites to ensure compatibility & risk minimization
- Extend TLS Toolkit
  - Better handle pre-existing certificates/certificate authorities
  - Help enterprise users securely deploy NiFi with less friction
  - Provide "sanity check" feature to test connectivity with provided keystore/truststore combos & identify potential issues
  - Refactor underway

Encrypted Logs

- NiFi Provides Alternatives to Log Grepping
  - Provenance Data
  - Processor Stats
- Pluggable Log Interceptors
  - Accept messages, key/value pairs
  - Sensitive keys are registered in a lookup
  - Values for these keys are encrypted (using log-specific key) before output
- Log File Encryption
  - More difficult to debug, but more complete protection
    - Restrict user/client access logs, logs of encryption/lookup operations, etc.
  - Batching/latency trade-off
- Alternative method is hashing
  - Searching logs only requires executing hash function on query to generate “real” search term

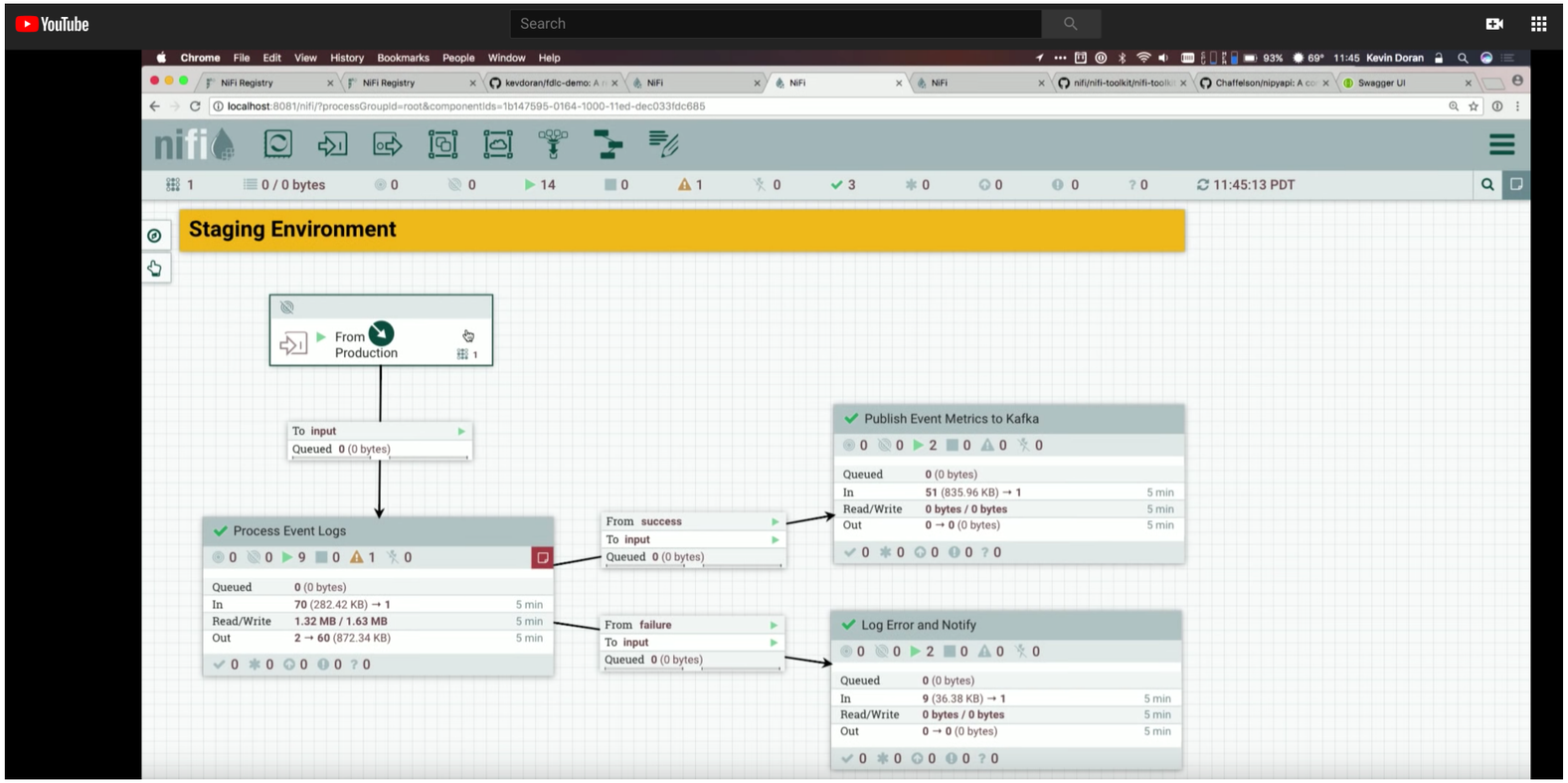
HORTONWORKS

Community

HORTONWORKS

More Resources

- *FDLC with Apache NiFi*, Kevin Doran
- *NiPyAPI Docs*, Daniel Chaffelson
- *DevOps Tips*, Tim Spann
- *Automate Workflow*, Pierre Villard



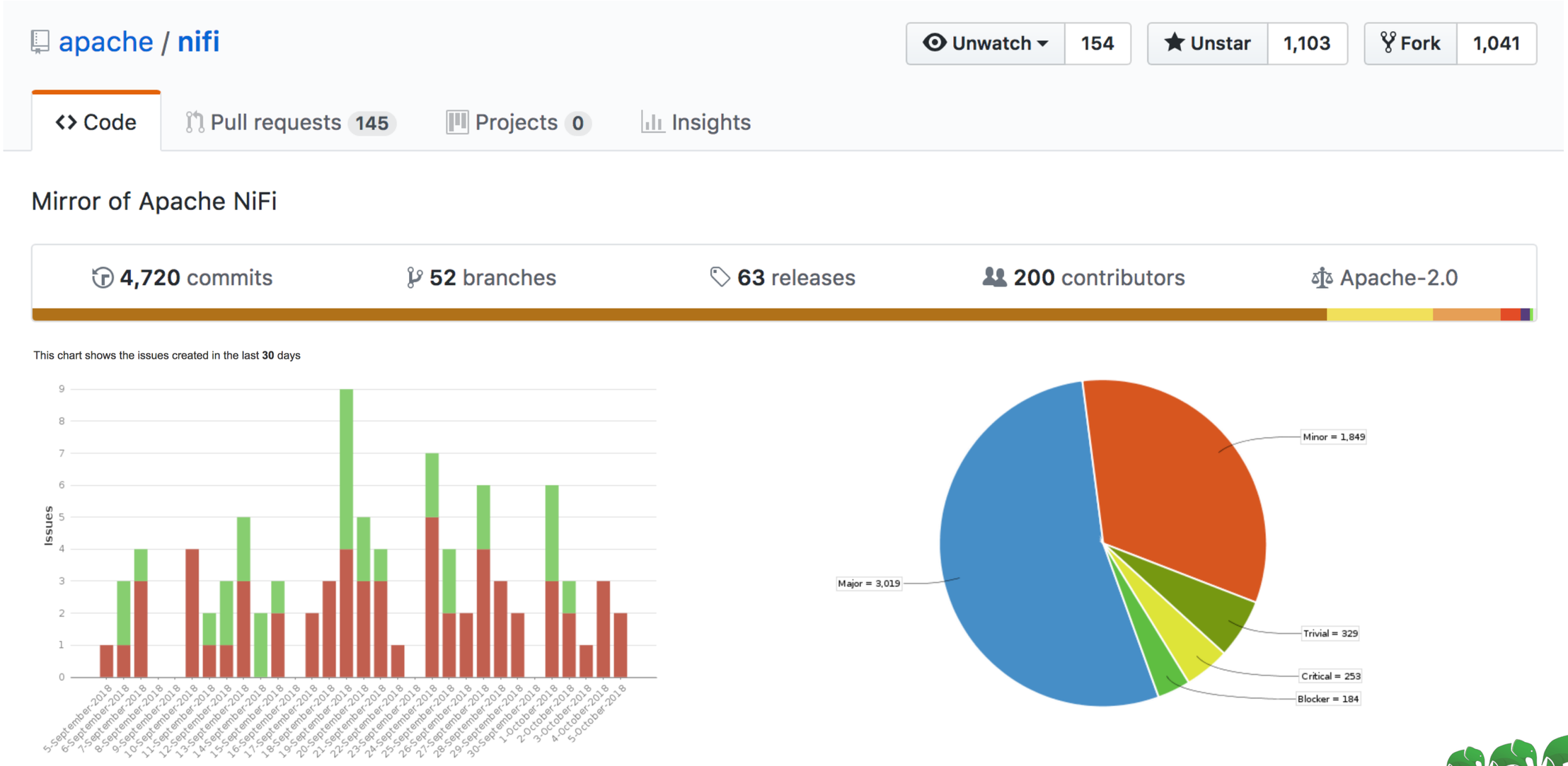
HORTONWORKS

New Announcements

- NiFi 1.8.0 — 26 Oct 2018 (212+ Jiras)
  - Jetty, DB improvements
  - Auto load-balancing queues
  - TLS Toolkit w/ external CA
  - Record processor improvements
- MiNiFi C++ 0.5.0 — 6 June 2018
- MiNiFi Java 0.5.0 — 7 July 2018
- NiFi Registry 0.3.0 — 25 Sept 2018

HORTONWORKS

Community Health



Learn more and join us

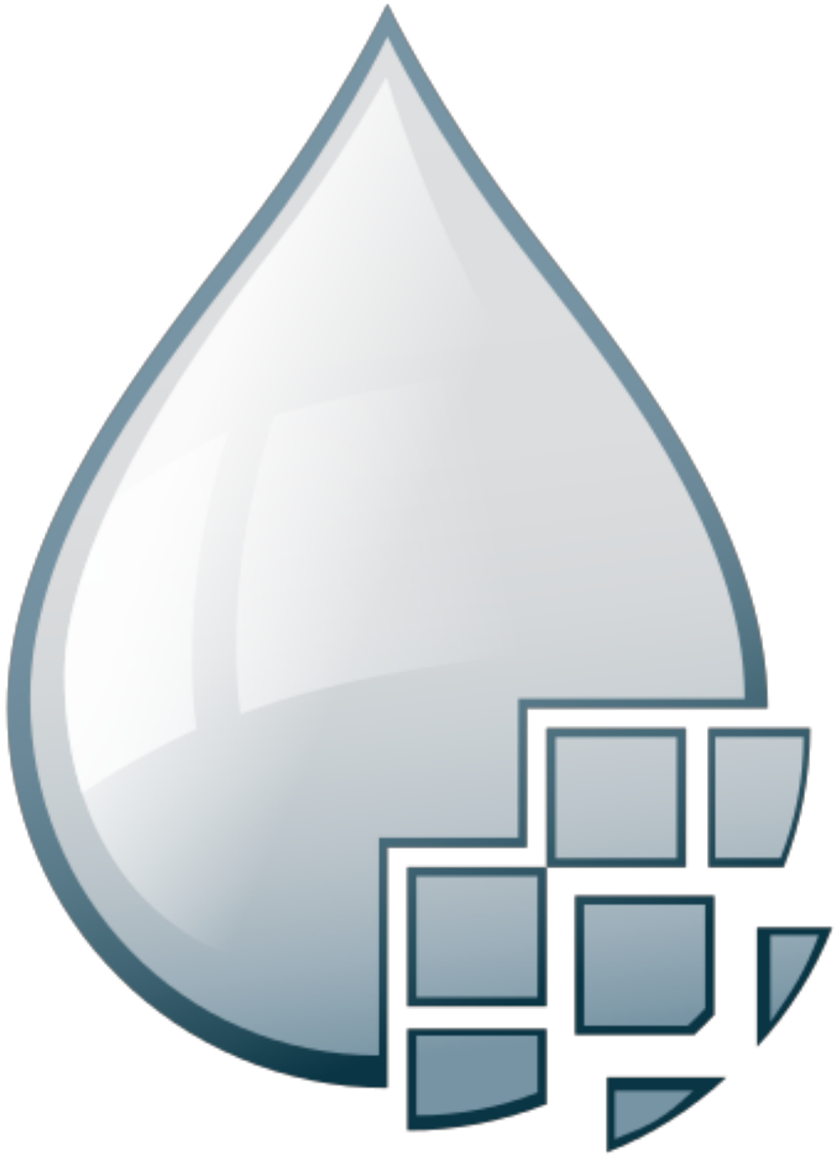
Apache NiFi site
https://nifi.apache.org

Subproject MiNiFi site
https://nifi.apache.org/minifi/

Subscribe to and collaborate at
dev@nifi.apache.org
users@nifi.apache.org

Submit Ideas or Issues
https://issues.apache.org/jira/browse/NIFI

Follow us on Twitter
@apachenifi



HORTONWORKS

Thank you

alopresto@hortonworks.com | alopresto@apache.org | @yolopey

github.com/alopresto/slides