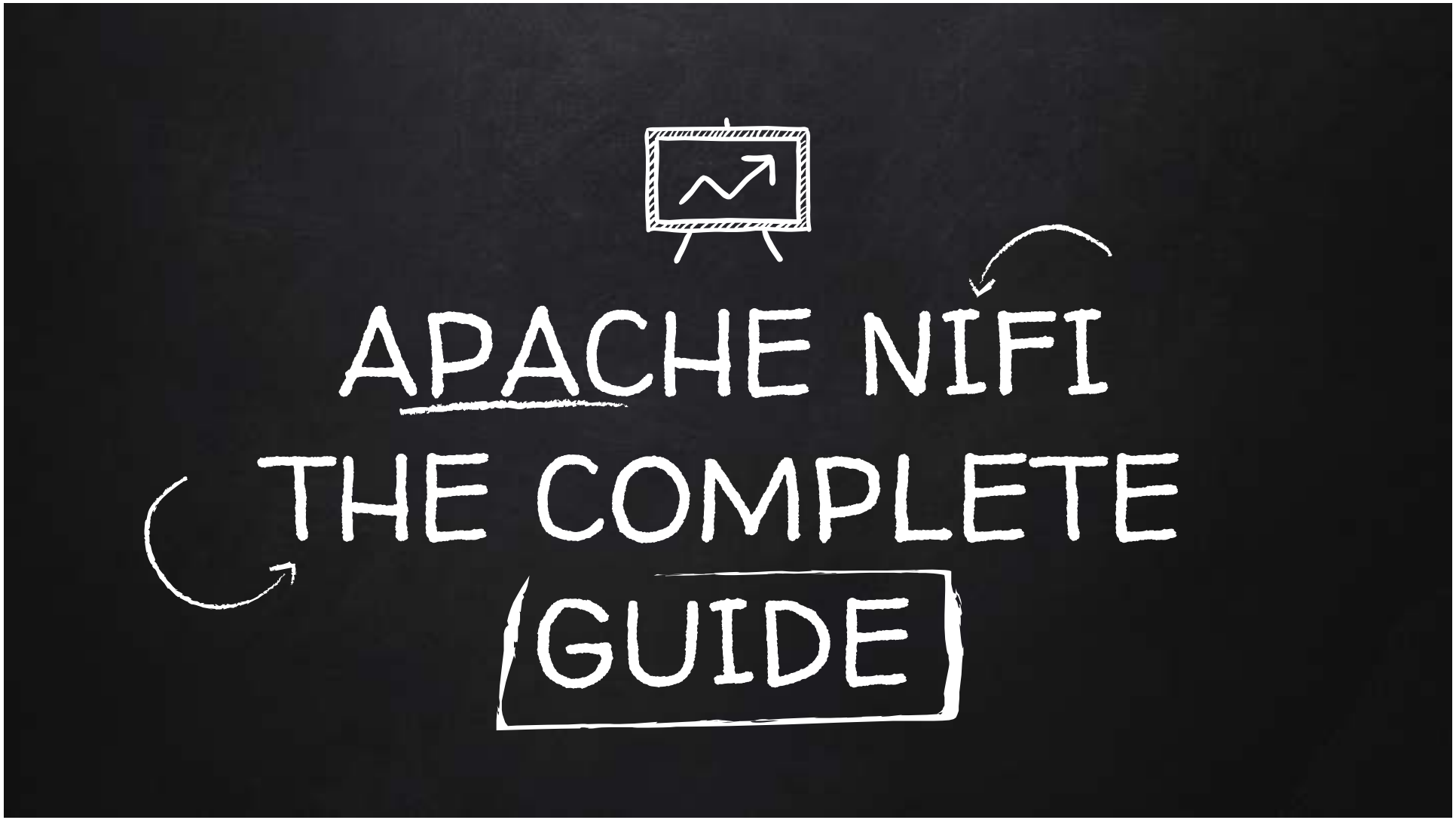

APACHE NIFI
THE COMPLETE
GUIDE

What is Data Flow, Data Pipeline & ETL?

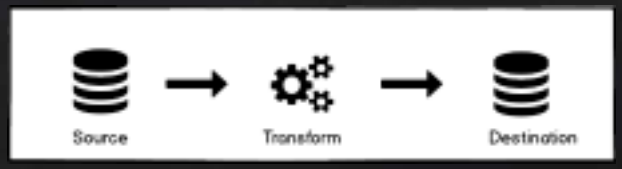
Let's start with the basic terms in the data world.

What is Data Flow?

- ✗ Data Flow -> Moving data/content from Source to Destination
- ✗ Data can be CSV, JSON, XML, Logs, HTTP data, Image, Videos, Telemetry data, etc..

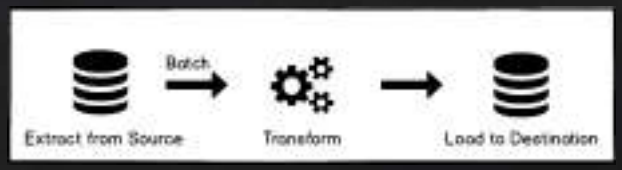
What is Data Pipeline?

- Data Pipeline -> Movement and Transformation of data/content from Source to Destination



What is ETL?

- ETL -> E – Extract, T – Transform, L – Load

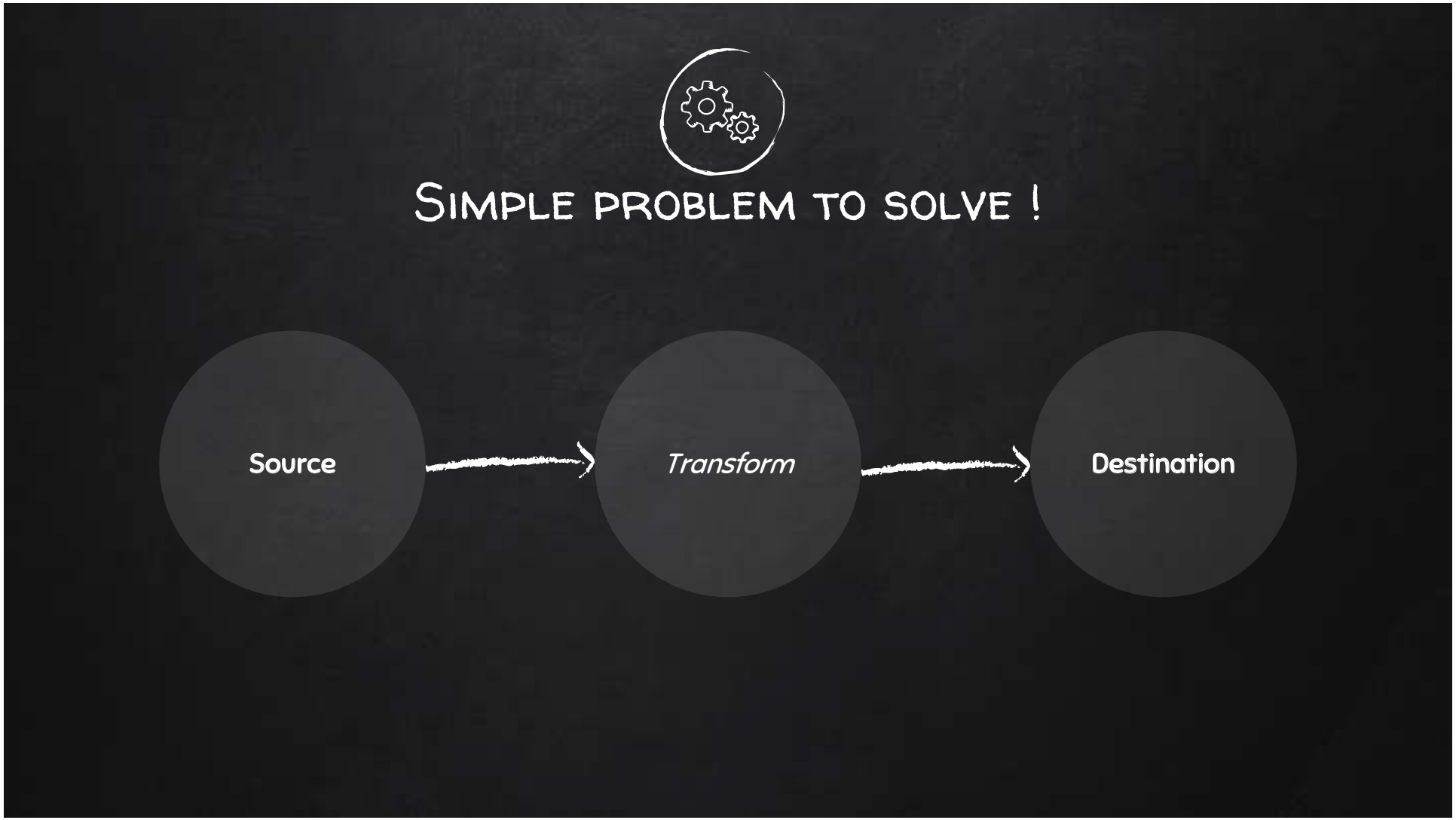


ETL vs Data Pipeline

- ✗ ETL -> Batch
- ✗ Data Pipeline -> Batch / Stream

Why we should use a Framework for Data Flow

Considerations while building your own Data Flow / Data Pipeline.

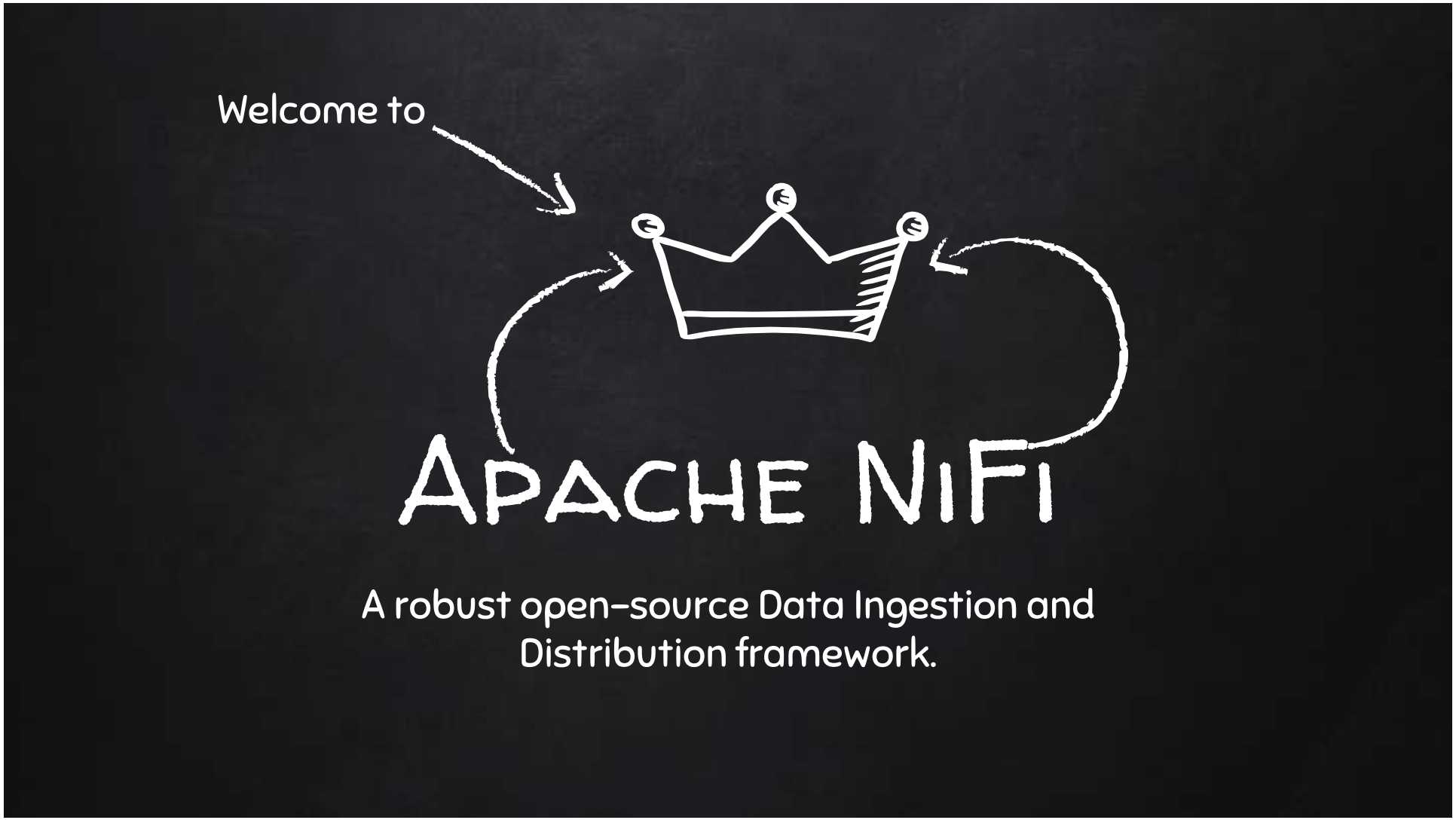

Simple problem to solve !
Source
Transform
Destination

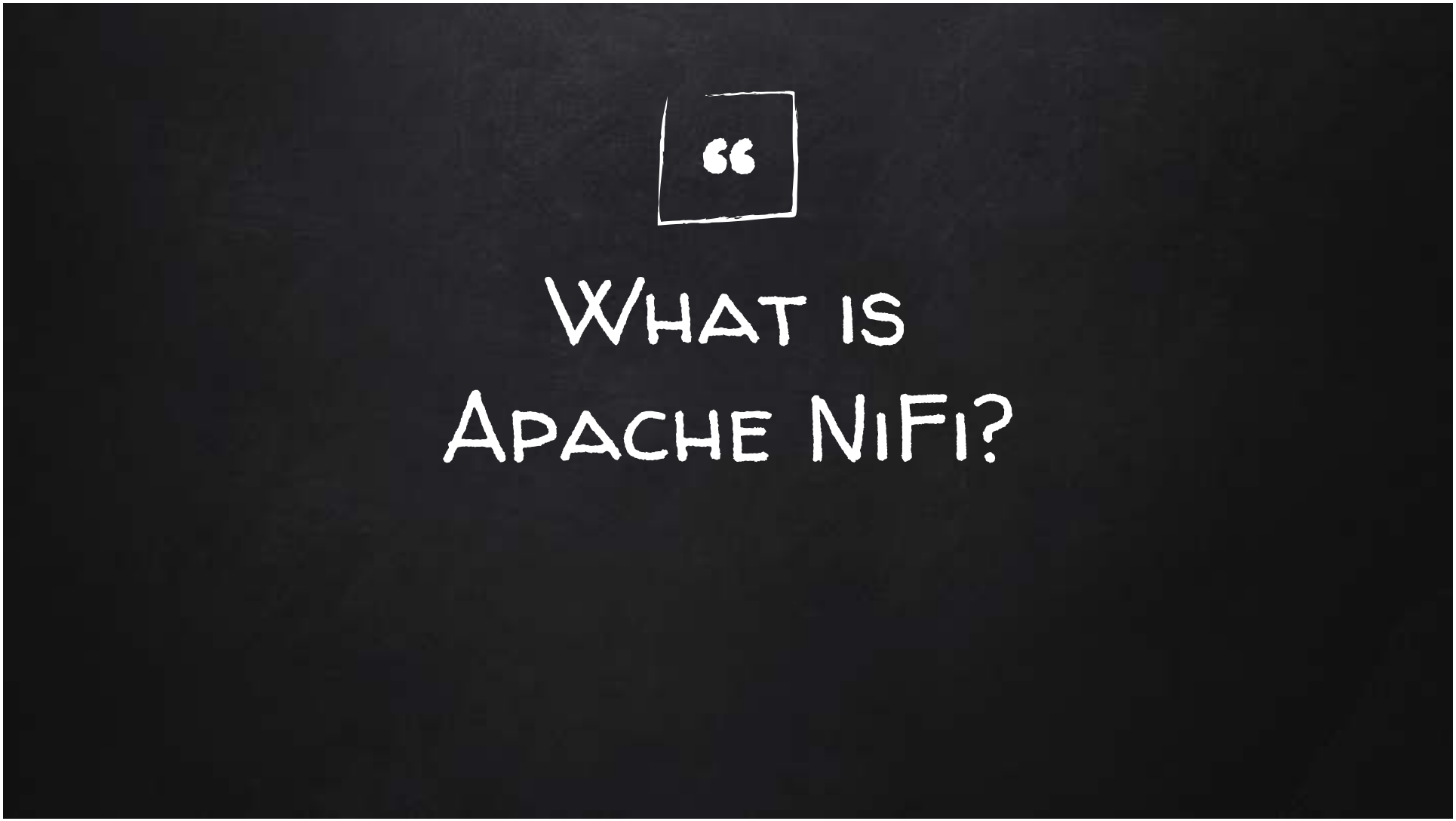
Four V's

- ✗ Volume –> refers to the vast amounts of data generated every second.
- ✗ Velocity –> refers to the speed at which new data is generated and the speed at which data moves around.
- ✗ Variety –> refers to the different types of data we can now use.
- ✗ Veracity – refers to the messiness or trustworthiness of the data.

Considerations

- ✗ Support for multiple data format – CSV, JSON, Plaintext, Images, Videos, etc.
- ✗ Support for various types of sources and destinations – FTP, HTTP, SQL Databases, NoSQL Databases, Search Engines, Cache Server, etc.
- ✗ Scalable and Reliable for large volume and high-velocity data.
- ✗ You should also consider Data Cleansing and Data Validation logics.


Welcome to
Apache NiFi
A robust open-source Data Ingestion and
Distribution framework.


“
What is
Apache NiFi?

Apache NiFi supports powerful and scalable directed graphs of data routing, transformation, and system mediation logic.

NiFi was built to automate the flow of data between systems. It can propagate any data content from any source to any destination.

Installing Apache NiFi in a Mac / Linux

Installing Apache NiFi in a Windows Machine

NiFi User Interface


Core NiFi
Terminologies

Flow-Based Programming (FBP)

- NiFi is based on Flow Based Programming (FBP).
- Flow Based Programming (FBP) is a programming paradigm that defines applications as networks of "black box" processes, which exchange data across predefined connections by message passing, where the connections are specified externally to the processes. These black box processes can be reconnected endlessly to form different applications without having to be changed internally. FBP is thus naturally component-oriented.

How NiFi works?

- ✗ NiFi consists of atomic elements which can be combined to groups to build a data flow.
- ✗ NiFi consists of Processors and Process Groups.

What is a Processor?

- ✗ Processors are an atomic elements in NiFi which can do some simple tasks.
- ✗ At the time of recording this video, NiFi has 280+ Processors.
- ✗ Each Processors in NiFi is unique in it's own way.
- ✗ NIFI have processors for almost anything.

What a Processor can do?

- ✗ We have tons of Data Source and Data Sink Processors.
- ✗ The Source and Sink can be anything – SQL, NoSQL, Search Engine, Cache Server, Messaging Queue, AWS Entities, etc.
- ✗ NiFi have processors for all your data needs.
- ✗ NiFi supports custom processors as well.

What is a FlowFile?

- FlowFile = Actual Data -> CSV, JSON, XML, Plaintext, SQL, Binary, etc.
- FlowFile – Abstraction of data in NiFi.
- A Processor can generate new Flow File by processing an existing Flow File or ingesting new data from any source.

What is a Connection?

- In NiFi all processors can be connected with each other to create a data processing flow.
- Processors are linked via connections.
- Each connection will act as a Queue for FlowFiles.

What is a Process Group?

- ✗ Set of Processors can be combined in NiFi to form a Process Group.
- ✗ Process Groups helps to maintain large and complex data flow.
- ✗ Input and Output ports are used to move data between Process Groups.

What is a Controller Service?

- ✗ A Controller Service is a shared service that can be used by a Processor.
- ✗ Controller Service can hold DB Connection details.
- ✗ We can create Controller Service for CSV Reader, JSON Writer and many more.

More on FlowFiles

Components of FlowFile

- A FlowFile contains data.

- A FlowFile is composed two components
  - Content
  - Attributes

Content & Attributes of FlowFile

- ✗ Content:
  - That's the actual content of the FlowFile.
    - It's the actual content of a file you would read using GetFile, GetHTTP, etc.
- ✗ Attributes:
  - These are the metadata from the FlowFile.
  - Contains information about the content:
    - When was it created, it's name, where is it from, what does it represent.

Construct using NiFi FlowFiles

- A processor can (either or both)
  - Add, update, or remove attributes of a FlowFile.
  - Change content of a FlowFile.

- Update the Attributes or Content or both using various processors available in NiFi to design your Dataflow.

Life Cycle of FlowFile

- ✗ FlowFiles are persisted in the disk.
- ✗ FlowFiles are passed-by-reference.
- ✗ A new FlowFile will be created if the content of the existing FlowFile is modified or new data is ingested from source.
- ✗ New FlowFile will not be created if the attributes of the existing FlowFile is modified.

What Processors are Available in NiFi?

The Swiss Army Knife of Data Flow.

Why it's Important?

- In order to create an effective dataflow, the users must understand the various types of Processors available for them to use.
- NiFi out of the box have different types of processors for all your data needs.
- At the time of recording this video, NiFi has 280+ Processors.

Data Ingestion Processors

- ✗ GenerateFlowFiles
- ✗ GetFile
- ✗ GetFTP
- ✗ GetSFTP
- ✗ GetJMSQueue
- ✗ GetJMSTopic
- ✗ GetHTTP
- ✗ ListenHTTP
- ✗ ListenUDP
- ✗ GetHDFS
- ✗ GetKafka
- ✗ QueryDatabaseTable
- ✗ GetMongo
- ✗ GetTwitter
- ✗ ListHDFS / FetchHDFS

Data Transformation Processors

- ✗ ConvertRecord
- ✗ UpdateRecord
- ✗ ConvertJSONToSQL
- ✗ ReplaceText
- ✗ CompressContent
- ✗ ConvertCharacterSet
- ✗ EncryptContent
- ✗ TransformXml
- ✗ JoltTransformJSON

Data Egress / Sending Data Processors

- ✗ PutEmail
- ✗ PutFile
- ✗ PutFTP
- ✗ PutSFTP
- ✗ PutJMS
- ✗ PutSQL
- ✗ PutKafka
- ✗ PutMongo
- ✗ PutHDFS

Routing and Mediation Processors

- ✗ ControlRate
- ✗ DetectDuplicate
- ✗ DistributeLoad
- ✗ RouteOnAttribute
- ✗ RouteOnContent
- ✗ ScanAttribute
- ✗ ScanContent
- ✗ ValidateXml
- ✗ ValidateCSV

Database Access Processors

- ✗ ConvertJSONToSQL
- ✗ ExecuteSQL
- ✗ PutSQL
- ✗ SelectHiveQL
- ✗ PutHiveQL
- ✗ ListDatabaseTables

Attribute Extraction Processors

- ✗ EvaluateJsonPath
- ✗ EvaluateXPath
- ✗ EvaluateXQuery
- ✗ ExtractText
- ✗ HashAttribute
- ✗ HashContent
- ✗ IdentifyMimeType
- ✗ UpdateAttribute
- ✗ LogAttribute

System Interaction Processors

- ✗ ExecuteProcess
- ✗ ExecuteStreamCommand

Splitting and Aggregation Processors

- ✗ SplitText
- ✗ SplitJson
- ✗ SplitXml
- ✗ SplitRecord
- ✗ SplitContent
- ✗ UnpackContent
- ✗ SegmentContent
- ✗ MergeContent
- ✗ QueryRecord

HTTP and UDP Processors

- ✗ GetHTTP
- ✗ ListenHTTP
- ✗ InvokeHTTP
- ✗ PostHTTP
- ✗ HandleHttpRequest
- ✗ HandleHttpResponse
- ✗ Listen UDP
- ✗ PutUDP
- ✗ ListenUDPRecord

Amazon Web Services Processors

- ✗ FetchS3Object
- ✗ PutS3Object
- ✗ PutSNS
- ✗ GetSQS
- ✗ PutSQS
- ✗ DeleteSQS
- ✗ GetDynamoDB
- ✗ PutDynamoDB
- ✗ PutLambda

Processor Configurations, Connections & Relationships In NiFi

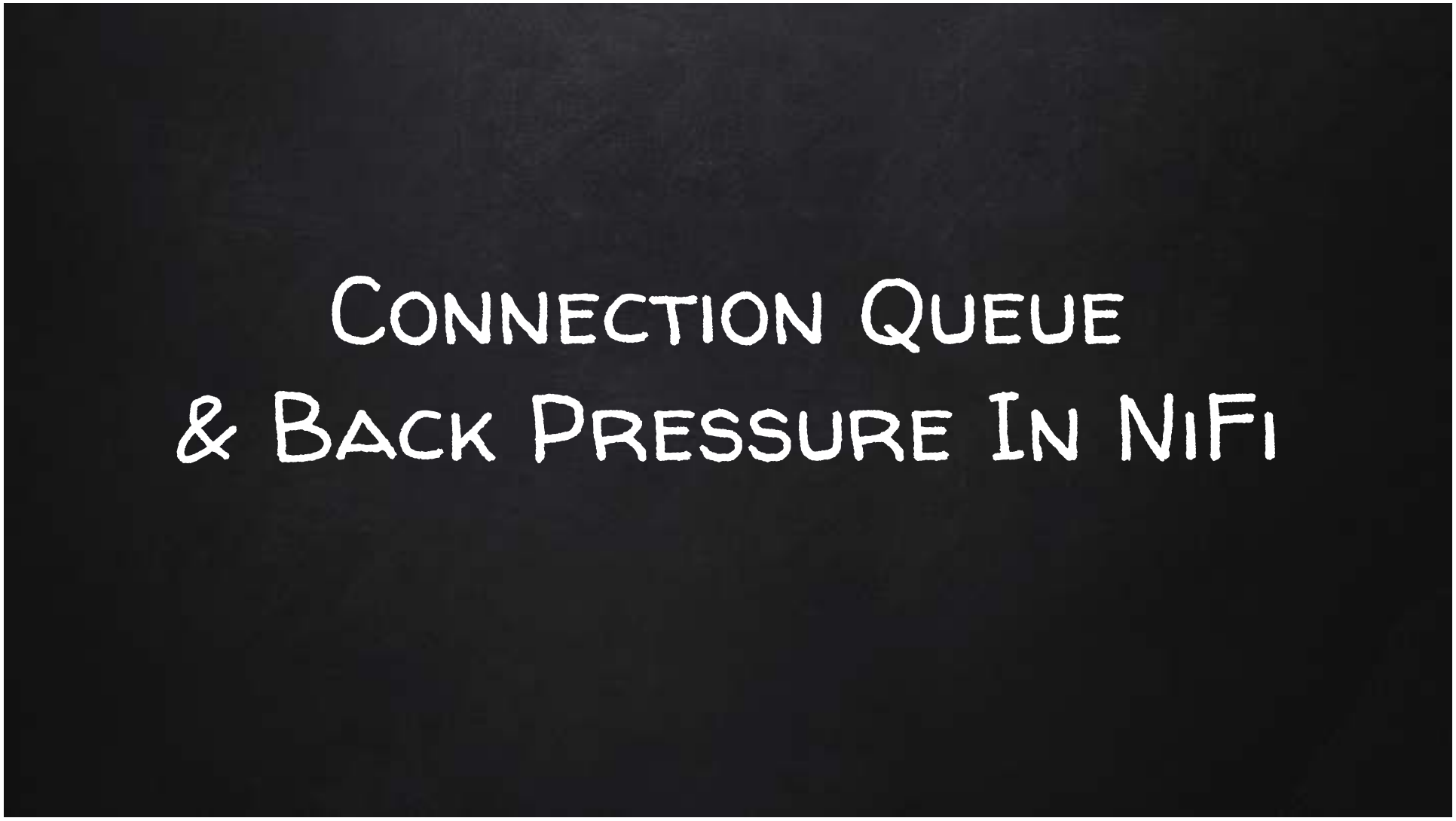
Configuration over coding !!

Type of NiFi Processor Configurations

- ✗ Standard Configurations – Configuration common across all processors
- ✗ Unique Configurations – Configuration specific to a processor

What is a Relationship?

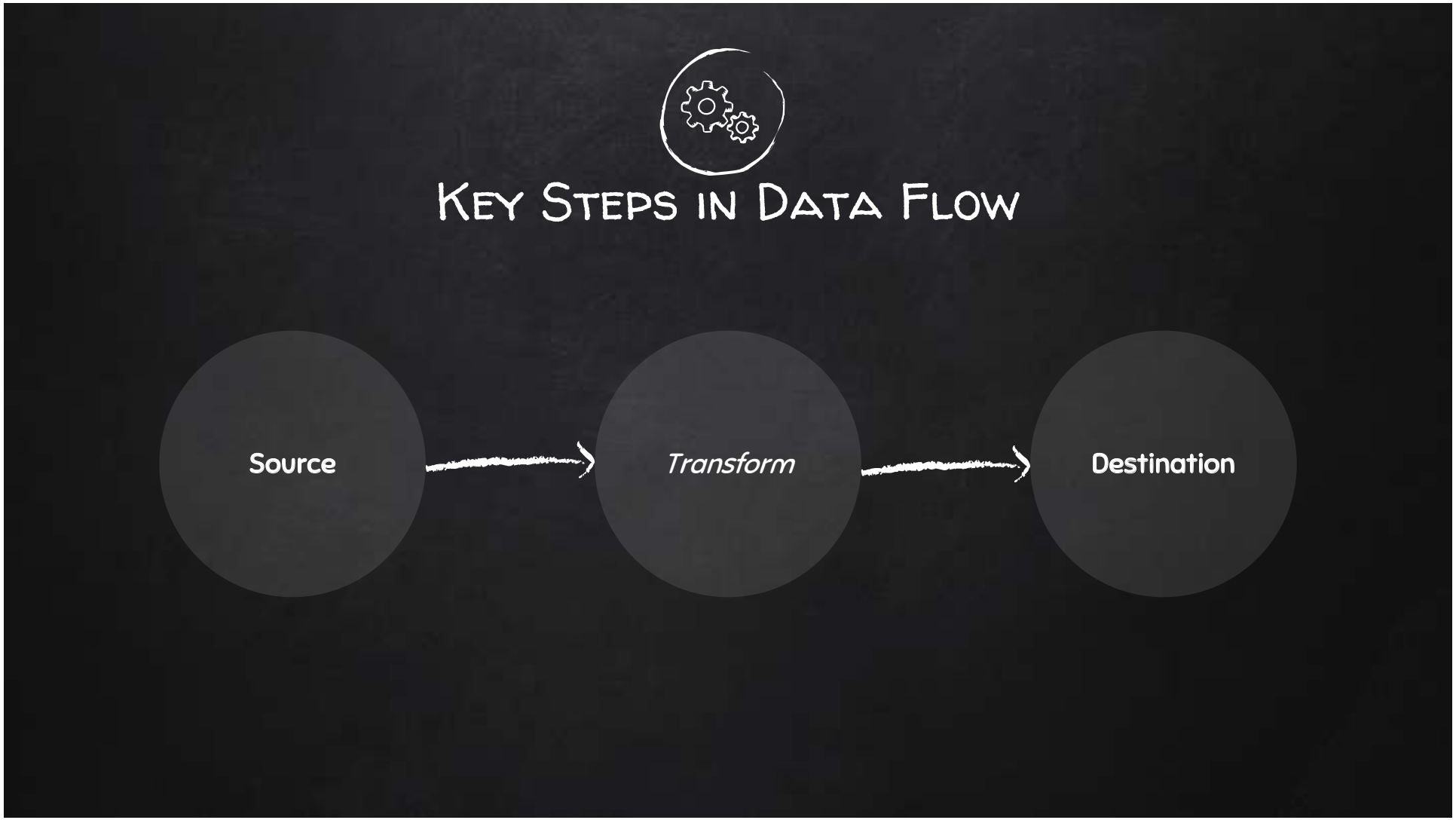
- Each Processor has zero or more Relationships defined for it.
- Relationships are named to indicate the result of processing a FlowFile.
- After a Processor has finished processing a FlowFile, it will transfer the FlowFile to one of the Relationships.
- The FlowFile creator needs to handle all the relationship of a processor or terminate unhandled relationship.


Connection Queue
& Back Pressure In NiFi

What is Back Pressure?

- Each Processor has zero or more Relationships defined for it.
- Relationships are named to indicate the result of processing a FlowFile.
- After a Processor has finished processing a FlowFile, it will transfer the FlowFile to one of the Relationships.
- The FlowFile creator needs to handle all the relationship of a processor or terminate unhandled relationship.

Working with Attributes & Content in NiFi


Key Steps in Data Flow
Source
Transform
Destination

Working with Expression Language in NiFi

Key Takeaway

- ✗ NiFi is Swiss Army Knife of Data Flow with Various Processors and Components.
- ✗ There is no single right approach.
- ✗ But there can be one efficient approach compared to the others.
- ✗ Always choose the efficient approach.

Key Factors Which Determines the Efficiency of a Data Flow

- ✗ Number of IO Operations
- ✗ Memory Utilization
  - CPU Utilization
  - For RAM Utilization

More on Expression Language Functions in NiFi

Working with Process Group, Input Port & Output Port in NiFi

Problem Statement

- ✗ How will you share a data flow with your friend or colleague to help in debugging?
- ✗ How will you move your data flow from one machine to another machine and continue your flow design from the new machine?
- ✗ How will you move your completed data flow from a development environment to a testing environment?

Working with Templates in NiFi

Import and Export data flow using Templates

Version Control in NiFi

- ✗ Though templates are good to share data flow it's not ideal for Version Control.
- ✗ **"NiFi Registry"** is the right tool for Version Control of Data Flow in NiFi.

Working with Funnel in NiFi

Combine data from several connections to single connection ! !


Working with
Controller Services
in NiFi

What is Controller Service?

- ✗ A Controller Service is a shared service which can be used across Processors or other Controller Services.
- ✗ "Controller Service" is not limited to Database configuration.
- ✗ You can do various abstraction of functionality using it.

Working with Variable Registry in NiFi

Why to use Variable Registry?

- ✗ To hold environmental or system specific properties.
- ✗ Helps to simplify the configuration management and migration of Data Flow across environments.
- ✗ Helps in CI/CD of Data Flow.

How to Define Variable Registry?

- ✗ Variable Registry can be defined using two ways
  - Using **"Variables"** window available in NiFi UI.
  - Using **"Custom Configuration File"** and referring them in **"nifi.properties"** file.

Advantages of UI Based Variables

- ✗ Used to add variables without restarting the NiFi server.
- ✗ Helps to override an incorrect variable defined using property file based approach.

Template & Variable Registry

- ✗ Templates created will have all the variables along with its value.
- ✗ The variables pointing to sensitive field like password will be ignored during template creation.

FlowFile Prioritization in NiFi

What is FlowFile Prioritization?

- ✗ Prioritizers comes handy when you have data coming from multiple sources, and you would like to process some data immediately after it arrives compared to other data.

Types of FlowFile Prioritization

- **FirstInFirstOutPrioritizer:**
  - Given two FlowFiles, the one that reached the connection first will be processed first.
- **NewestFlowFileFirstPrioritizer:**
  - Given two FlowFiles, the one that is newest in the dataflow will be processed first.
- **OldestFlowFileFirstPrioritizer:**
  - Given two FlowFiles, the one that is oldest in the dataflow will be processed first. 'This is the default scheme that is used if no prioritizers are selected.'

Types of FlowFile Prioritization (Cont.)

- **PriorityAttributePrioritizer:**
  - Given two FlowFiles that both have a "priority" attribute, the one that has the highest priority value will be processed first. Note that an UpdateAttribute processor should be used to add the "priority" attribute to the FlowFiles before they reach a connection that has this prioritizer set. Values for the "priority" attribute may be alphanumeric, where "a" is a higher priority than "z", and "1" is a higher priority than "9", for example.

FlowFile Expiration in NiFi

What is FlowFile Expiration?

- ✗ FlowFile expiration is a concept by which data that cannot be processed within a particular timeframe can be automatically removed from the flow.

- ✗ This will come in handy when the volume of data is expected to exceed the amount that can be processed by NiFi.

How FlowFile Expiration works?

- ✗ The expiration period is based on the time that the data entered the NiFi instance.
- ✗ We can use the expiration in conjunction with Prioritizers to ensure that the highest priority data is processed first and then anything that cannot be processed within a specified period can be dropped.

Monitoring NiFi

With great power comes great responsibility ! !


Monitoring NiFi
Using
Reporting Task

What is a Reporting Task?

- A reporting task in NiFi runs in the background and provides various statistics of the NiFi instance.

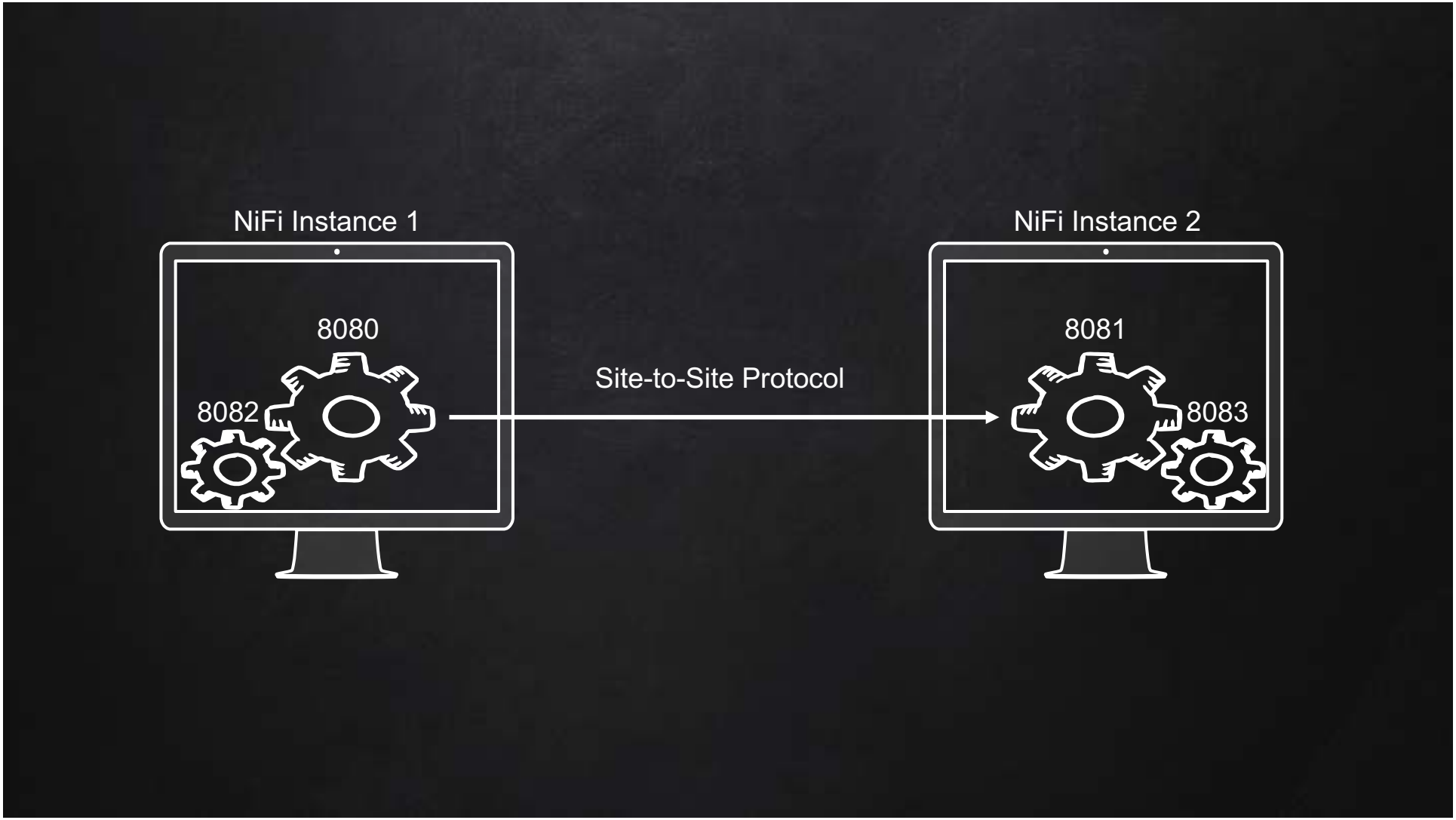

Remote Monitoring NiFi
Using
Reporting Task

Remote Monitoring Using Reporting Task?

- ✗ It's practically impossible for someone to sit in front of a computer and monitor for any errors using the UI.
- ✗ NiFi provides various Reporting Tasks to monitor your NiFi instance remotely.
- ✗ We will explore "SiteToSiteBulletinReportingTask" and "SiteToSiteMetricsReportingTask."

What is Site-to-Site?

- ✗ Site-to-Site is a protocol used to send data from one NiFi instance to another NiFi instance, in a smooth, effective, and secure way.
- ✗ You can also use the Site-to-Site protocol to transmit data from any application which produces data to a NiFi instance.


NiFi Instance 1
8080
8082
Site-to-Site Protocol
NiFi Instance 2
8081
8083


Data Provenance
in NiFi

What is Data Provenance?

- NiFi keeps a comprehensive track of the data by recording all the events applied on a flowfile starting from its ingestion point till it's removed from the data flow.

- As the data is processed through the system, NiFi captures all the details like when the data got transformed, split, routed, aggregated, or dispersed to other endpoints.

- All these information's are stored and indexed in a separate repository called the Provenance Repository.

Why Data Provenance?

- ✗ To Debug your Data Flow at Production level.
- ✗ To Identify the Origin, Destination & Transformation happened to your data.

Apache NiFi Architecture

High Level Overview of Key Apache NiFi Features

Overview of NiFi Registry

Version Control for your Data Flow ! !

What is NiFi Registry?

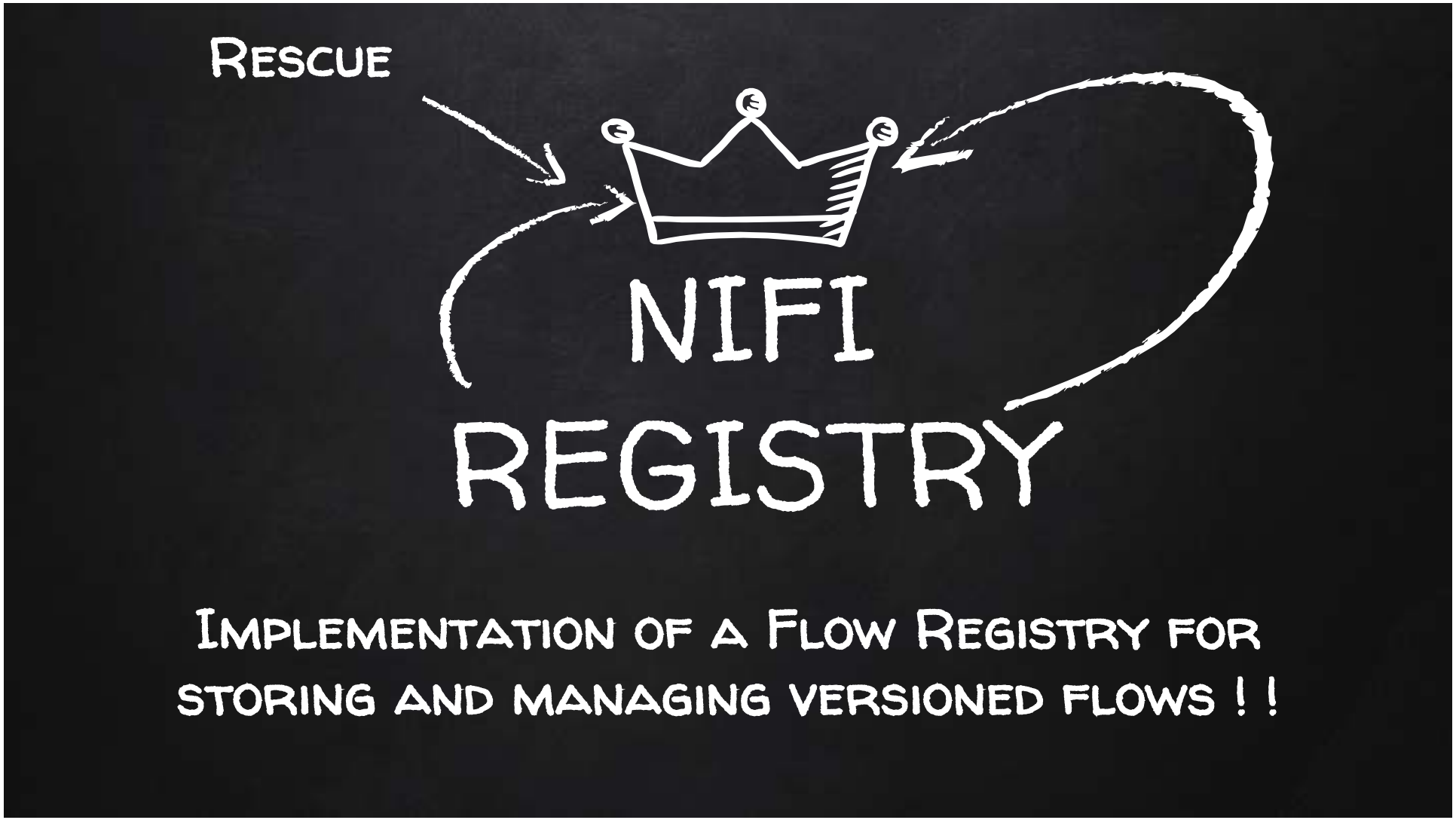
- NiFi Registry is a complementary project that provides a central location for storing and managing shared resources across one or more NiFi instances.
- It is a separate sub-project of Apache NiFi.
- This means we must download it separately and it will follow a different release cycle and version.

Why NiFi Registry?

- ✗ Assume you are working in a team, and more than one person is working on a Data Flow, then version control of the flow becomes complex.
- ✗ For a very long-time, people used NiFi templates to enable version control and it's such a pain.
- ✗ NiFi template is not created or optimized for doing version control in the first place.

Why NiFi Registry? (Cont.)

- ✗ We have to manually download the template and commit your changes to any other version control tools like TFS or SVN or GIT.
- ✗ Merge is going to be a nightmare.
- ✗ More complexity is involved when we need to take the latest version of the template from the repository and merge it with your un-committed local version.


Rescue
NIFI
REGISTRY
Implementation of a Flow Registry for
storing and managing versioned flows ! !

Installation of NiFi Registry


Configuring
NiFi & NiFi Registry
to Enable
Version Control

What is a Bucket?

- A bucket is nothing but logical segregation of the Data Flow, and one bucket can have more than one Flow associated with it.
- You can consider a Bucket like a Folder available in the File System where we will keep related files inside it.
- In our case, the files are nothing but the related flows.

Configuring NiFi Registry with Multiple NiFi Instances

Current Limitation of NiFi Registry

- ✗ You can't take the latest version of the flow without reverting your local changes.
- ✗ To put it in simpler words, you can't merge your local changes with a newer version available in the registry.
- ✗ You can only revert your local changes and take the latest version and redo your changes again and commit the same.

Configuring NiFi Registry to Enable Git Persistence


Overview on
NiFi Clustering

Why NiFi Cluster?

- ✗ Sometimes you may find it difficult to use one NiFi instance to process huge amount of data.
- ✗ Instead you can use multiple NiFi servers to process large data sets.
- ✗ This can be done by segregating the large data set into multiple smaller data sets and sending these small data sets to different servers to process them separately.

Common Problems in Clustering

- ✗ Each time when you want to change or update the dataflow, you must make those changes on each server and then monitor each server separately.

- ✗ Segregating big data set into multiple smaller datasets and processing it separately using different servers will have its own complexity.

Features of NiFi Cluster

- In NiFi we can cluster multiple NiFi servers and each server in the cluster can perform the same set of tasks on the data, but each can operate on a different set of data.

- In NiFi cluster, if you make change in one node it automatically gets replicated to all the nodes of the cluster.

- Using a single interface, the Data Flow Manager can monitor the health and status of all the nodes.

How NiFi Cluster Works?

- ✗ NiFi follows a Zero-Master Clustering paradigm.
- ✗ In NiFi, we can use Apache ZooKeeper for Cluster Management, and any failover is handled by ZooKeeper.

What is ZooKeeper?

- ZooKeeper is an open-source server which enables highly reliable distributed coordination.
- Many open source software like SOLR uses ZooKeeper for its Cluster Management.

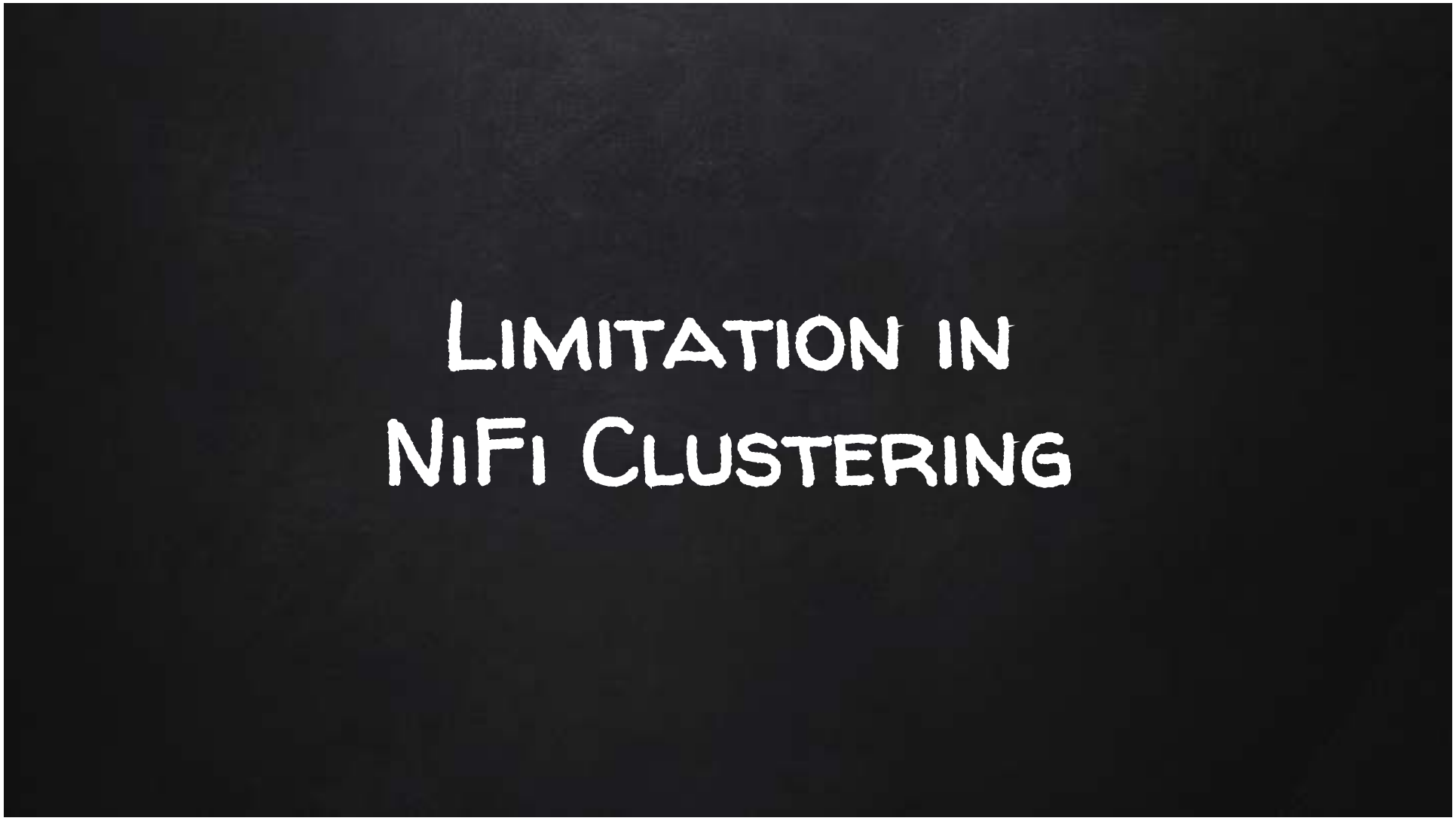
How NiFi Cluster Works? (Cont.)

- ZooKeeper elects one of the NiFi nodes as the Cluster Coordinator and all nodes in the cluster will send status information to this node.
- The Cluster Coordinator is responsible to disconnect nodes that do not emit any heartbeat status for some amount of time.
- When a new node opts to join the cluster, that node must first connect to the currently elected Cluster Coordinator in order to obtain the latest flow.

How NiFi Cluster Works? (Cont.)

- ✗ If the Cluster Coordinator decides to allow the node to join the cluster, the current flow is provided to that node, and that node will then able to join the cluster.

- ✗ If the version of the flow configuration available in the new node differs from the version available in the Cluster Coordinator, the node will not be able to join the cluster.


Limitation in
NiFi Clustering

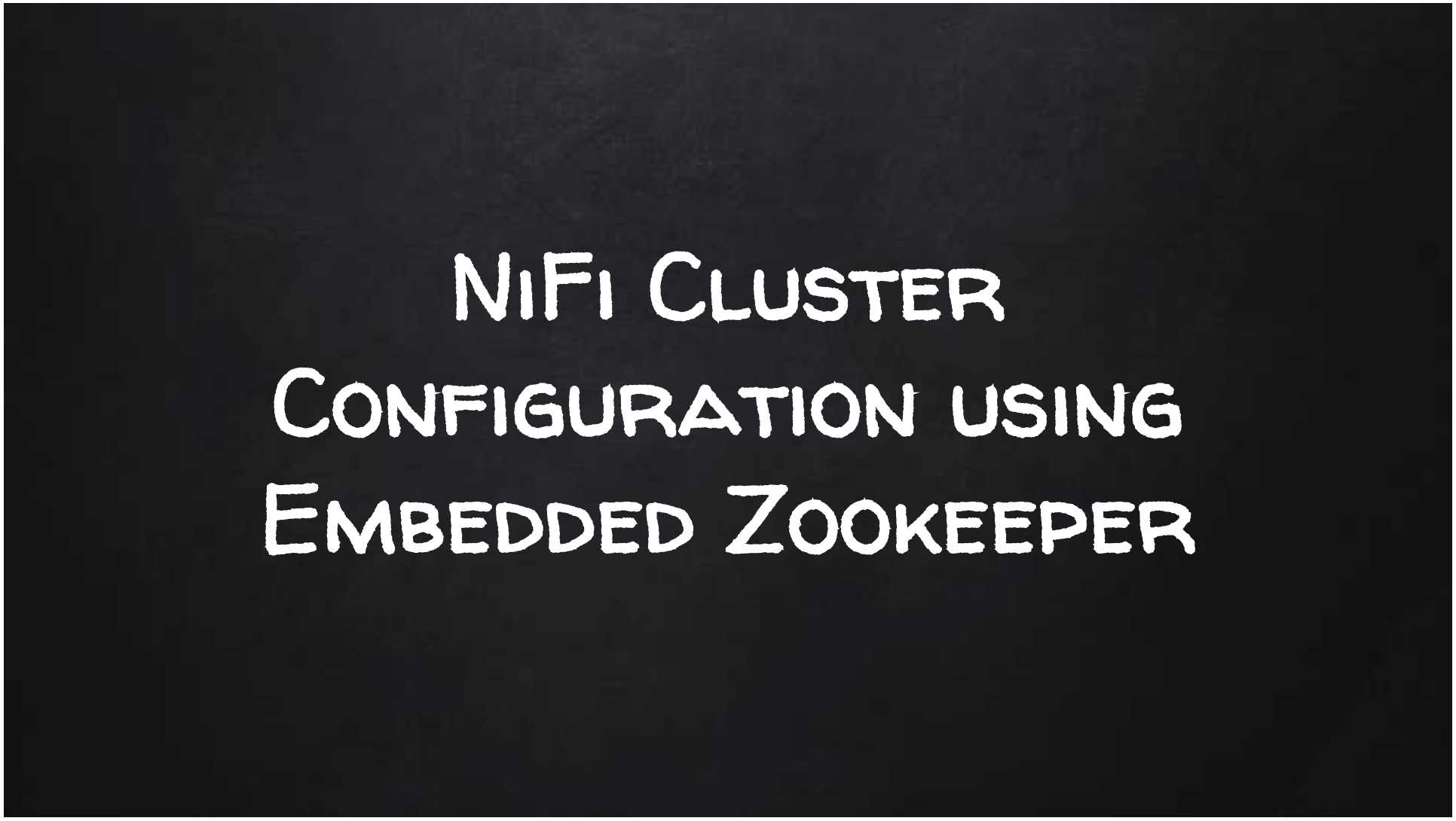
Limitation in NiFi Clustering

- ✗ In a NiFi cluster, data is **"Distributed"** for processing, but it's not **"Replicated."**
- ✗ In a NiFi Cluster when we split large data set into multiple smaller data sets, each node will be given the smaller data set it needs to process, and the same node keeps this data in its disk storage.
- ✗ A copy of this data is not maintained or replicated in any other nodes.

Side-Effect of not having Data Replication

- ✗ In a NiFi cluster, if one node goes down, we have a provision to offload that node and the data in that node will be distributed to other active nodes slowly, provided that node is still connected to the network.

- ✗ But, if the node completely goes out of the network for some reason, says someone pulled the network cable of that machine. The data inside that node will not be processed or distributed to other active nodes until it comes back to the network.


NiFi Cluster
Configuration using
Embedded Zookeeper

Embedded Zookeeper

- ✗ Each NiFi instance is bundled with a ZooKeeper instance inside it.
- ✗ We will be using three NiFi instances and the three embedded Zookeeper instances available within it.
- ✗ The only key prerequisites here is you should have more than one machine available to follow along and connectivity between these machines must be enabled.

Digital Ocean Droplets

- ✗ I will not be using my local machine like I used to do previously. Instead I will be using three "**Digital Ocean Droplets**".

- ✗ If you are coming across Digital Ocean Droplets for the first time, just remember it's something similar to "**AWS EC2**".

- ✗ To put it simpler, I have loaned commodity machines according to my required specification and I will be billed based on the number of hours I have used these machines.

NiFi Cluster Configuration using External Zookeeper

Pitfall Using Embedded Zookeeper

- ✗ Using Embedded Zookeeper is perfectly fine and it can be used in production.
- ✗ Drawback in this approach is, both NiFi and Zookeeper are running in the same server and if the server goes down, we will lose the NiFi and Zookeeper instance at the same time.
- ✗ Using an external Zookeeper, we can run the NiFi and the Zookeeper instances in different machines.

Overview on NiFi Custom Processor

Hope for the Best !! Plan for the Worst !!

NiFi Processor Recap

- ✗ We have tons of Data Source and Data Sink Processors.
- ✗ At the time of recording this video, NiFi has 280+ Processors.
- ✗ Each Processors in NiFi is unique in it's own way.
- ✗ NiFi have processors for all your data needs.

Why Custom Processors?

- ✗ There will be situations where you will not be able to use any of these in-built processors which comes bundled with NiFi to cater your requirements.
- ✗ This is where the Custom Processors of NiFi comes in handy.

The Million Dollar Question

- ✗ NiFi provides maven archetypes to create our own processor or controller service which is compatible and easy to include as part of our data flow.

- ✗ Whenever someone tells, you can extend their tool. You will get tons of questions regarding how easy it's going to be and how can we can migrate this custom code when a new version of the tool arrives.

Why its Simple?

- A processor in NiFi takes an input flow file and do some processing on top of it and produces an output flow file. It can also have some properties using which you can configure the way the processor processes the input data.

- The FlowFile abstraction of NiFi makes it so simple and easy to create custom processors.


Rescue
Maven
Archetype
Auto Generates Everything We Need ! !

Advantage of Using Maven Archetype

- ✗ The NiFi custom processor maven archetype makes your life easy by auto-generating the code required for us to get started.
- ✗ The auto-generated code has everything we need, and all we have to do is to write your custom Java code inside it to process the input flow file to produce the output flow file.

Our First Custom Processor

What is a NAR?

- ✗ NAR is NiFi Archive file.
- ✗ It's similar to a WAR or JAR file.
- ✗ We can deploy the NAR files in NiFi, similar to deployment of war files in tomcat or any other application servers.