Spring XD

Distributed Mode

Setup and Configuration

Distributed Mode

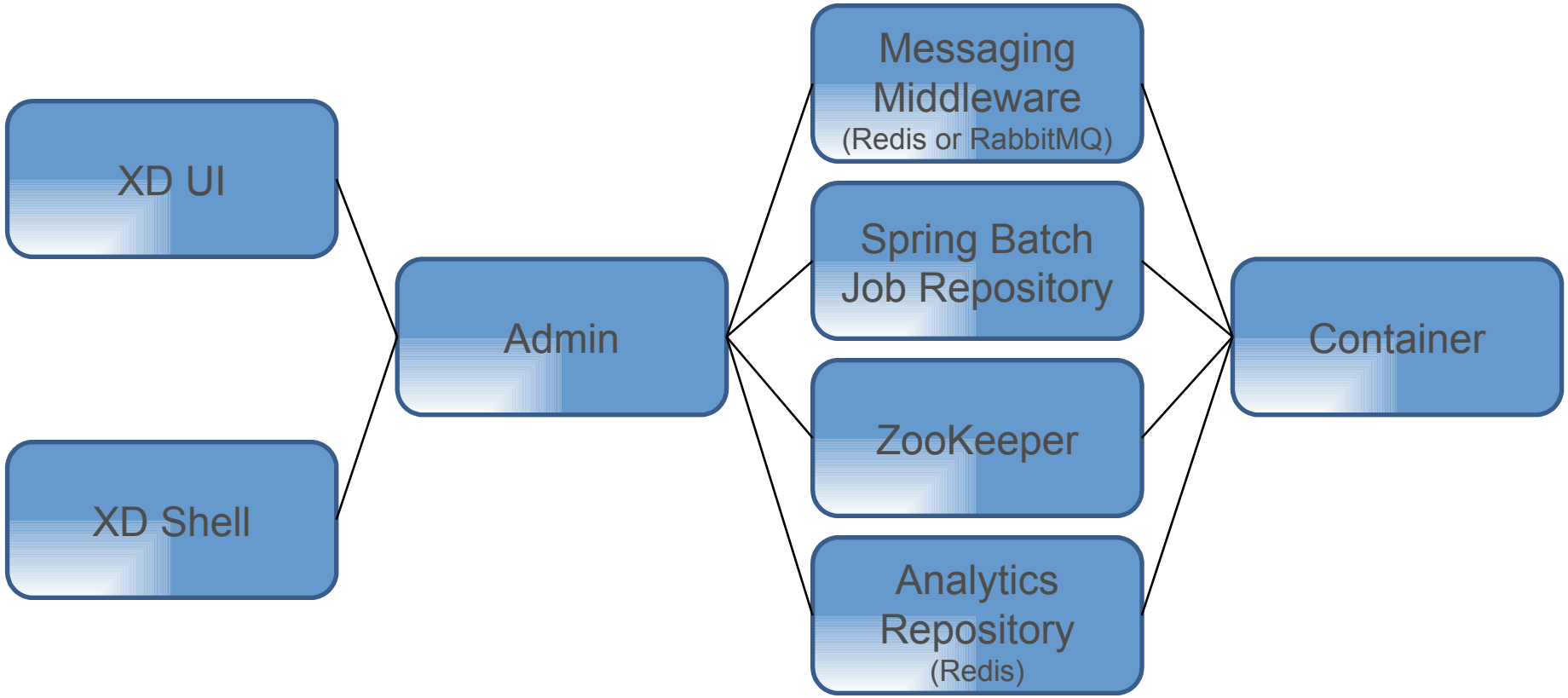
- Why Distributed Mode?
- XD Distributed Runtime (DIRT) Installation
  - Redhat/CentOS
  - Yarn
- Customizing the Configuration
- Lab

Why Distributed Mode?

- Scalability
  - Run multiple streams and jobs across multiple containers
  - Improved speed
  - Able to handle data sets that don't fit into a single JVM
- High availability
  - Provides container server redundancy in the event of an outage
- Pluggable Architecture
  - Permits users to plug in different technologies for the transport and job repositories

Distributed Deployment

- Consists of multiple components



Prerequisites for Distributed Deployment

- Recall that Spring XD is an amalgamation of products:
  - Spring Integration
  - Spring Batch
  - Apache ZooKeeper
  - Redis or RabbitMQ
  - RDBMS
  - tcServer

Prerequisites for Distributed Deployment

- JDBC compliant RDBMS for batch jobs
- RabbitMQ (preferred), or Redis for data transport
- Redis for analytics repository
- ZooKeeper for distributed configuration synchronization
  - ZooKeeper is an Apache project that Spring XD leverages for managing the distributed components

Distributed Components

- Admin
  - Manages Stream and Job deployments and other end user operations
  - Provides REST services to access runtime state, system metrics, and analytics
- Container
  - Hosts deployed modules (stream processing tasks), batch jobs
- ZooKeeper
  - Provides all runtime information for the XD cluster.
  - Tracks running containers, in which containers modules and jobs are deployed, stream definitions, deployment manifests, etc.

Distributed Components

- Spring Batch Job Repository Database
  - Required for storing job metadata, status, etc.
- Transport (Messaging Middleware)
  - Required for data transport
  - Pluggable
  - Current support for Rabbit MQ and Redis, for stream and job processing
  - Rabbit MQ is recommended for greater reliability
  - Must be deployed to separate server

Distributed Components

- Analytics Repository
  - Required for storing metrics
  - XD currently uses Redis
- XD UI
  - Web administration interface
- XD Shell
  - Command line administration interface

Spring Batch Job Repository Database

- HSQLDB in-memory database included for development and testing
  - To start HSQLDB:

```
$ cd hsqldb/bin
$ ./hsqldb-server
```

- Separate RDBMS required for production
- Edit the **`spring:datasource`** section in **`xd/config/servers.yml`**
  - Stubs included for MySQL, Postgres
  - Customize as necessary for others

Spring Batch Job Repository Database

```
#spring:
#  datasource:
#    url: jdbc:hsqldb:hsql://${hsql.server.host:localhost}:${hsql.server.port:9101}/
${hsql.server.dbname:xdjob}
#    username: sa
#    password:
#    driverClassName: org.hsqldb.jdbc.JDBCDriver
#    validationQuery: select 1 from INFORMATION_SCHEMA.SYSTEM_USERS

#Config for use with MySQL - uncomment and edit with relevant values for your
environment
#spring:
#  datasource:
#    url: jdbc:mysql://yourDBhost:3306/yourDB
#    username: yourUsername
#    password: yourPassword
#    driverClassName: com.mysql.jdbc.Driver
#    validationQuery: select 1

#Config for use with Postgres - uncomment and edit with relevant values for your
environment
#spring:
#  datasource:
#    url: jdbc:postgresql://yourDBhost:5432/yourDB
#    username: yourUsername
#    password: yourPassword
#    driverClassName: org.postgresql.Driver
#    validationQuery: select 1
```

ZooKeeper

- Must be installed separately
- Installation instructions can be found at

  http://zookeeper.apache.org/doc/trunk/zookeeperStarted.html
- Configure connection in the ZooKeeper Properties section in **`xd/config/servers.yml`**
- Minimum 3 members recommended for production

```
#ZooKeeper properties
# client connect string: host1:port1,host2:port2,...,hostN:portN
#zk:
#  client:
#     connect: localhost:2181
```

Analytics Repository

- Redis by default
- Linux/Mac version included
- Download and install from http://redis.io/download
- Windows version available from Microsoft Open Tech group: https://github.com/MSOpenTech/redis
- Configure connection in the **`Redis Properties`** section in **`xd/config/servers.yml`**
- If using Redis for messaging middleware, also configure **`XD data transport`**

Transport

- RabbitMQ or Redis
- If RabbitMQ, download and install appropriate RabbitMQ version from http://www.rabbitmq.com/download.html
- Configure connection in the RabbitMQ Properties section in **`xd/config/servers.yml`**

```
# RabbitMQ properties
#spring:
#  rabbitmq:
#   addresses: localhost:5672
#   username: guest
#   password: guest
#   virtual_host: /
#   useSSL: false
#   sslProperties:
```

Transport

- If using RabbitMQ for messaging middleware, also configure **`XD data transport`** section

```
#XD data transport (default is redis for distributed, local for single node)
#xd:
#  transport: rabbit

#  messagebus:
#    rabbit:
#      default:
#        ackMode:                    AUTO
           # Valid: AUTO (container acks), NONE (broker acks), MANUAL (consumer acks).
           # Upper case only.
           # Note: MANUAL requires specialized code in the consuming module and is
           # unlikely to be used in an XD application. For more information, see
           # http://docs.spring.io/spring-integration/reference/html/amqp.html#amqp-inbound-ack
#        backOffInitialInterval:    1000
#        backOffMaxInterval:        10000
#        backOffMultiplier:         2.0
#        concurrency:               1
#        deliveryMode:              PERSISTENT
```

Transport

```
#         maxAttempts:                3
#         maxConcurrency:             1
#         prefix:                     xdbus.
            # prefix for queue/exchange names so policies (ha, dle etc.) can be
applied
#         prefetch:                   1
#         replyHeaderPatterns:        STANDARD_REPLY_HEADERS,*
#         requestHeaderPatterns:      STANDARD_REQUEST_HEADERS,*
#         requeue:                    true
#         transacted:                 false
#         txSize:                     1
```

Starting Spring XD in Distributed Mode

- Start ZooKeeper, Redis, and RabbitMQ
- Start the container and admin servers

```
xd/bin>$ ./xd-admin
xd/bin>$ ./xd-container
```

- To override the XDAdmin listening port:

```
xd/bin>$ ./xd-admin --httpPort <httpPort>
```

Running Spring XD on YARN

- YARN is the Hadoop NextGen MapReduce framework
- Consists of multiple components for scheduling, managing, and running map-reduce jobs in Hadoop
- Run Spring XD on YARN if you have an existing Hadoop cluster

Prerequisites for YARN Distributed Deployment

- Requires all components necessary for distributed mode:
  - Redis or RabbitMQ for data transport
  - ZooKeeper
  - Spring Batch Job RDBMS Repository
- Requires Hadoop cluster based on Apache Hadoop 2.2.0 or later, including
  - Apache Hadoop 2.2.0
  - Pivotal HD 2.0
  - Hortonworks HDP 2.1
  - Cloudera CDH5

Prerequisites for YARN Distributed Deployment

- JDBC compliant RDBMS for batch jobs
- RabbitMQ or Redis for data transport
- ZooKeeper

Installing on YARN

- Download and unzip **`spring-xd-<version>-yarn.zip`**
- Configuration is in **`config/servers.yml`**
- Need to configure
  - Hadoop settings
  - Transport choice
    - Redis/RabbitMQ settings
  - ZooKeeper settings
  - JDBC datasource properties
- Depending on Hadoop distribution, may need to change
  - **`siteYarnAppClasspath`**, **`SiteMapReduceAppClasspath`**

Installing on YARN – XD Options

- Define number of
  - admin servers
  - containers
  - HDFS directory where Spring XD binary and config files will be stored

```
spring:
    xd:
        adminServers: 1
        containers: 3
    yarn:
        applicationDir: /xd/app/
```

Installing on YARN – Hadoop Settings

- Specify
  - YARN Resource Manager host
  - HDFS URL

```
# Hadoop properties
spring:
  hadoop:
    fsUri: hdfs://localhost:8020
    resourceManagerHost: localhost
```

Installing on YARN – ZooKeeper Settings

- Connection properties

```
---
#ZooKeeper properties
# client connect string: host1:port1,host2:port2,...,hostN:portN
zk:
  client:
     connect: localhost:2181
```

Installing on YARN – Transport Options

- Specify transport type
  - Redis
  - RabbitMQ

```
# Transport used
transport: redis
```

- Specify transport connection properties

```
---
# Redis properties
spring:
  redis:
   port: 6379
   host: localhost
```

Installing on YARN – JDBC Configuration

- Specify JDBC connection properties for
  - Batch jobs
  - JDBC sink

```
---
#Config for use with MySQL - uncomment and edit with relevant values for your
environment
spring:
  datasource:
    url: jdbc:mysql://yourDBhost:3306/yourDB
    username: yourUsername
    password: yourPassword
    driverClassName: com.mysql.jdbc.Driver
```

Installing on YARN – Modules

- To customize modules
  - Edit **`config/modules.yml`**
  - Add it to existing **`config/modules-config.zip`** by running:

```
$ jar -uf modules-config.zip modules.yml
```

- To add custom modules
  - Replace **`custom-modules.zip`** with your own archive
  - Place module definitions into modules directory within the archive
  - Module definitions must follow Spring XD module semantics

Installing on YARN – Starting Spring XD

- Push the Spring XD application binaries and config to HDFS

```
$ ./bin/xd-yarn push
```

- Submit the Spring XD Admin server

```
$ ./bin/xd-yarn start admin
```

- Submit the Spring XD container

```
$ ./bin/xd-yarn start container
```

- Check the status, verify xd-admin and xd-container are running

```
$ yarn application -list
```

Customizing the Configuration

- Override the default location of the Spring XD install by changing the **`XD_HOME`** environment variable

```
$ export XD_HOME=<XD install directory>
```

Summary

- Distributed runtime allows for scalability, high availability, and pluggable technologies for Spring XD components
- Each component can be installed on separate servers
- Possible to install and run Spring XD on YARN
- When running Spring XD in distributed mode, it is necessary to individually install repository datastores, messaging middleware, and ZooKeeper

Lab

Installing Spring XD in Distributed Mode