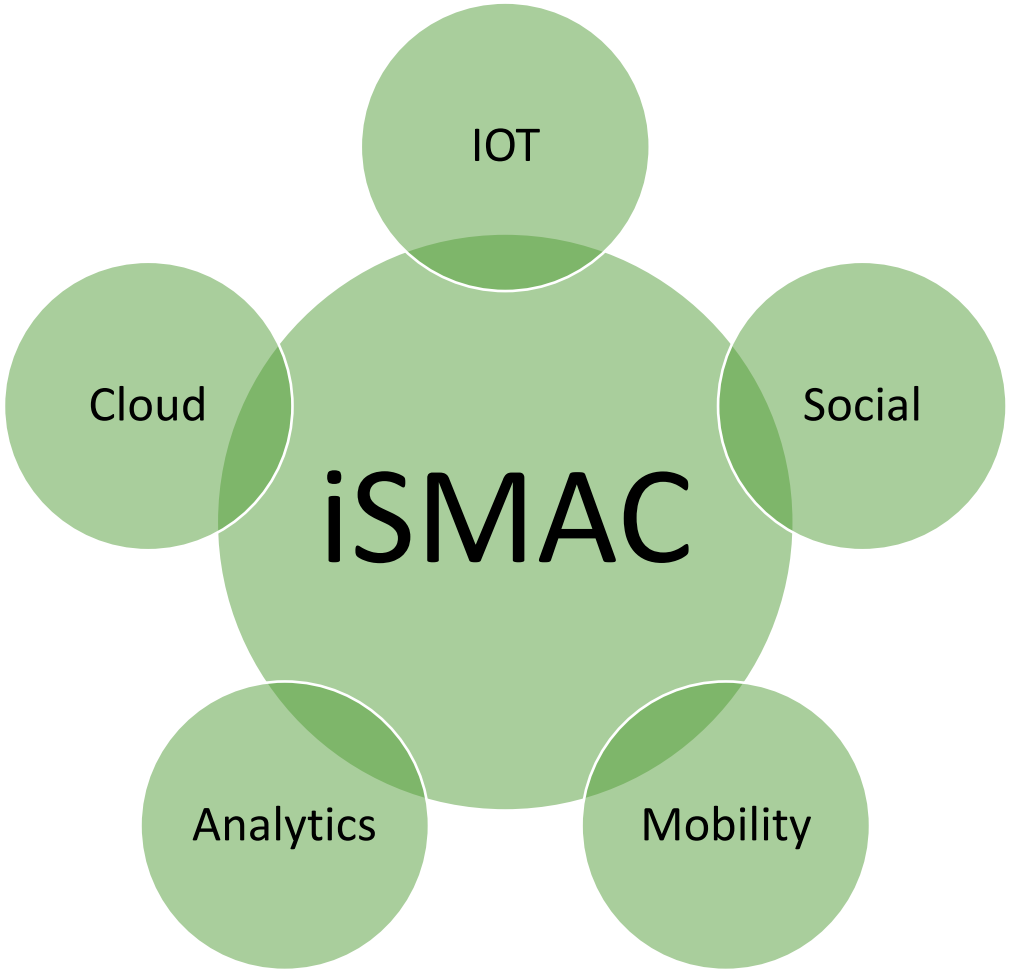
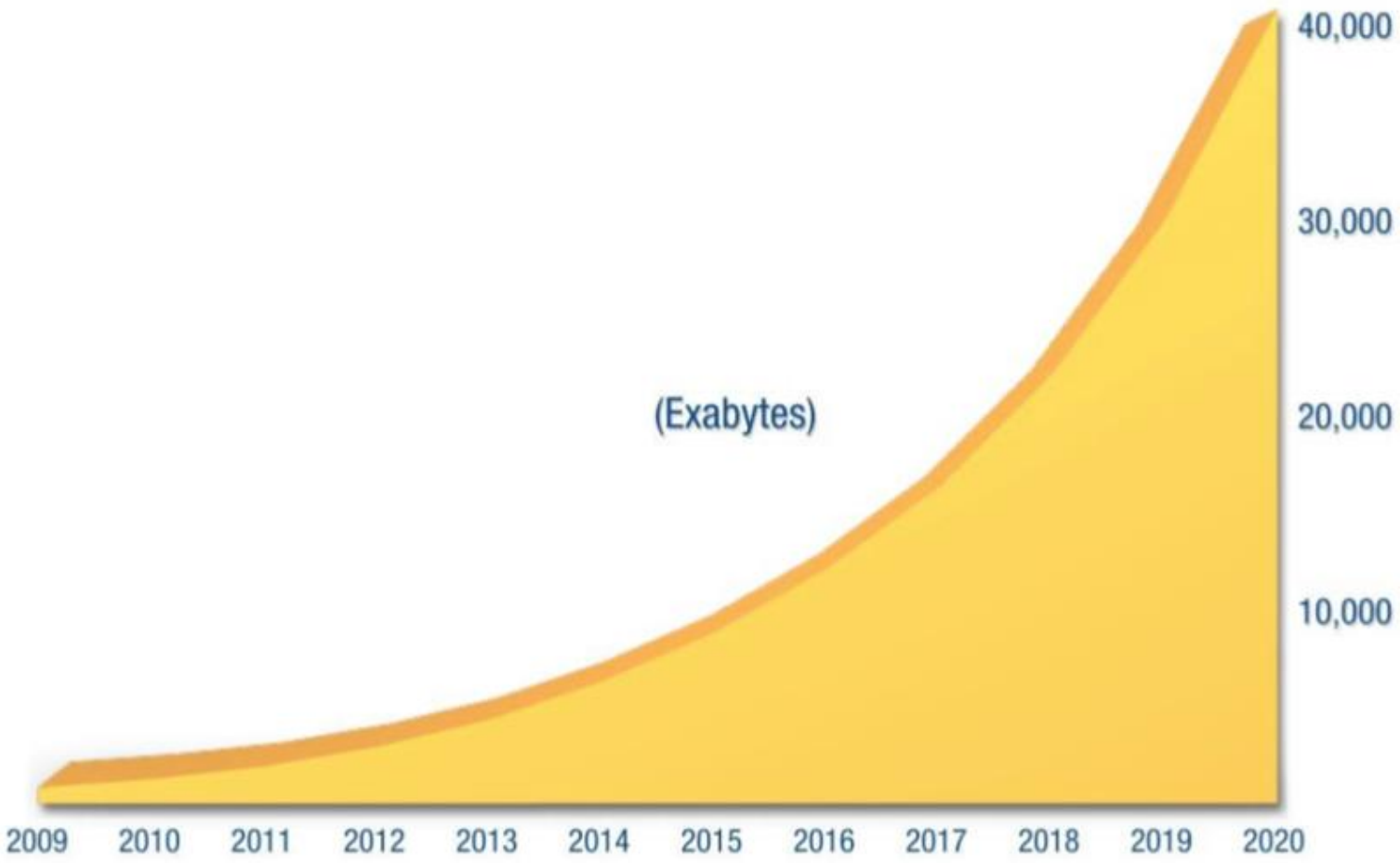
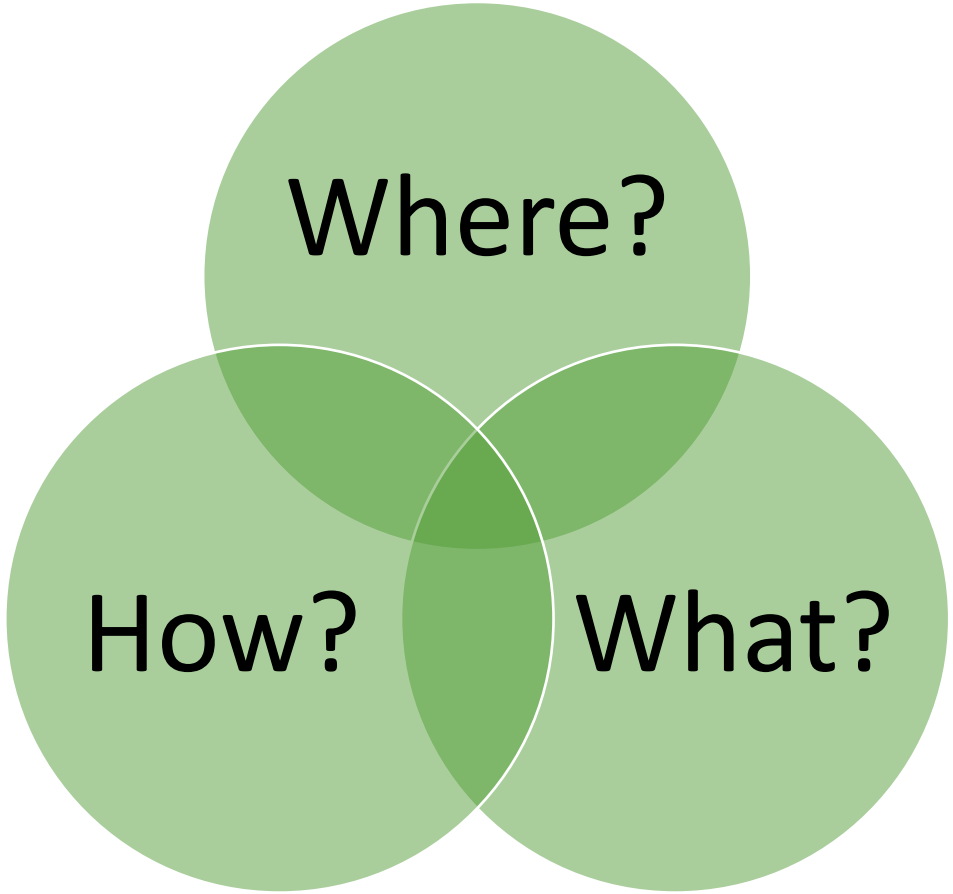
iSMAC



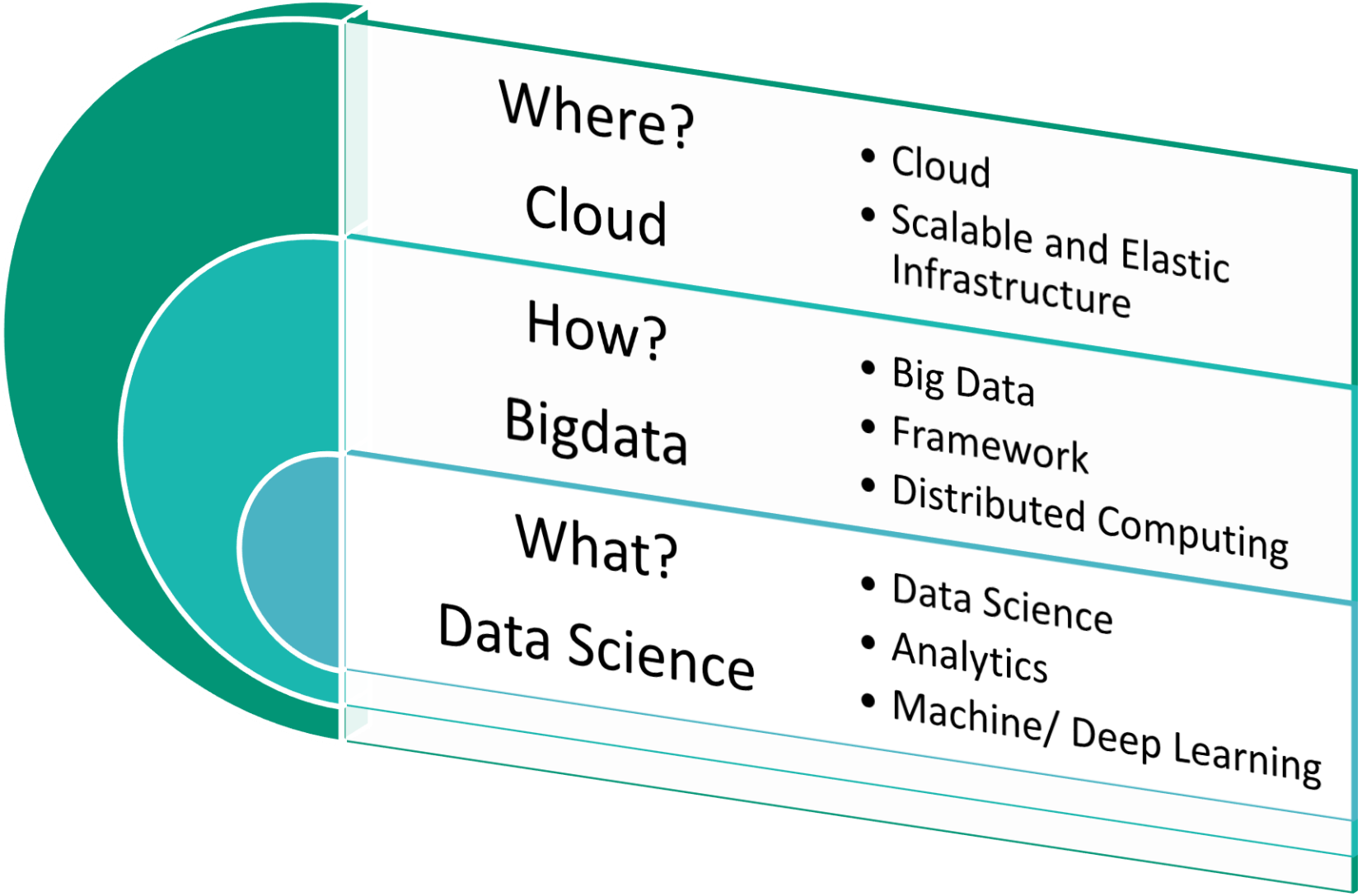
Data Growth


40,000
30,000
20,000
10,000
(Exabytes)
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020

Magic Triangle



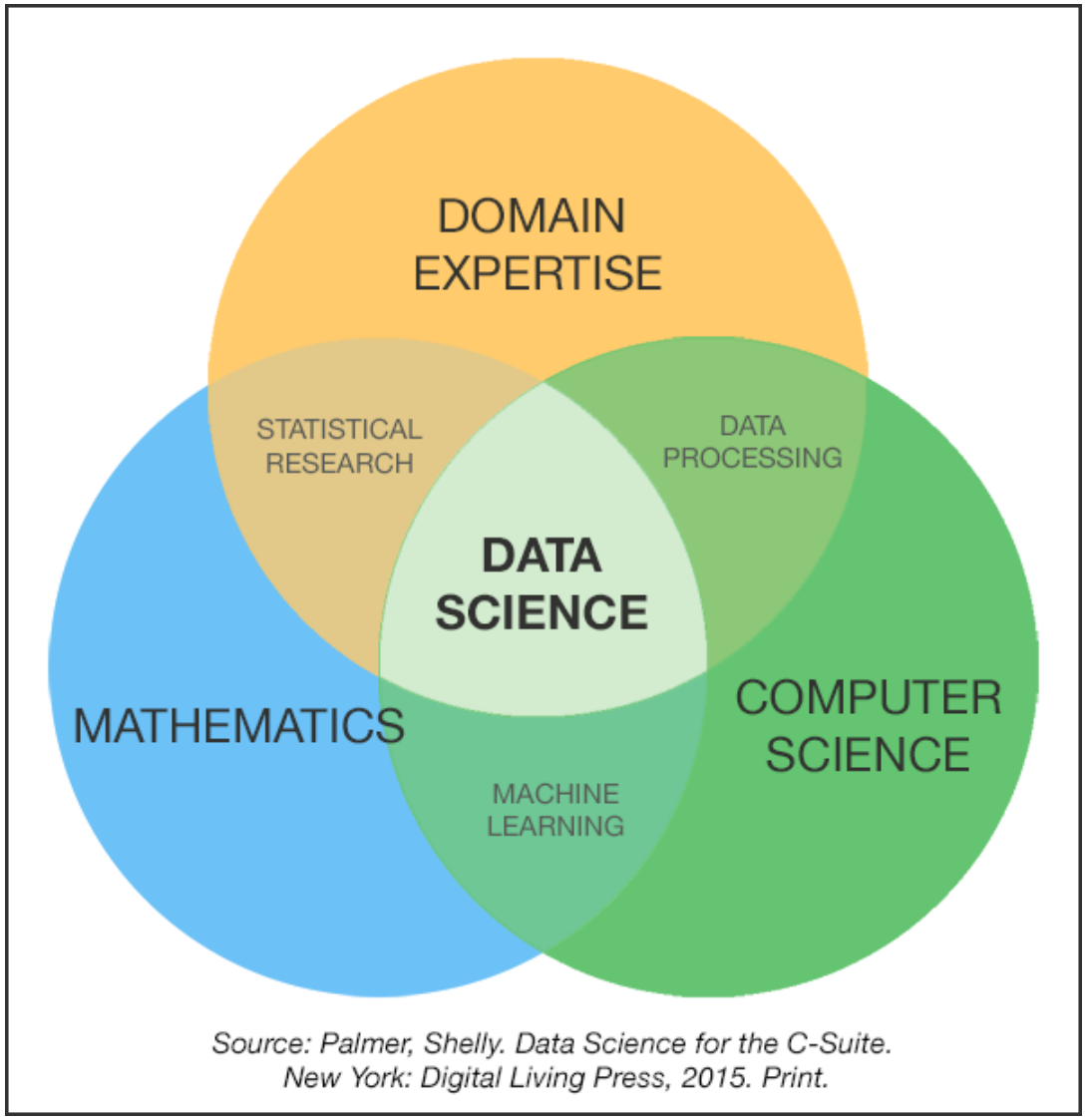
Where? How? What?



Spectrum of Analytics

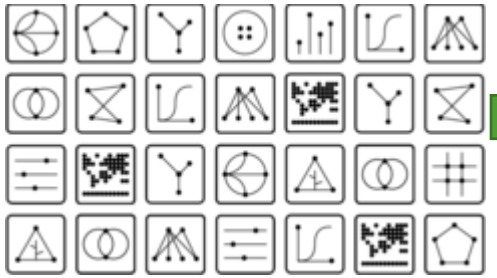
Descriptive	• What happened? - PAST • Descriptive Statistics • Data Clustering
Diagnostic	• Why did it happen? / Why is it happening? • Sensitivity Analysis
Predictive	• What will happen? • Linear and Logistic Regression
Prescriptive	• What should I do? What should happen? • Simulation • Non Linear Programming

Data Science Venn



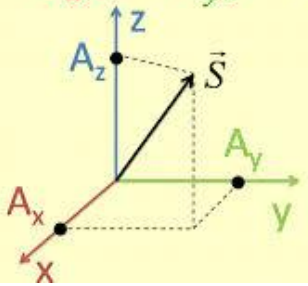
Source: Palmer, Shelly. Data Science for the C-Suite. New York: Digital Living Press, 2015. Print.

Image Source : https://pixabay.com



Algorithm

$$\vec{S} = A_x\hat{x} + A_y\hat{y} + A_z\hat{z}$$

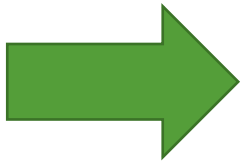


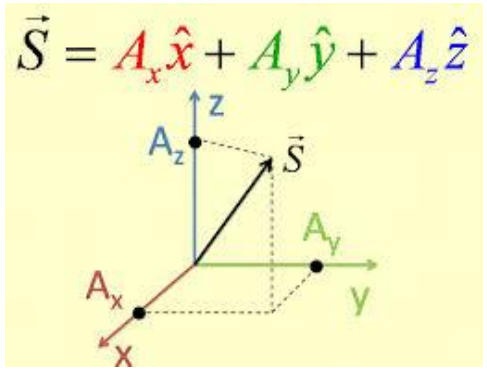
Model

Birds

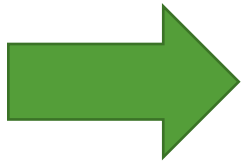
Animals

Human

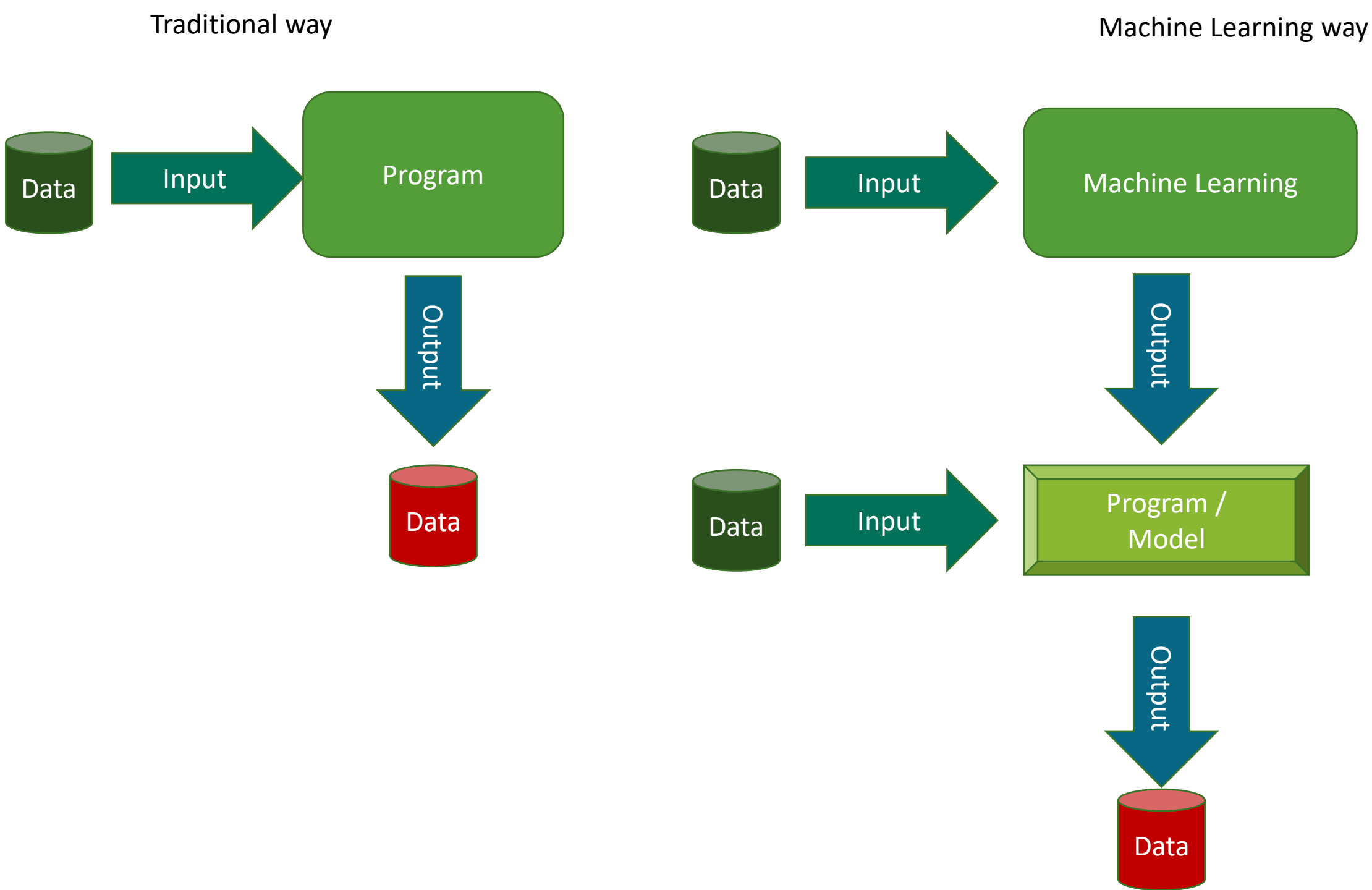


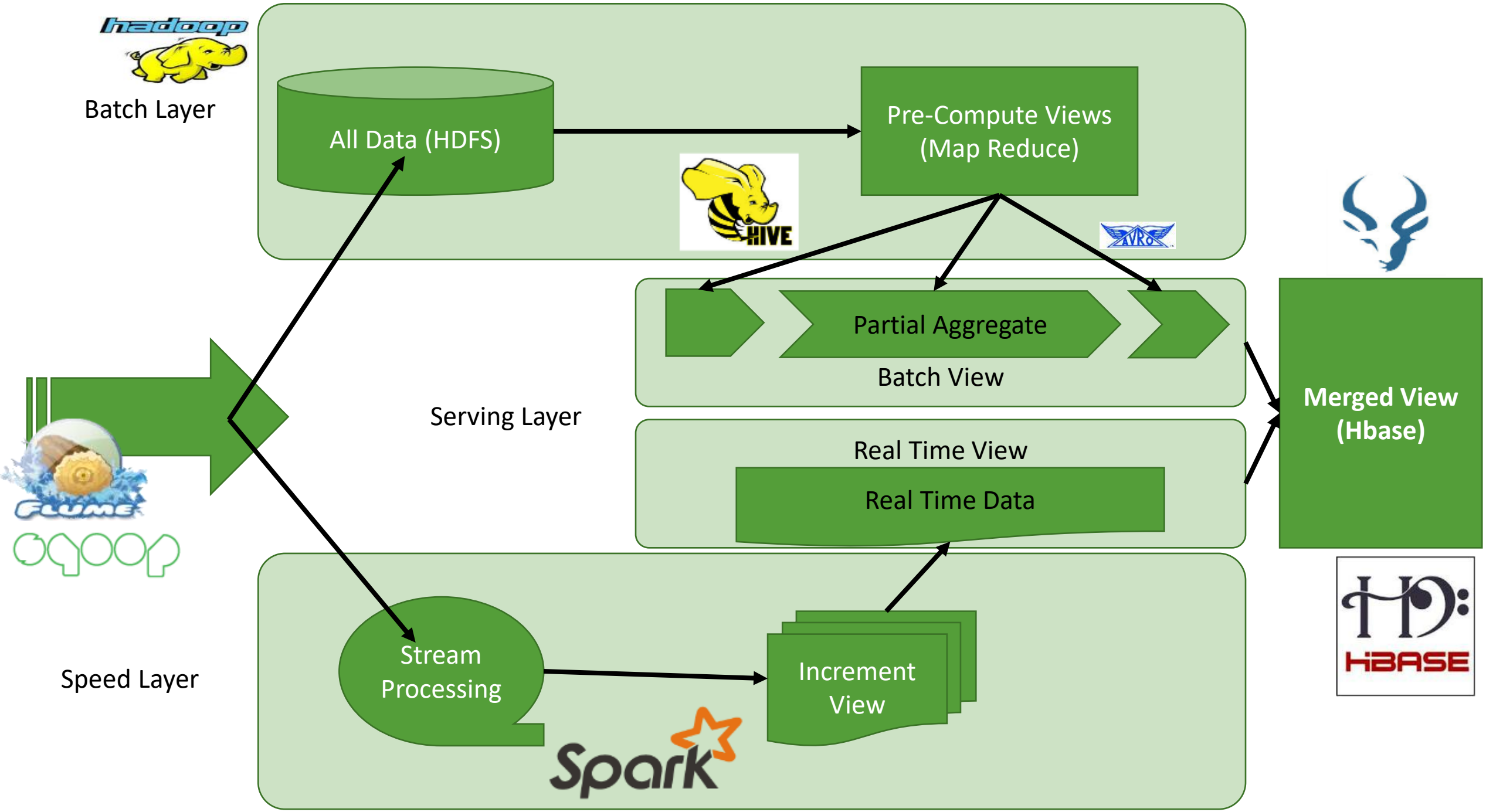


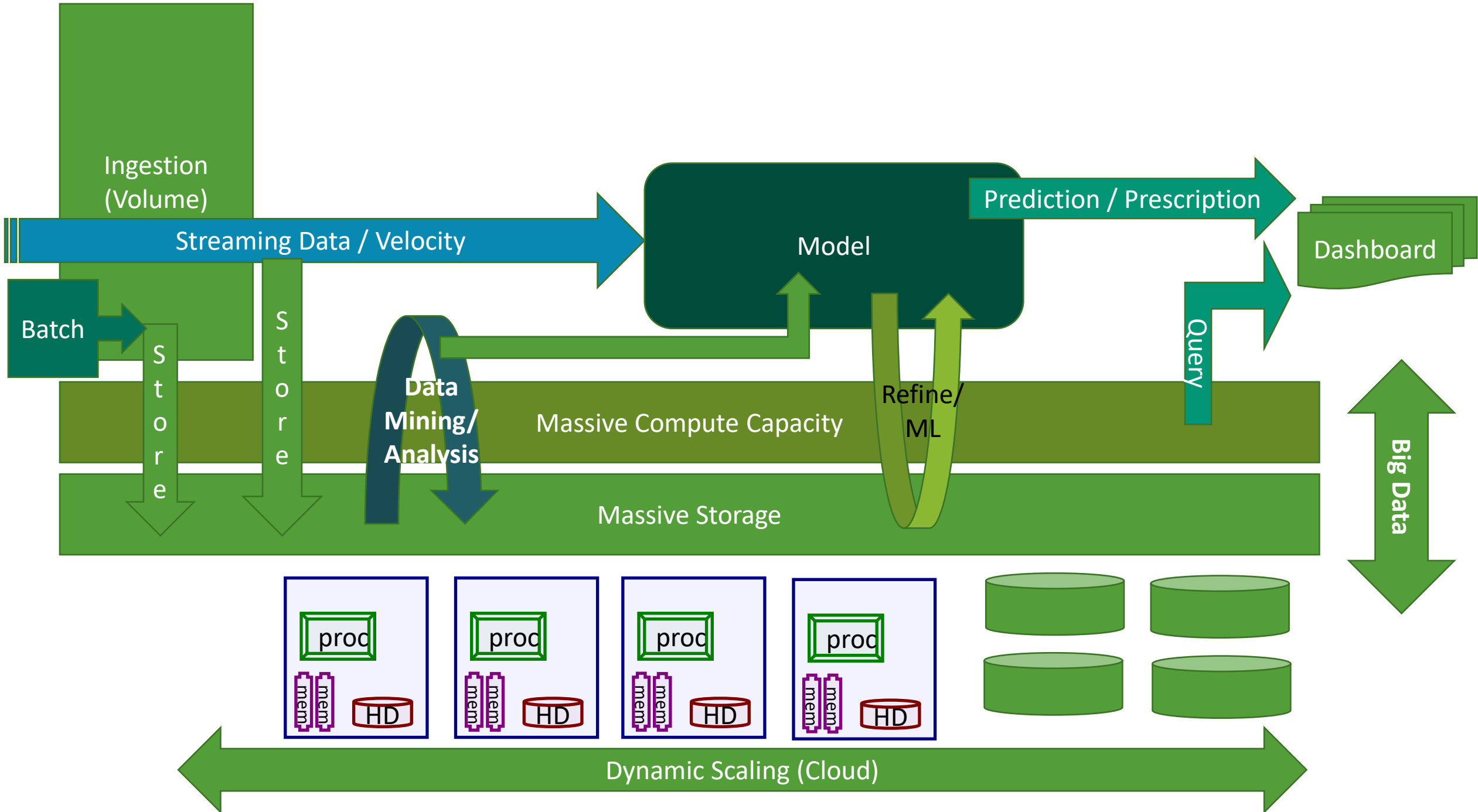
Model



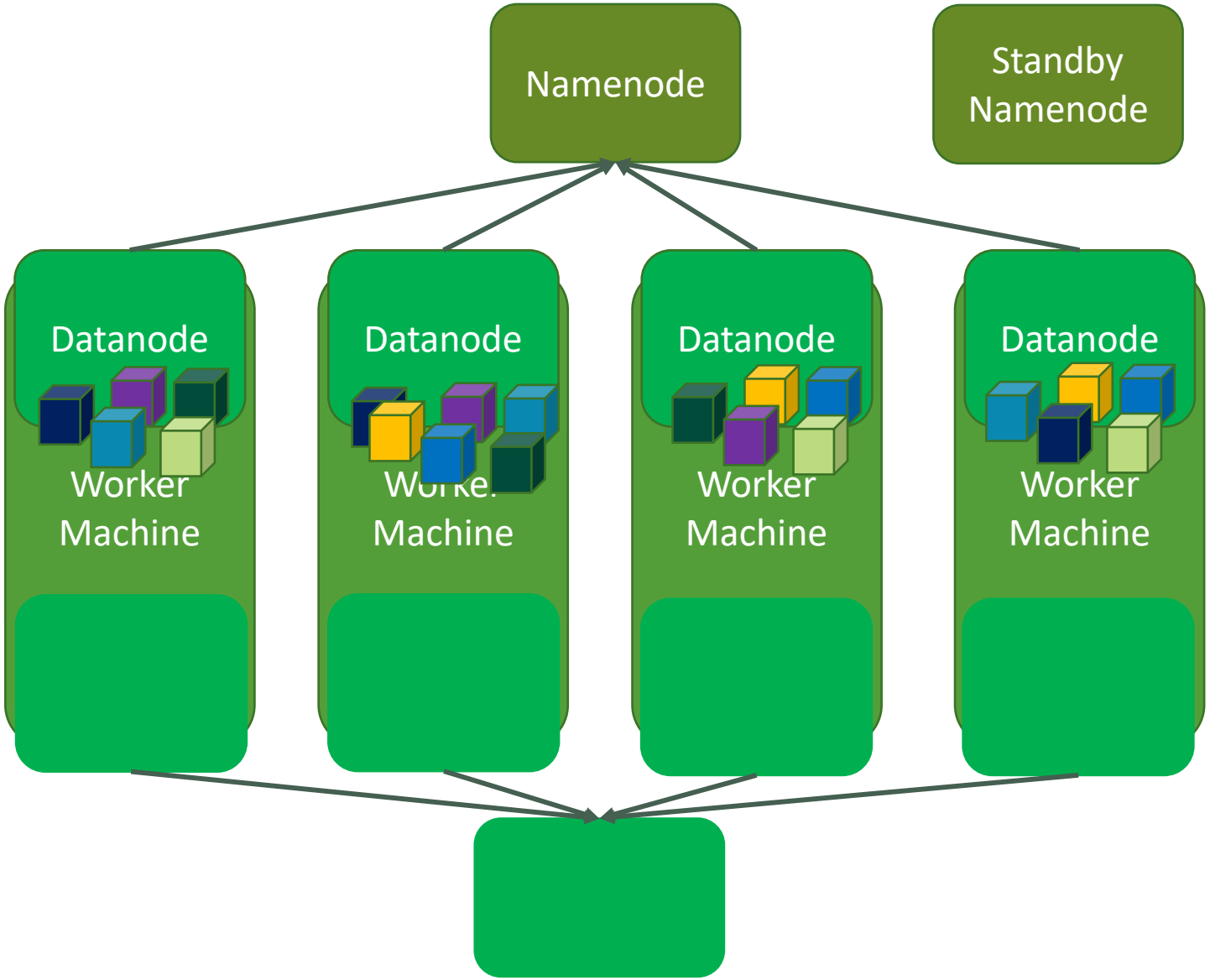
Bird – 7%
Animal – 86%
Human – 6%
Don’t know – 1%


Traditional way
Data
Input
Program
Output
Data
Machine Learning way
Data
Input
Machine Learning
Output
Data
Input
Program /
Model
Output
Data

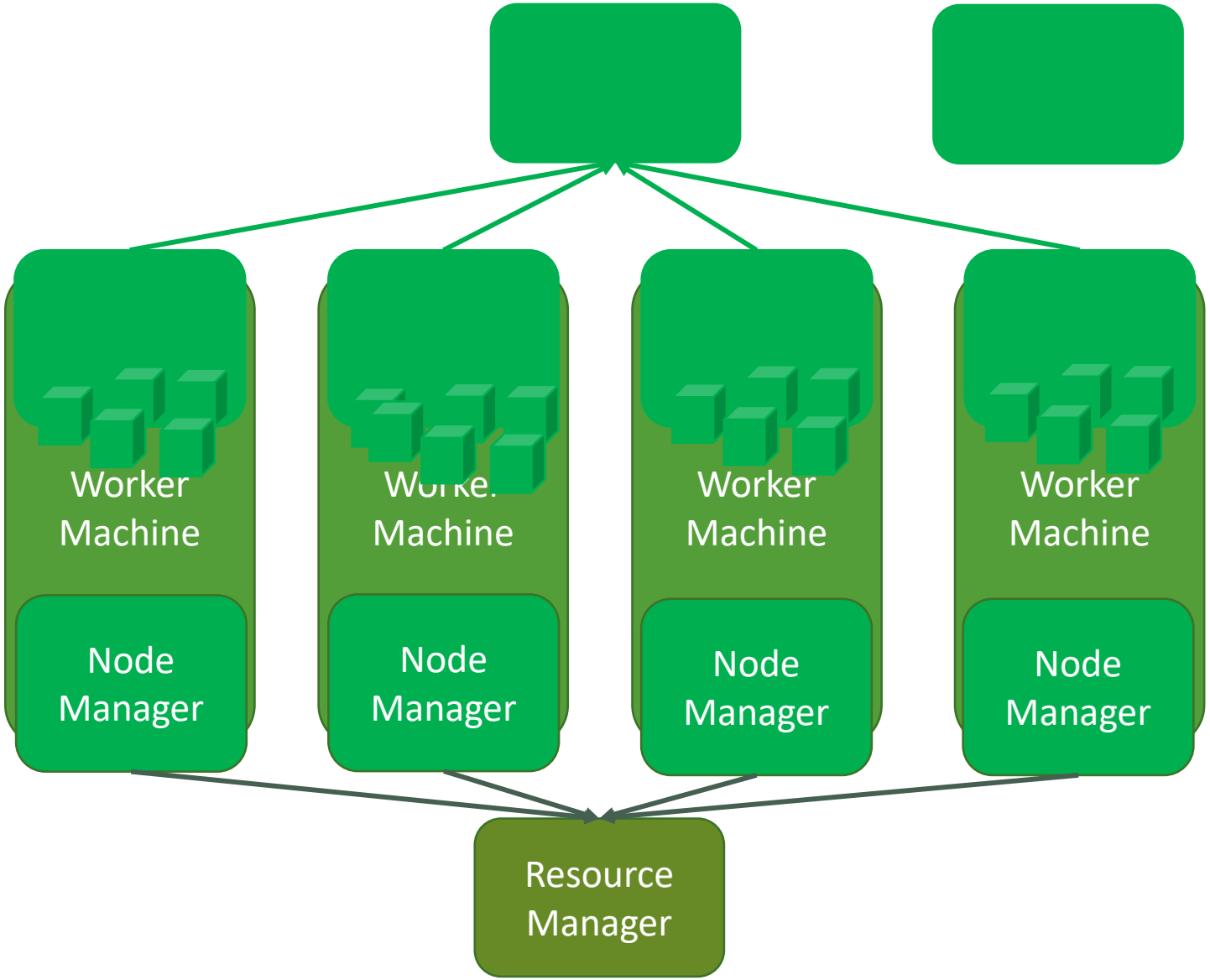

Batch Layer
All Data (HDFS)
Pre-Compute Views
(Map Reduce)
HIVE
AVRO
Partial Aggregate
Batch View
Serving Layer
Real Time View
Real Time Data
Merged View
(Hbase)
HBASE
FLUME
Speed Layer
Stream
Processing
Increment
View
Spark


Ingestion
(Volume)
Streaming Data / Velocity
Batch
Store
Store
Model
Prediction / Prescription
Dashboard
Query
Data
Mining/
Analysis
Massive Compute Capacity
Refine/
ML
Massive Storage
Big Data
proc
mem
mem
HD
Dynamic Scaling (Cloud)

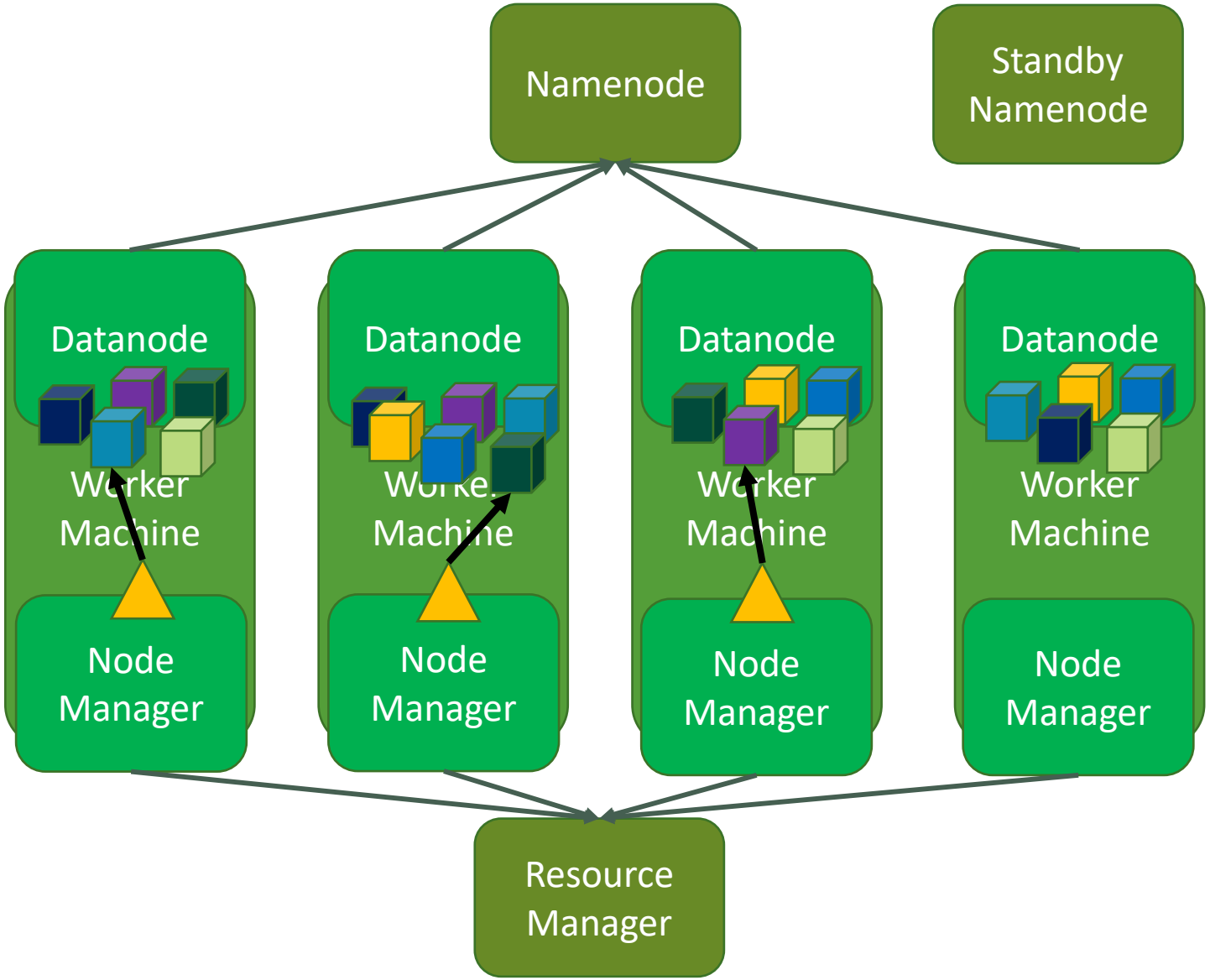
HDFS


Namenode
Standby Namenode
Datanode
Worker Machine
Datanode
Worker Machine
Datanode
Worker Machine
Datanode
Worker Machine

YARN


Worker Machine
Worker Machine
Worker Machine
Node Manager
Node Manager
Node Manager
Resource Manager

HDFS and YARN



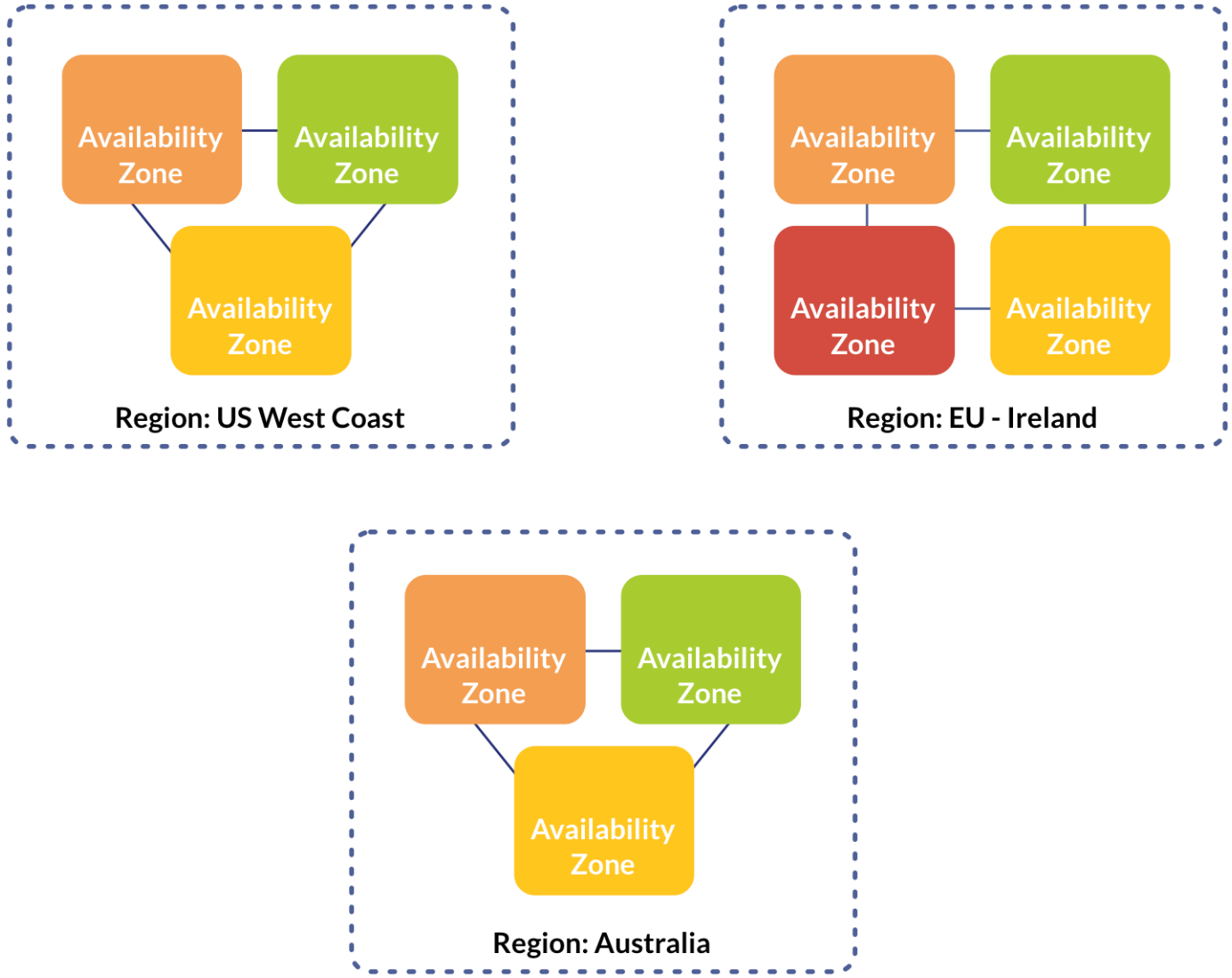
Hadoop EcoSystem							
Data Visualization							
	SAS Visual Analytics	Tableau	Qlick	SAP Lumira	R		
	D3.js	iCharts	Timeline JS	Apache Zeppelin	Pentaho		
System Deployment							
	Apache Ambari	Apache Mesos	Marathon	Apache Bigtop	HortonWorks HOYA	Deploop	
	Apache Eagle	Cloudera Manager	Brooklyn	Apache Helix	Buildoop	SequenceIQ Cloudbreak	
	Myriad						
Data Ingestion	**Service Programming**		**Scheduling & DR**	**Security**	**Frameworks**	**Metadata**	**Machine Learning**
Apache Flume	Apache Thrift	Apache Karaf	Apache Oozie	Apache Sentry	Jumbune	Metascope	Apache Mahout
Apache Sqoop	Apache Zookeeper	Twitter Elephant Bi	Linkedin Azkaban	Apache Knox Gatew	Spring XD	Apache Tika	WEKA
Facebook Scribe	Apache Avro	Linkedin Norbert	Apache Falcon	Apache Ranger	Cask Data App Platform		Cloudera Oryx
Apache Chukwa	Apache Curator		Shedoscope				Deeplearning4j
Apache Kafka							MADlib
Netflix Suro	**Distributed Programming**				**SQL on Hadoop**		H2O
Apache Samza	Apache Ignite	Apache Flink	Apache Twill	Kangaroo	Apache Hive	Facebook Presto	Sparkling Water
Cloudera Morphlin	Apache Mapreduce	Apache Apex	Damballa PARKOUR	TinkerPop	Apache Hcatalog	Datasalt Splout SQL	Apache SystemML
HIHO	Apache Pig	Netflix Pigpen	Apache Hama	Pachyderm Mapred	Apache Trafodion	Apache Tajo	
Apache Nifi	JAQL	AMPLAB SIMR	Datasalt Pangool	Apache Beam	Apache HAWQ	Apache MRQL	
Apache ManifoldCF	Apache Spark	Facebook Corona	Apache Tez		Apache Drill	Kylin	
	Apache Storm	Apache REEF	Apache DataFu		Cloudera Impala		
	NoSQL Databases					**NewSQL Databases**	**Deep Learning**
	Wide Column	**Document**	**Key-Value**	**Graph**	**Stream Data Model**	TokuDB	Caffe
	Apache Hbase	MongoDB	Redis	Giraph	EventStore	HandlerSocket	Microsoft CNTK
	Apache Cassandra	RethinkDB	Linkedin Voldemort	Neo4J		Akiban Server	Tensor Flow
	Hypertable	ArangoDB	RocksDB	TitanDB		Drizzle	Theano
	Apache Accumulo	CouchDB	OpenTSDB	OrientDB		Haeinsa	Torch
	Apache Kudu	DynamoDB				SenseiDB	Mxnet
	Apache Parquet	Gemfire				Sky	Chainer
						BayesDB	Keras
						InfluxDB	
						VoldDB	
						SAP HANA	
Distributed File System							
	Apache HDFS	Quantcast File Syste	Lustre File System	GridGain			
	Red Hat GlusterFS	Ceph File System	Alluxio	XtreemFS			

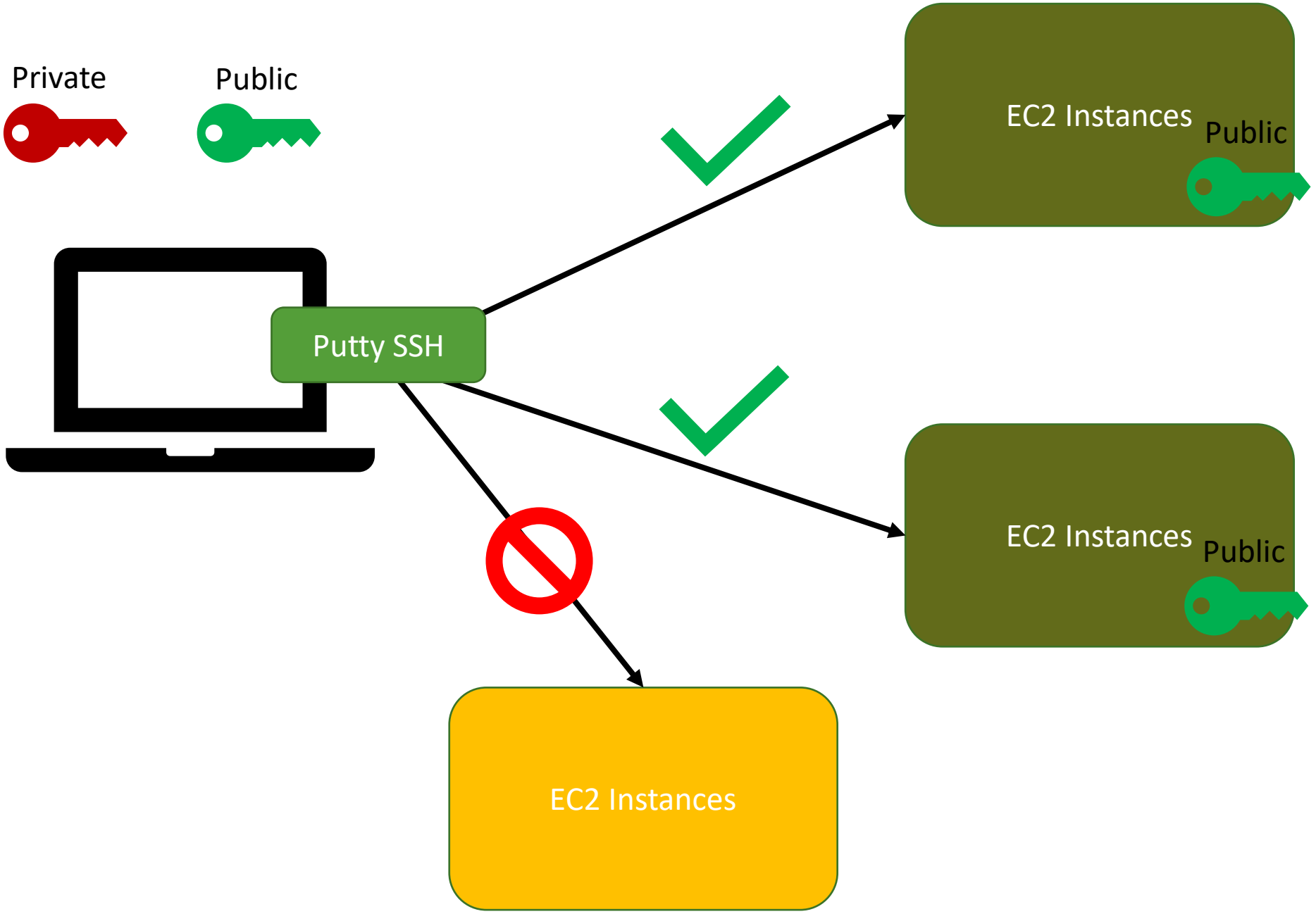
AWS Regions and AZs

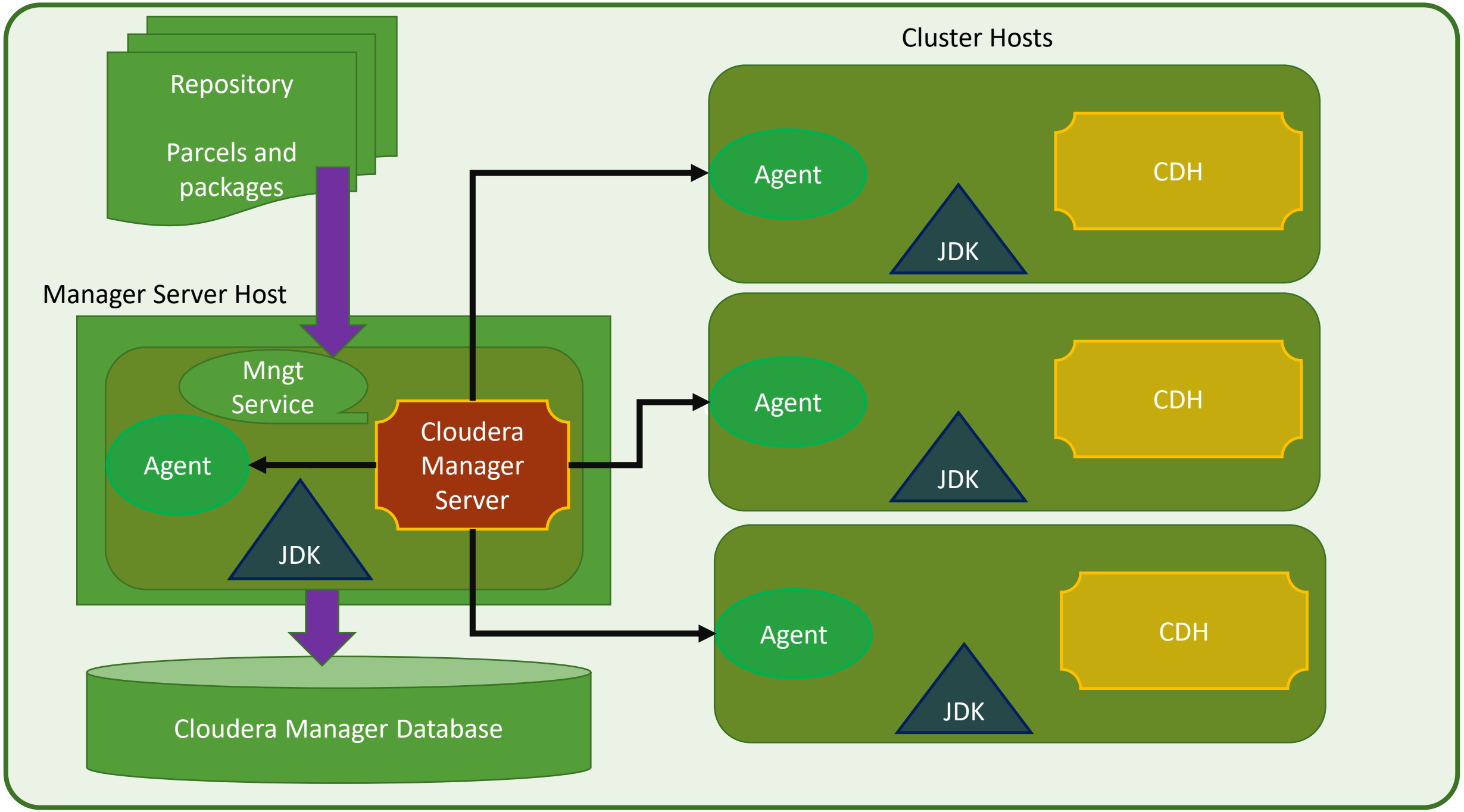
15 Regions

40 Availability Zones

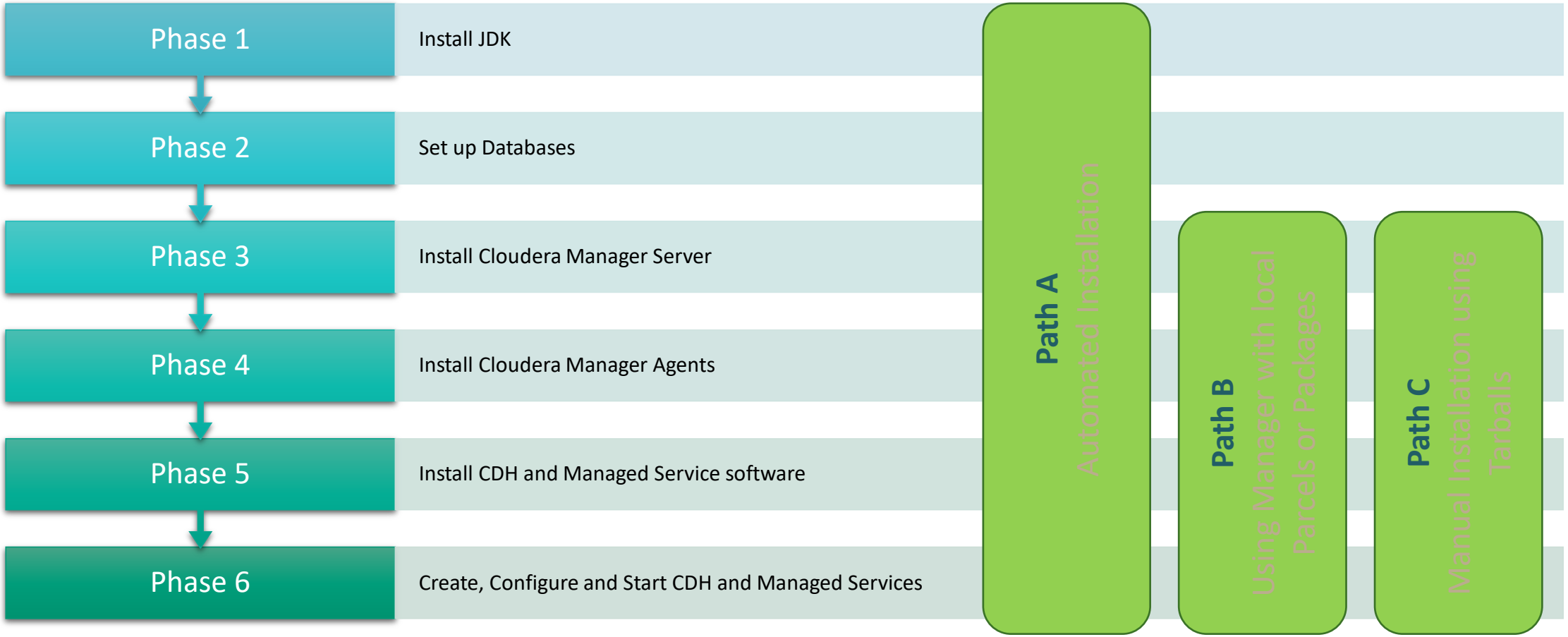
68 Edge Locations


Availability Zone
Availability Zone
Availability Zone
Region: US West Coast
Availability Zone
Availability Zone
Availability Zone
Availability Zone
Region: EU - Ireland
Availability Zone
Availability Zone
Availability Zone
Region: Australia


Private
Public
Putty SSH
EC2 Instances
Public
EC2 Instances
Public
EC2 Instances


Repository
Parcels and packages
Cluster Hosts
Agent
CDH
JDK
Manager Server Host
Mngt Service
Agent
Cloudera Manager Server
JDK
Agent
CDH
JDK
Agent
CDH
JDK
Cloudera Manager Database

Cloudera Installation Phases and Paths

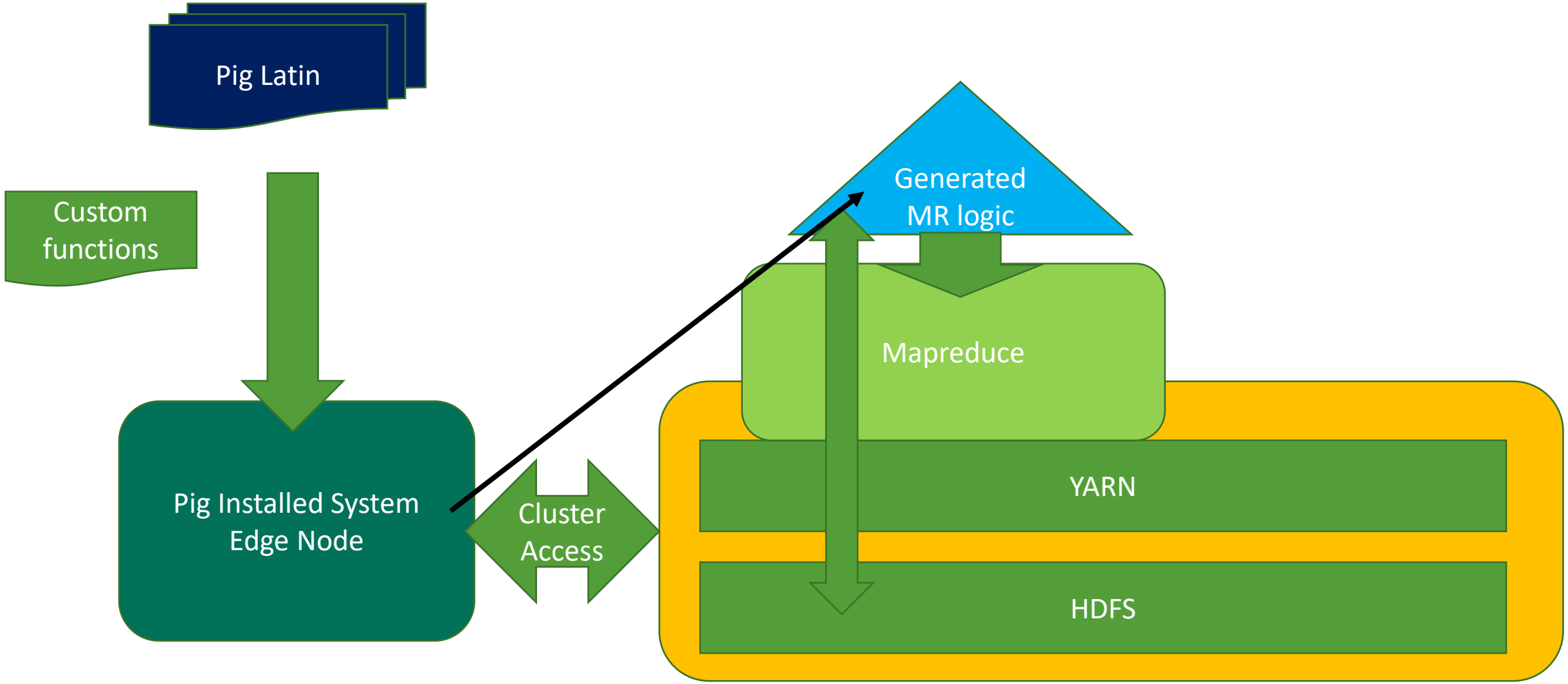


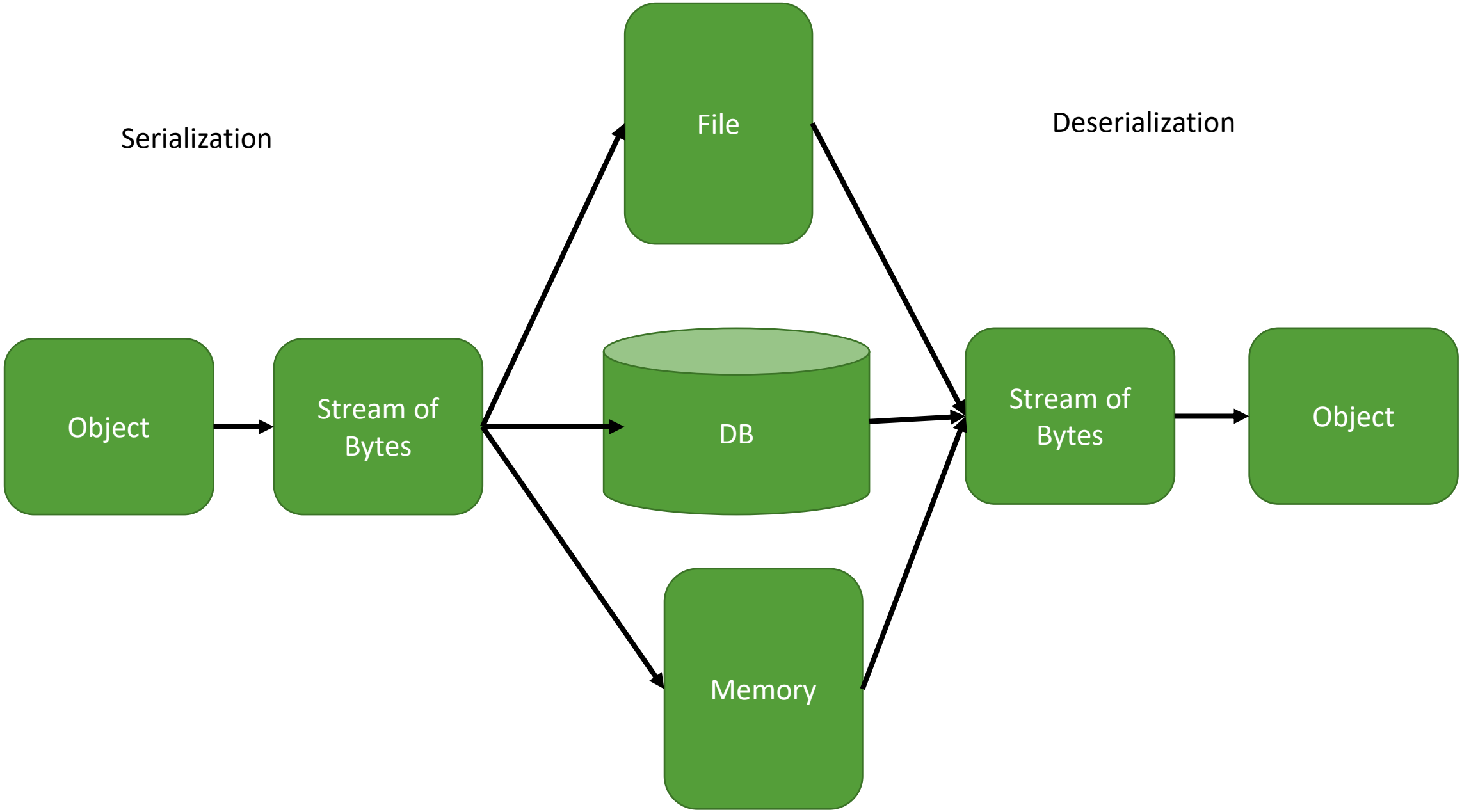
Advantages of using Parcels

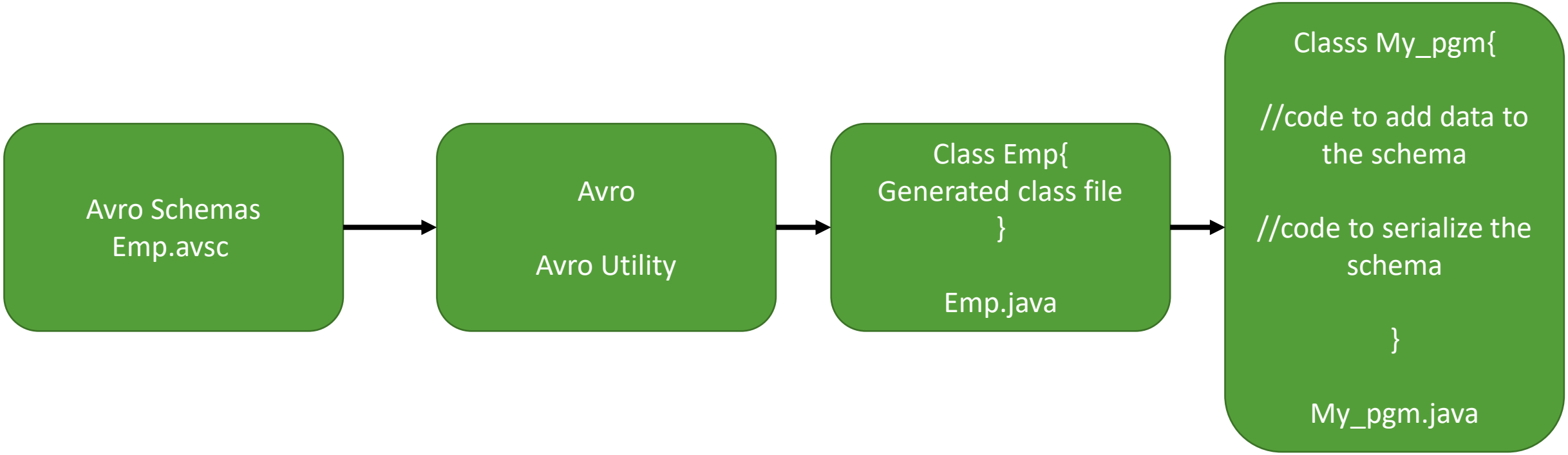
- Distribution of CDH as a single object - Instead of having a separate package for each part of CDH, **parcels have just a single object to install**
- Internal consistency - All CDH components are matched, **eliminating the possibility of installing parts from different versions of CDH**
- Installation **outside of /usr**
- Installation of CDH without sudo - Parcel installation is handled by the Cloudera Manager Agent running as root or another user, so **you can install CDH without sudo.**

Types of Hosts and their roles

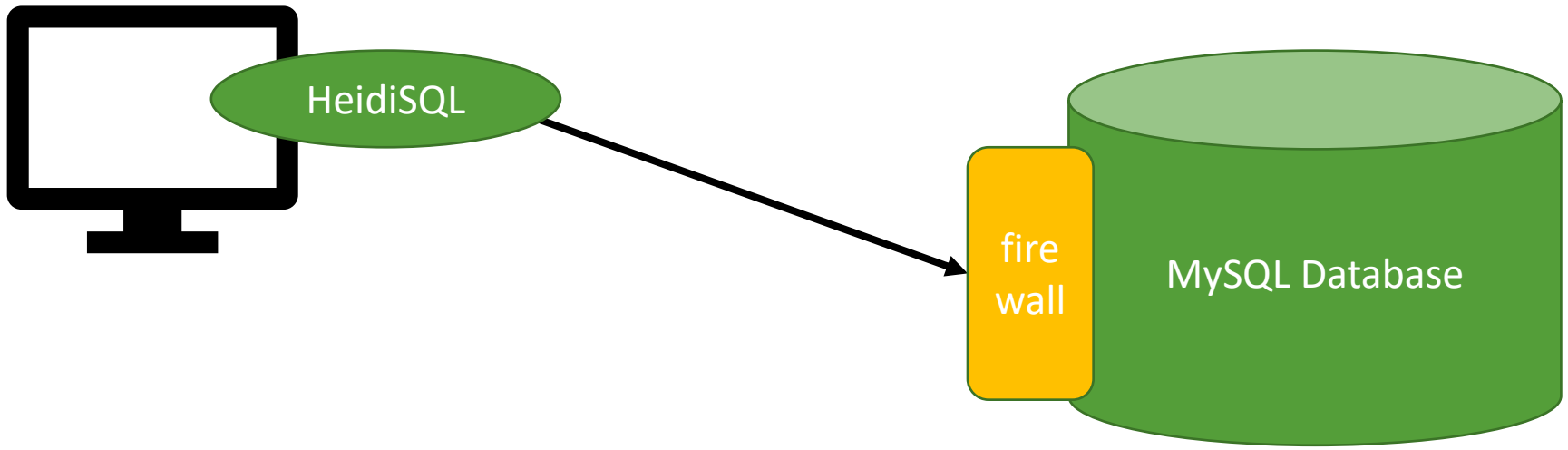
- **Master hosts** run Hadoop master processes such as the HDFS NameNode and YARN Resource Manager.
- **Worker hosts** primarily run DataNodes and other distributed processes such as Impalad, Datanode, Node Manager.
- **Utility hosts** run other cluster processes that are not master processes such as Cloudera Manager and the Hive Metastore.
- **Edge hosts** are client access points for launching jobs in the cluster. The number of Edge hosts required varies depending on the type and size of the workloads.


Pig Latin
Custom functions
Pig Installed System Edge Node
Cluster Access
Generated MR logic
Mapreduce
YARN
HDFS

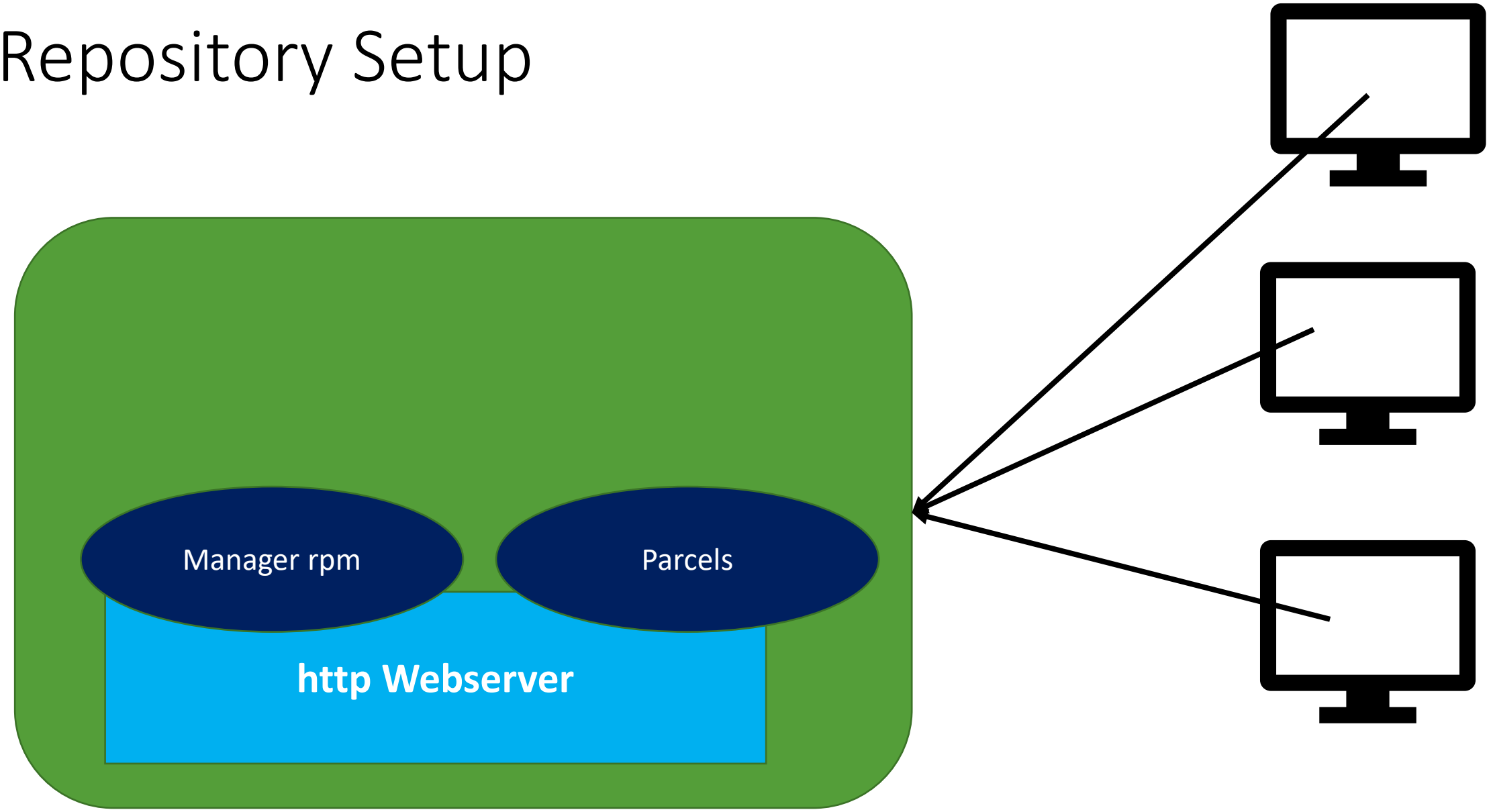

Serialization
Deserialization
File
Object
Stream of Bytes
DB
Stream of Bytes
Object
Memory


Avro Schemas
Emp.avsc
Avro
Avro Utility
Class Emp{
Generated class file
}
Emp.java
Classs My_pgm{
//code to add data to the schema
//code to serialize the schema
}
My_pgm.java

MySQL Database Installation and Overview



Repository Setup

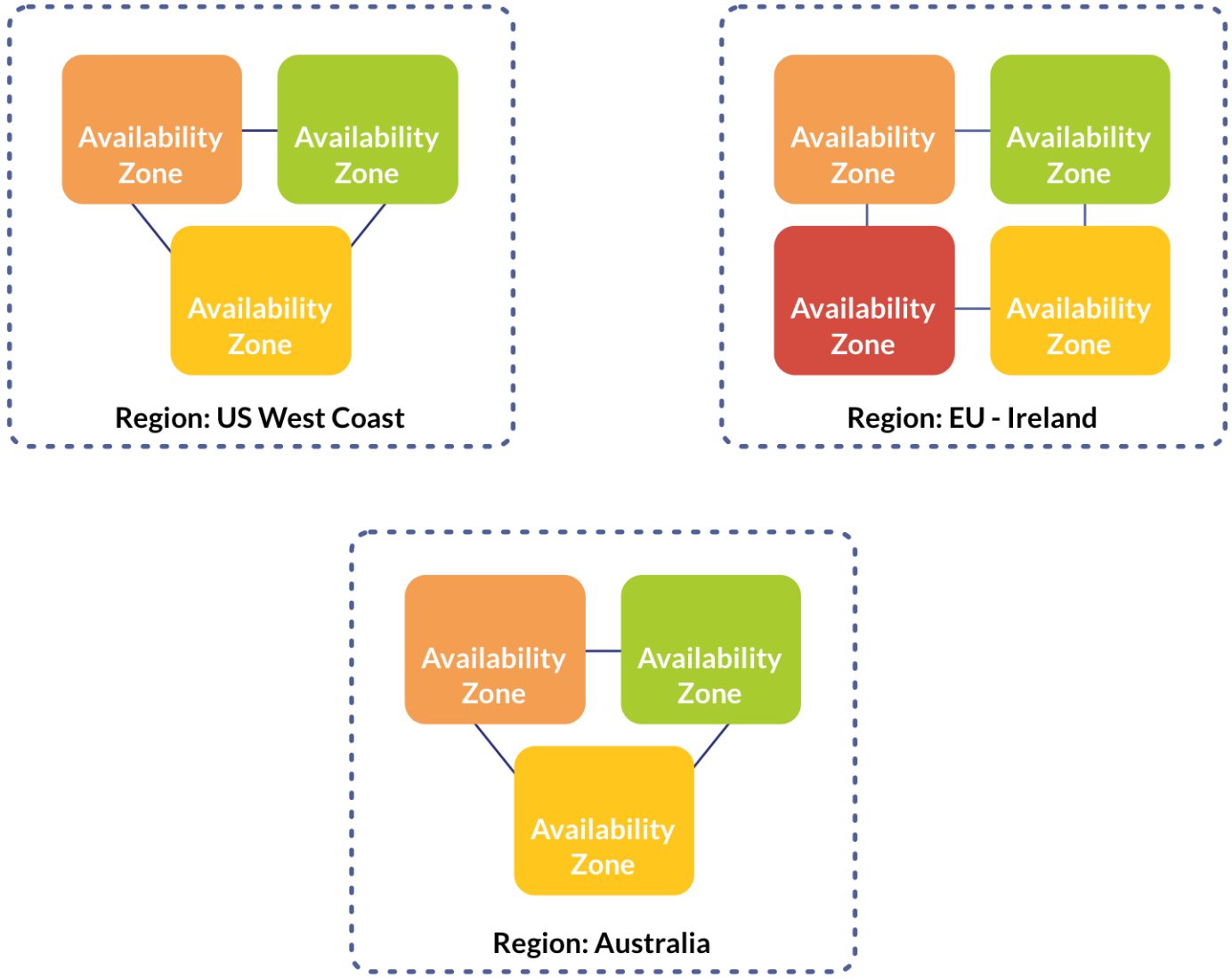


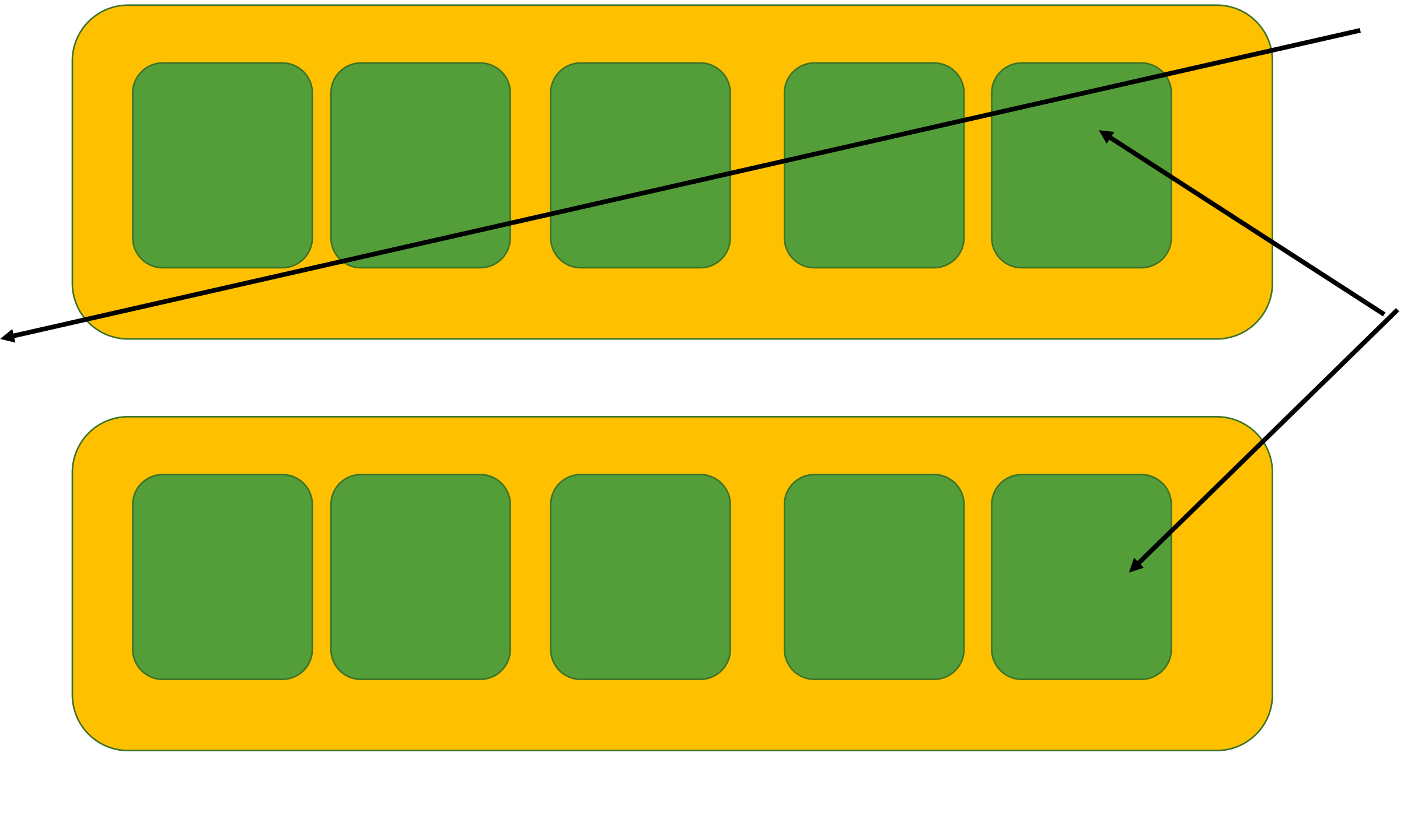
AWS Regions and AZs

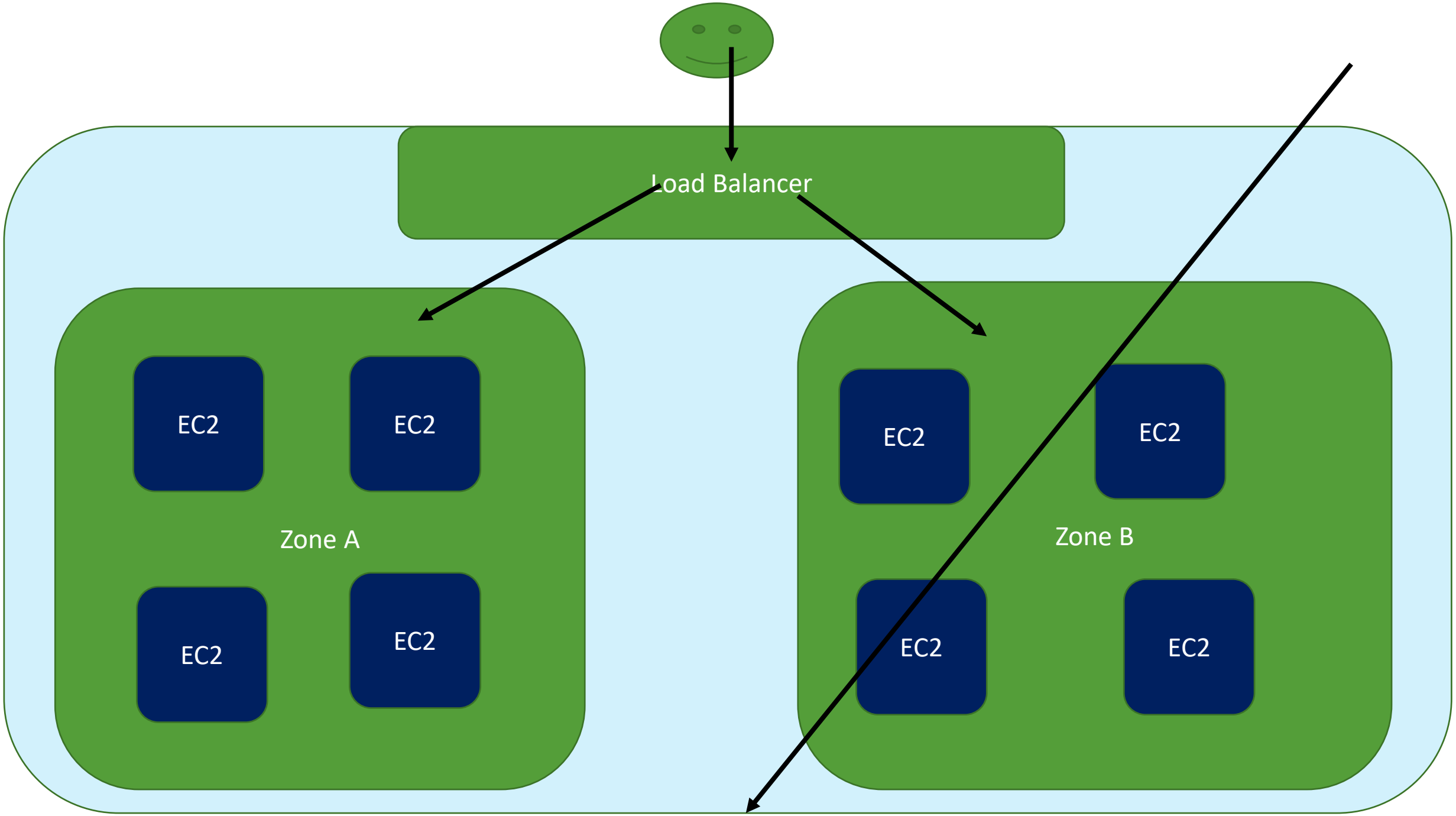
15 Regions

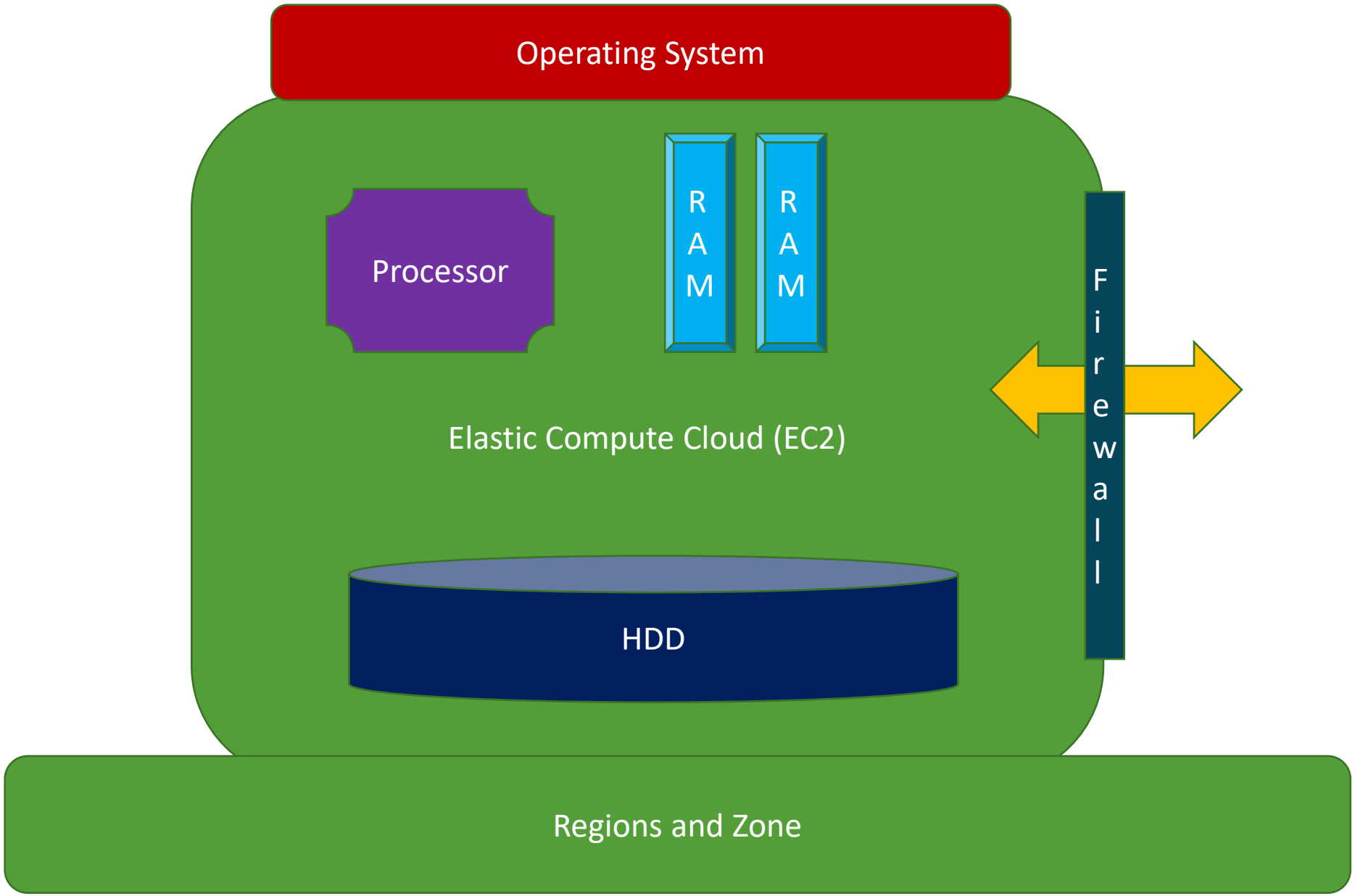
40 Availability Zones

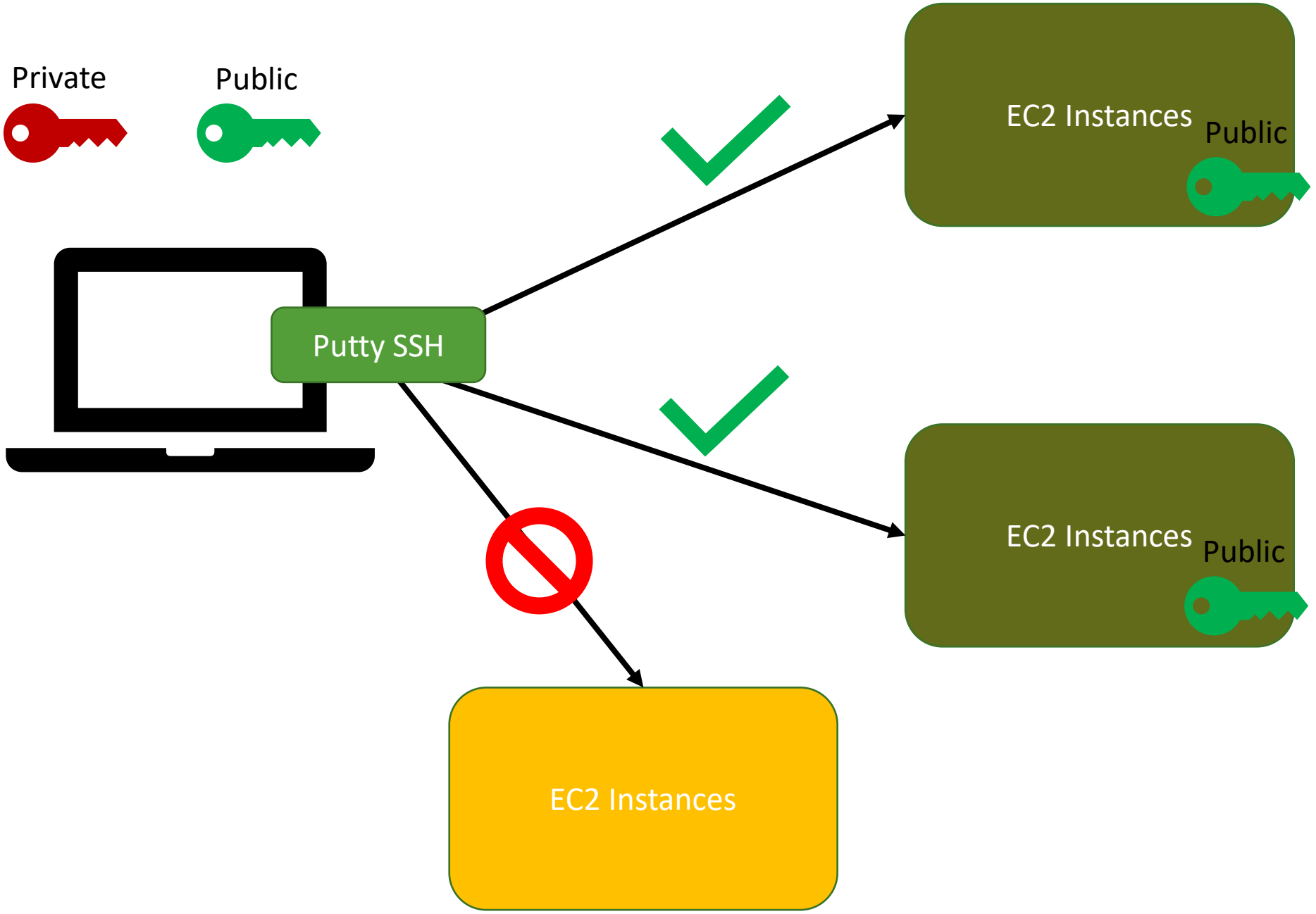
68 Edge Locations

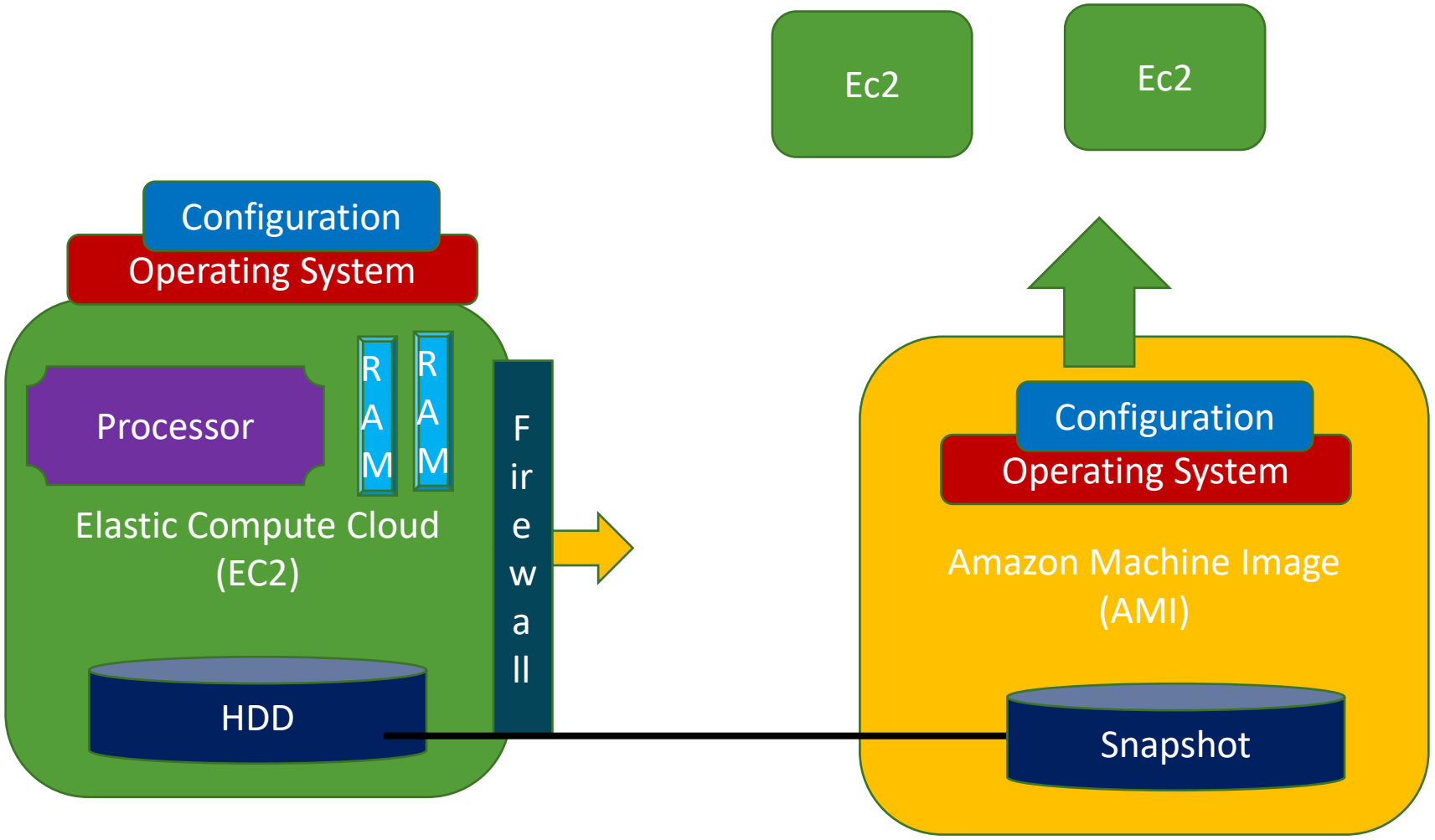

Availability Zone
Availability Zone
Availability Zone
Region: US West Coast
Availability Zone
Availability Zone
Availability Zone
Availability Zone
Region: EU - Ireland
Availability Zone
Availability Zone
Availability Zone
Region: Australia

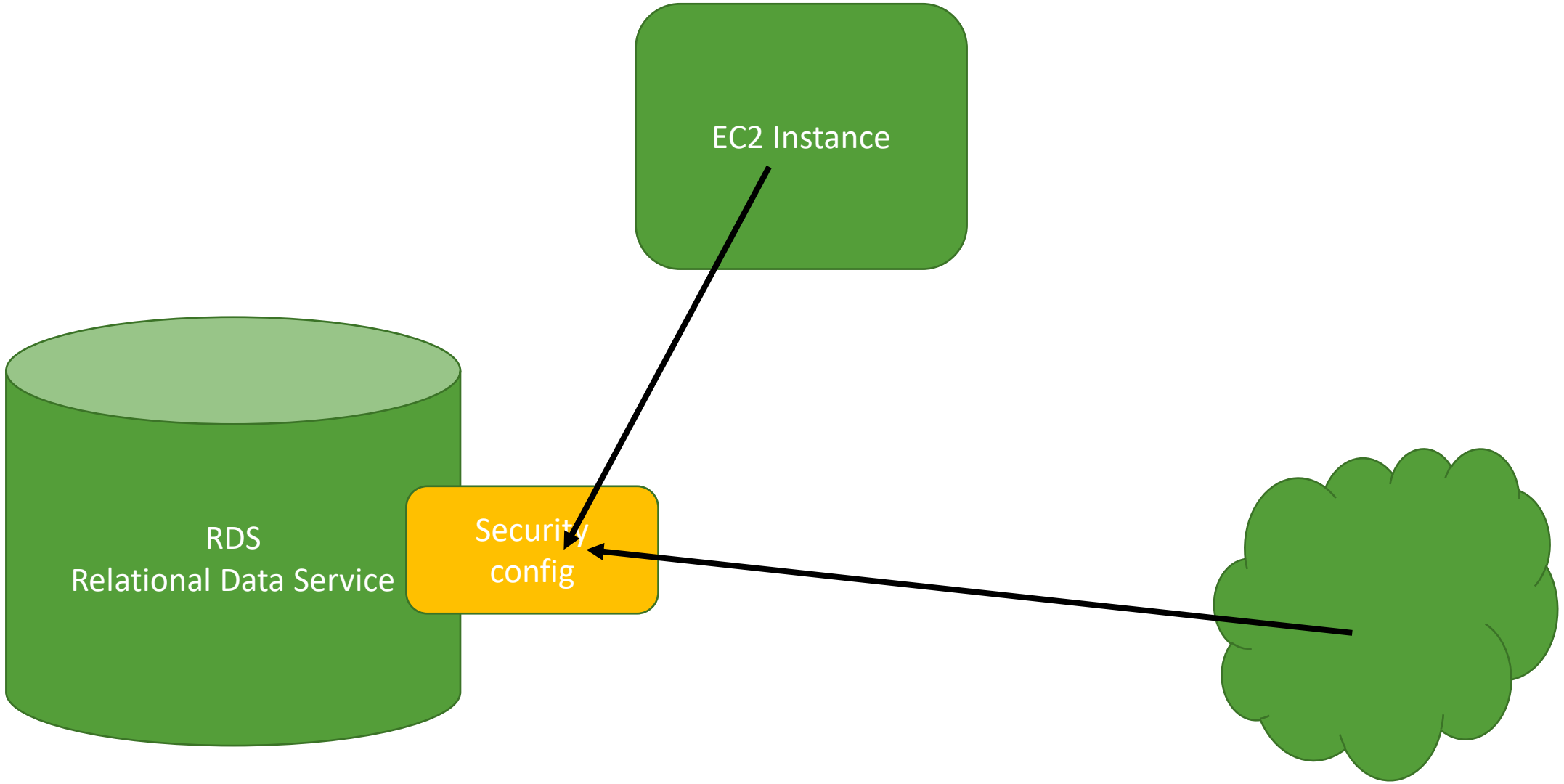


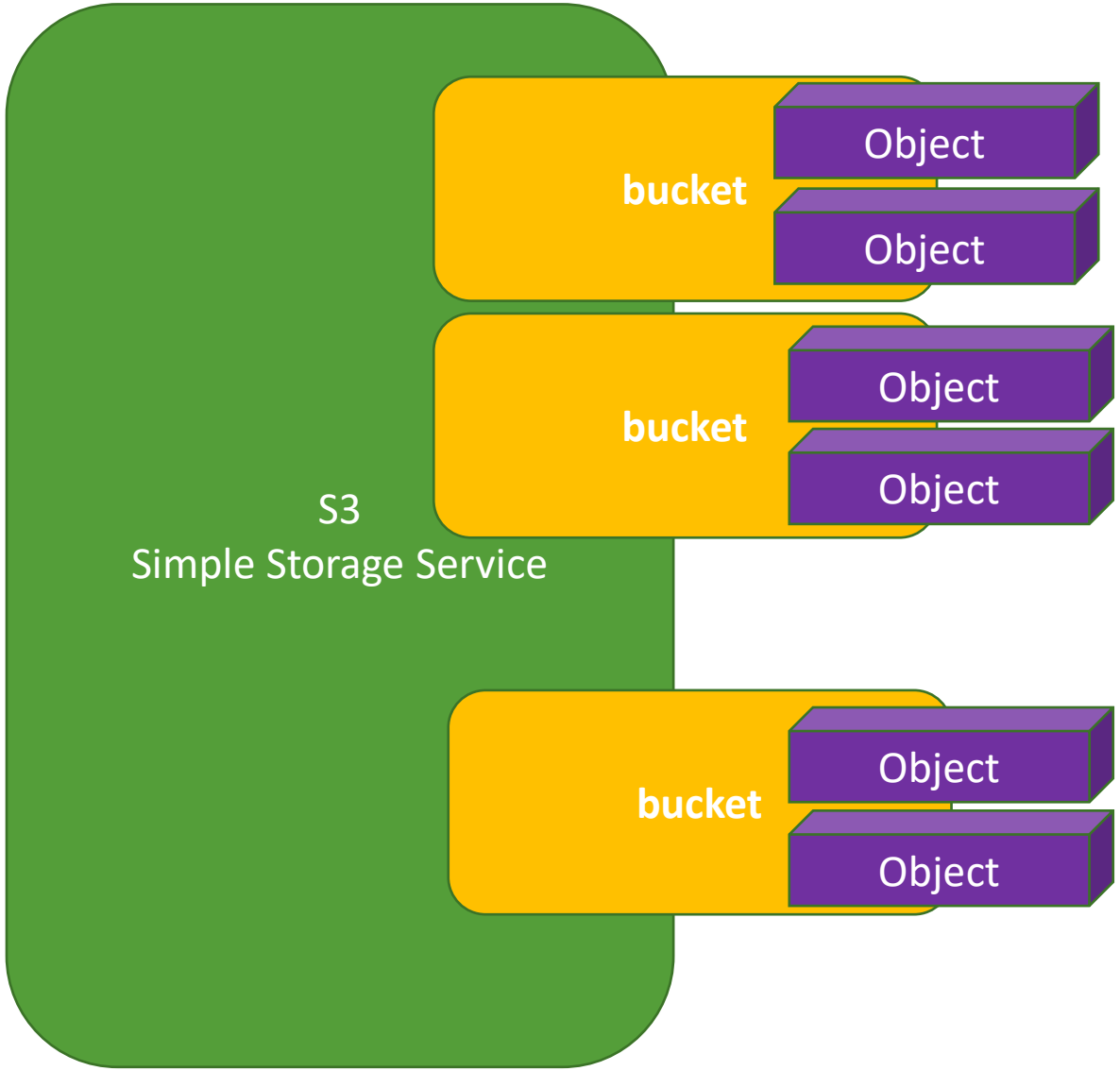

Load Balancer
EC2
EC2
Zone A
EC2
EC2
EC2
EC2
Zone B
EC2
EC2

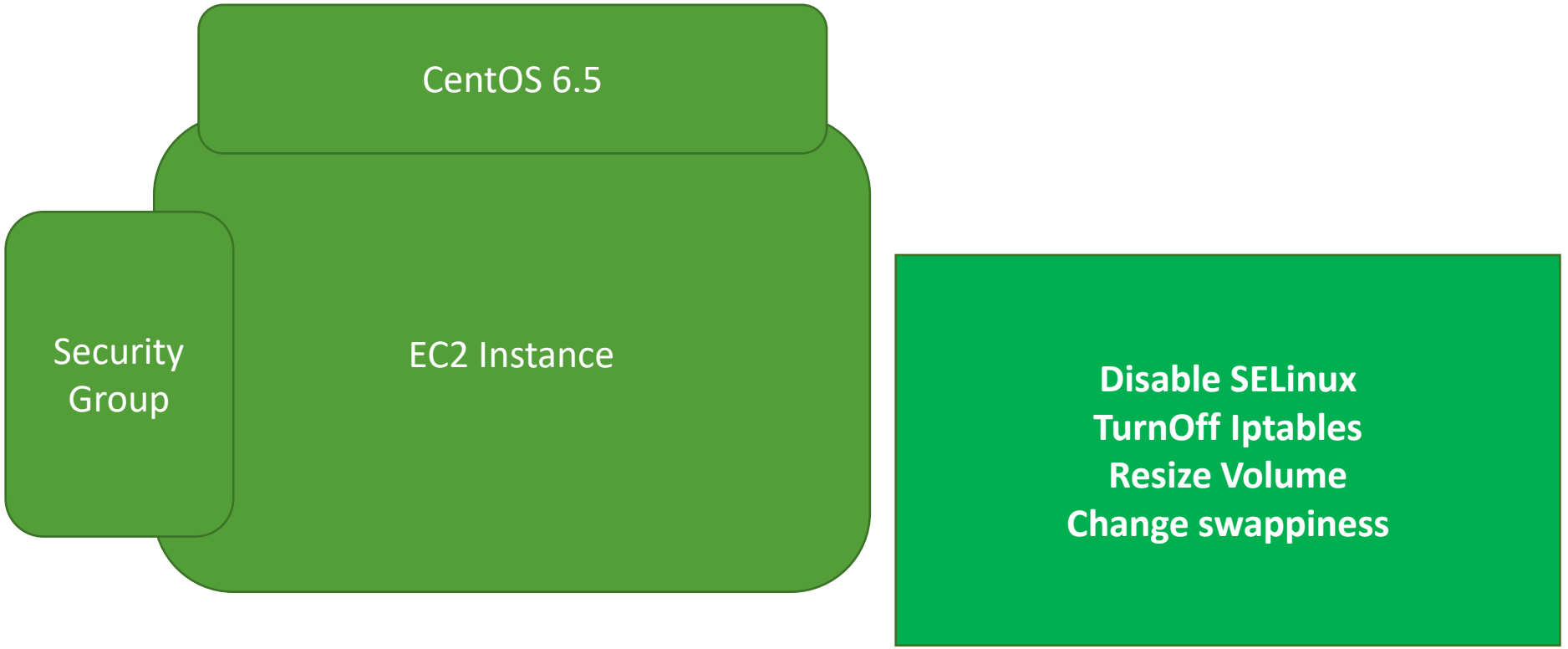

Operating System
Processor
RAM
RAM
Firewall
Elastic Compute Cloud (EC2)
HDD
Regions and Zone

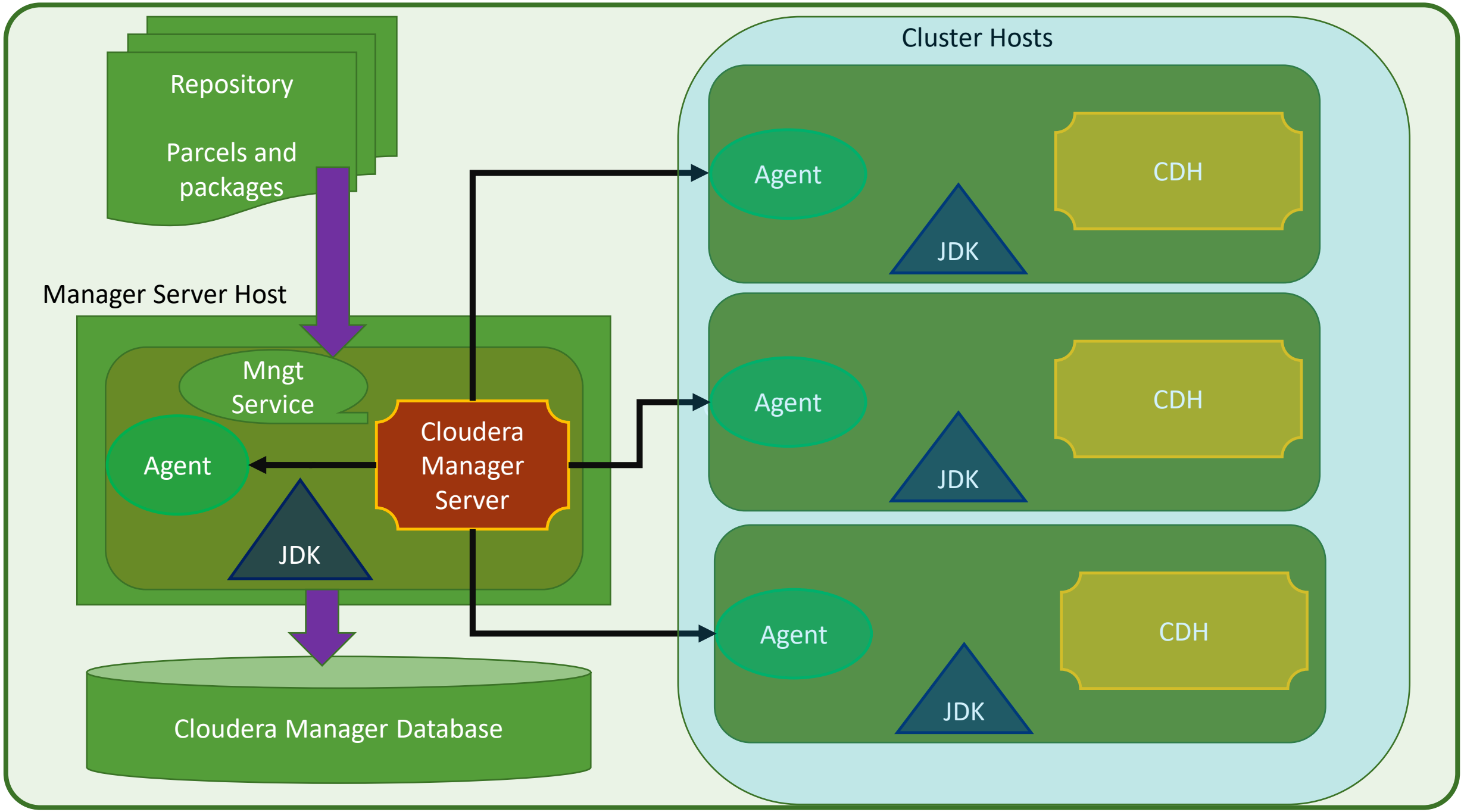

Private
Public
Putty SSH
EC2 Instances
Public
EC2 Instances
Public
EC2 Instances


Ec2
Ec2
Configuration
Operating System
Processor
RAM
RAM
Elastic Compute Cloud
(EC2)
HDD
Firewall
Configuration
Operating System
Amazon Machine Image
(AMI)
Snapshot

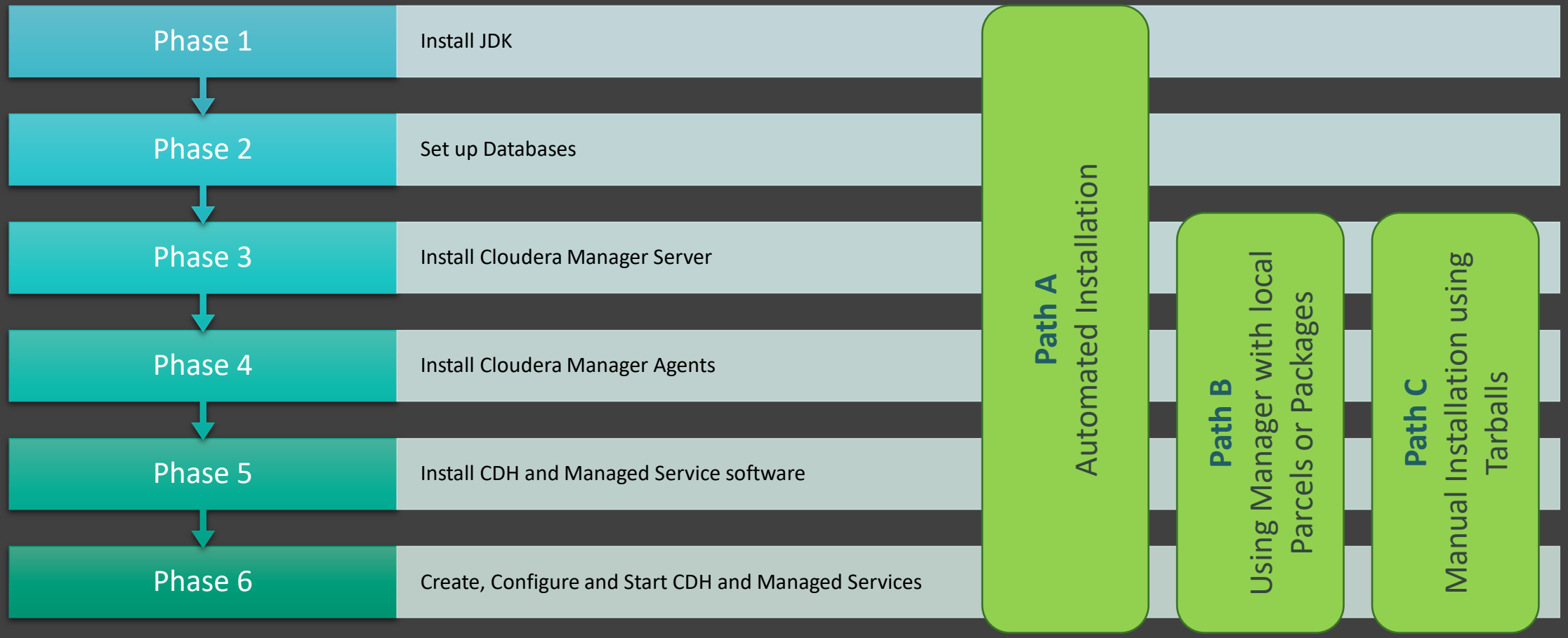

EC2 Instance
RDS
Relational Data Service
Security
config


bucket
Object
Object
bucket
Object
Object
S3
Simple Storage Service
bucket
Object
Object


CentOS 6.5
Security Group
EC2 Instance
Disable SELinux
TurnOff Iptables
Resize Volume
Change swappiness


Repository
Parcels and packages
Cluster Hosts
Agent
CDH
JDK
Manager Server Host
Mngt Service
Agent
Cloudera Manager Server
JDK
Agent
CDH
JDK
Agent
CDH
JDK
Cloudera Manager Database

Cloudera Installation Phases and Paths

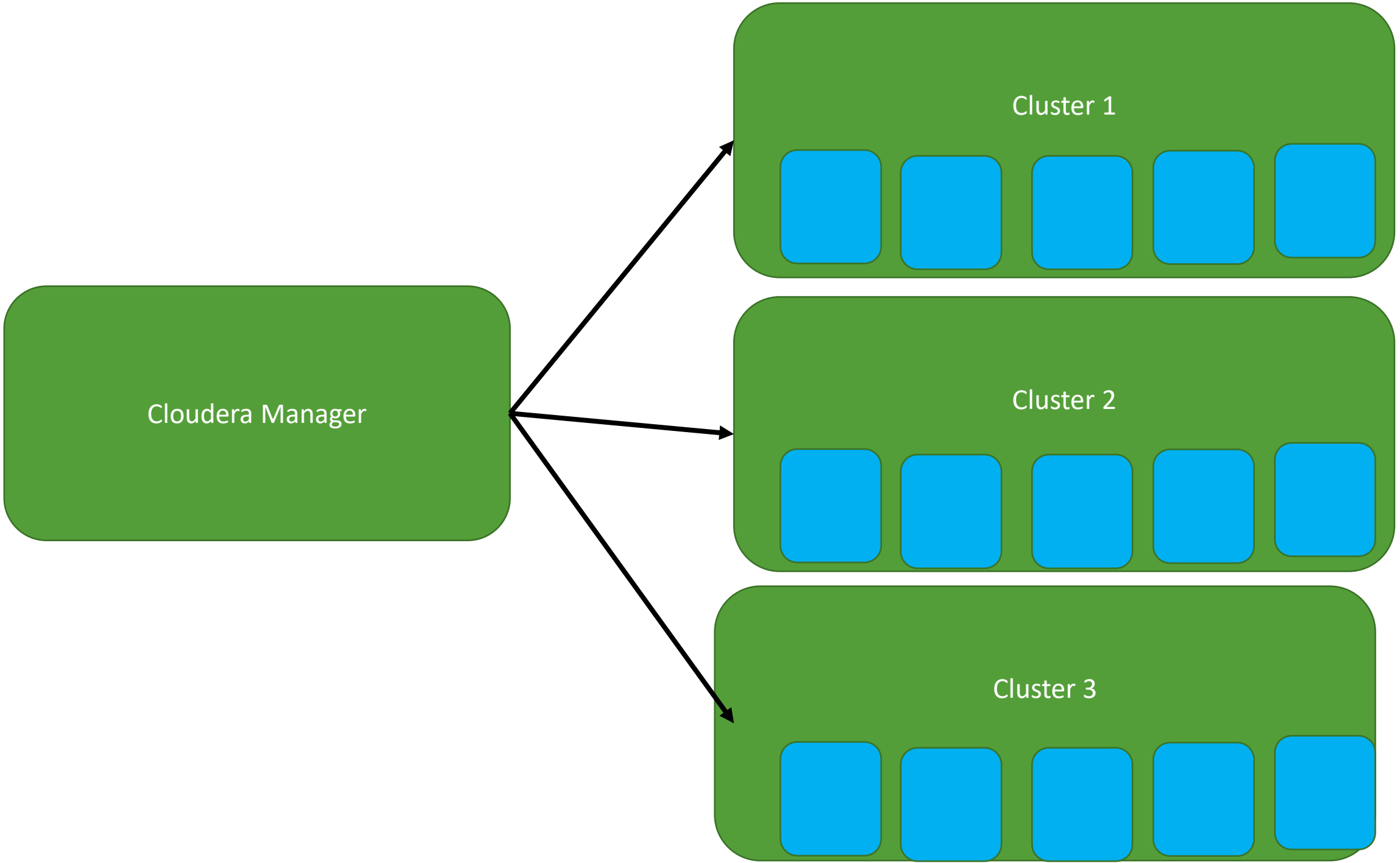


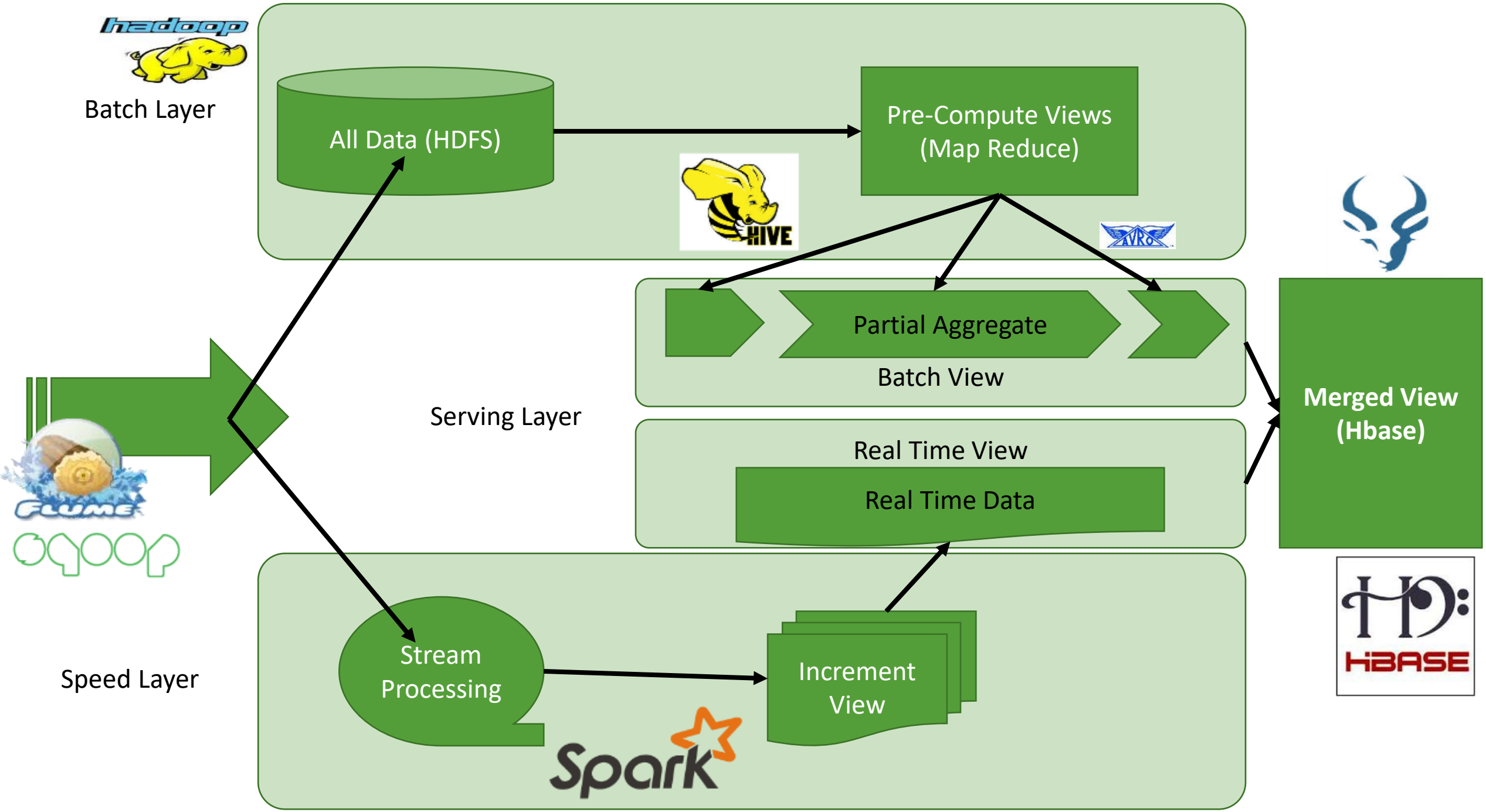
Advantages of using Parcels

- Distribution of CDH as a single object - Instead of having a separate package for each part of CDH, **parcels have just a single object to install**
- Internal consistency - All CDH components are matched, **eliminating the possibility of installing parts from different versions of CDH**
- Installation **outside of /usr**
- Installation of CDH without sudo - Parcel installation is handled by the Cloudera Manager Agent running as root or another user, so **you can install CDH without sudo.**

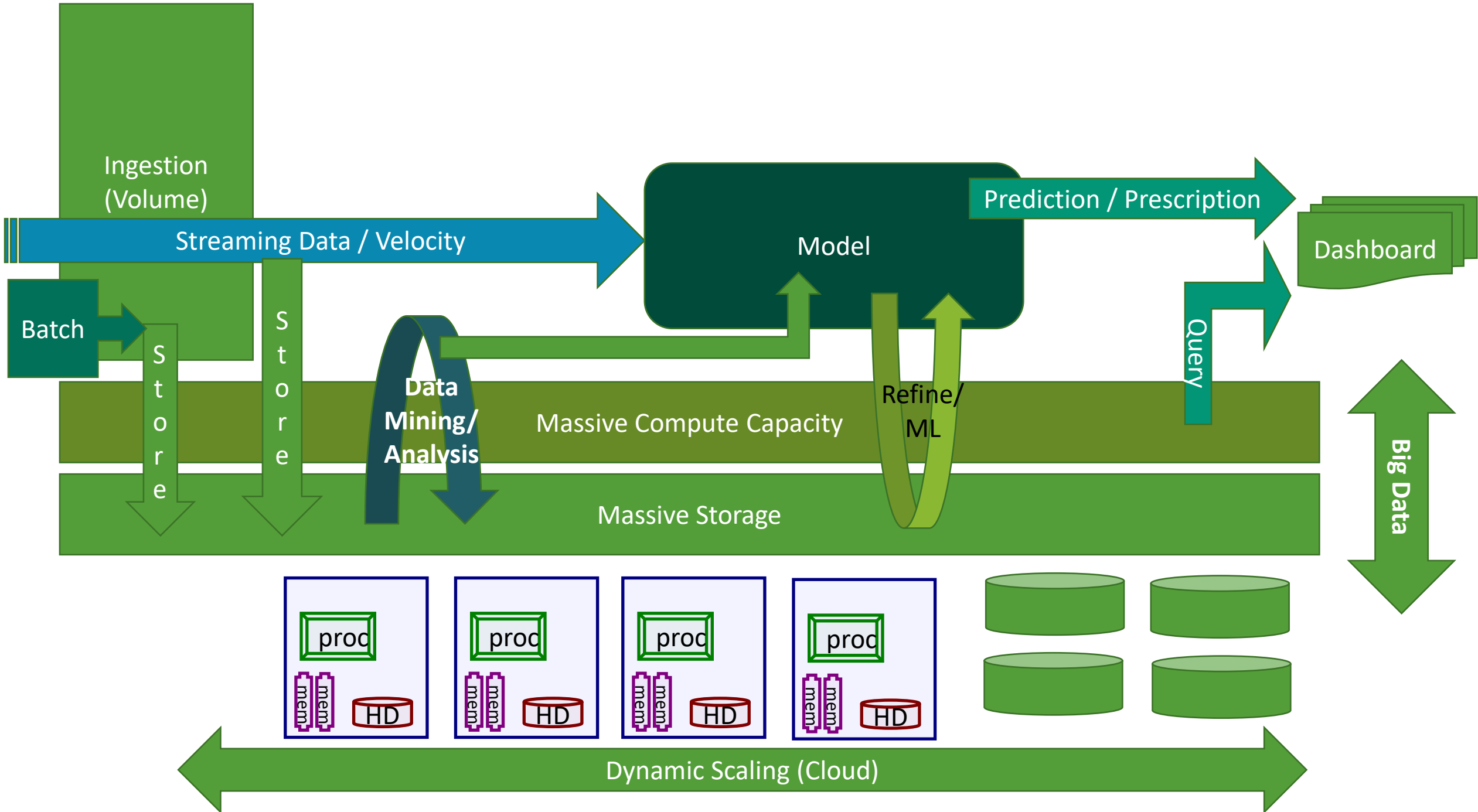
Types of Hosts and their roles

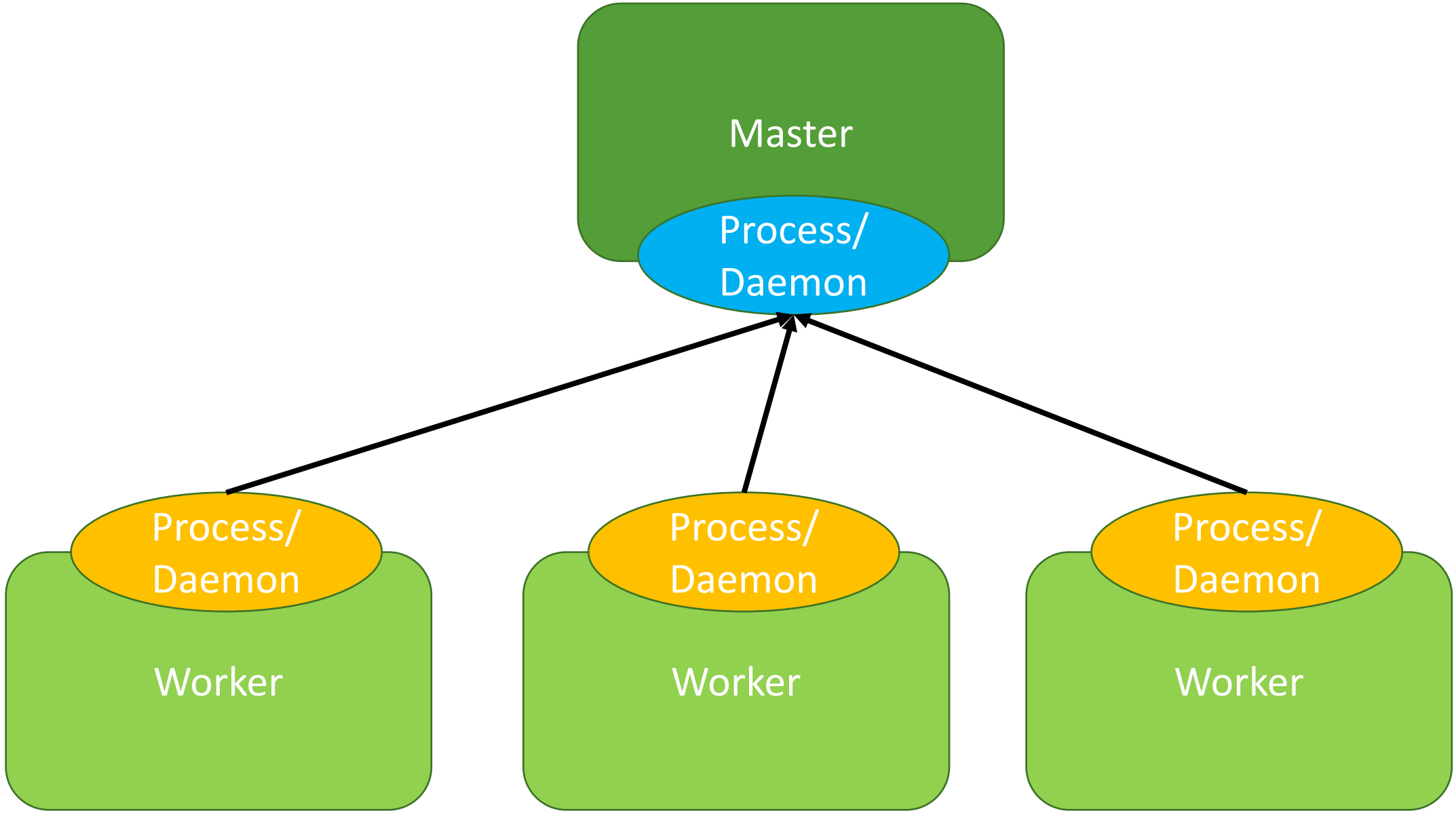
- **Master hosts** run Hadoop master processes such as the HDFS NameNode and YARN Resource Manager.
- **Worker hosts** primarily run DataNodes and other distributed processes such as Impalad, Datanode, Node Manager.
- **Utility hosts** run other cluster processes that are not master processes such as Cloudera Manager and the Hive Metastore.
- **Edge hosts** are client access points for launching jobs in the cluster. The number of Edge hosts required varies depending on the type and size of the workloads.

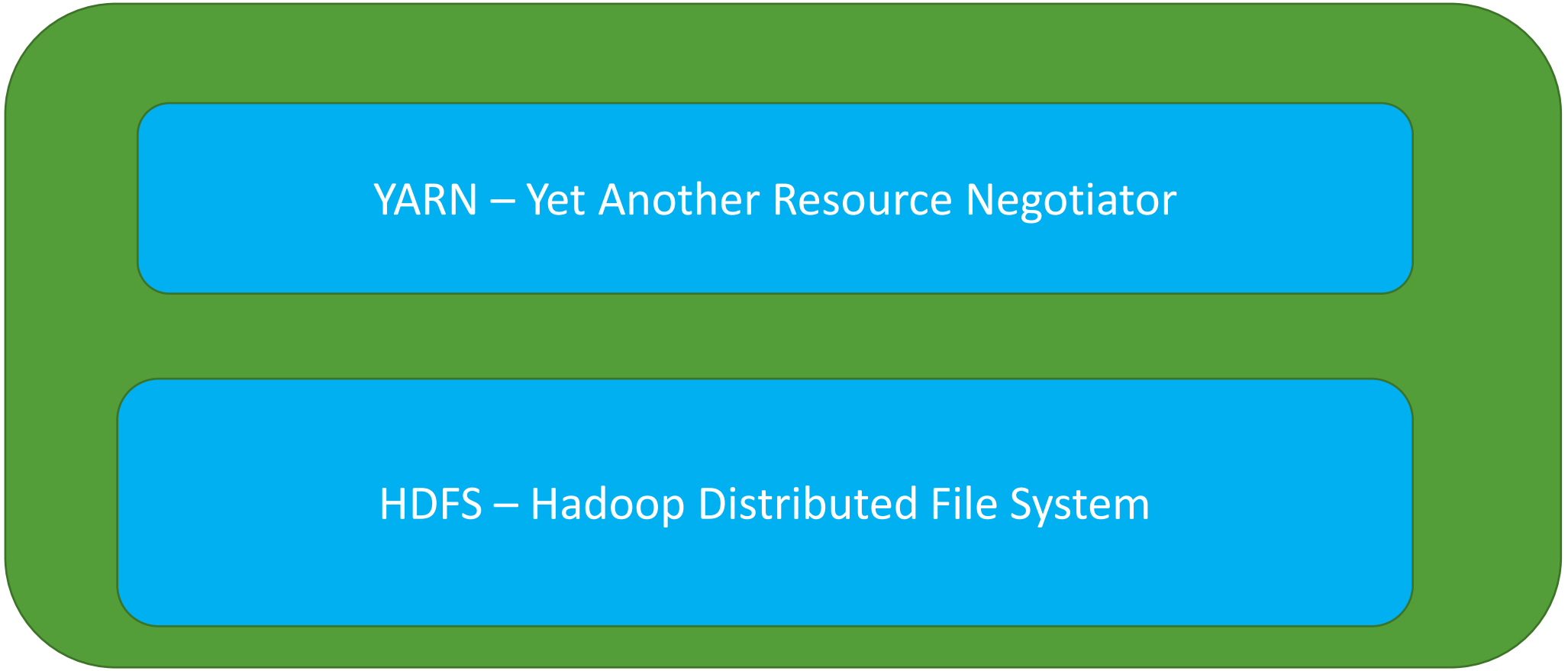

Cloudera Manager
Cluster 1
Cluster 2
Cluster 3


Batch Layer
All Data (HDFS)
Pre-Compute Views
(Map Reduce)
HIVE
AVRO
Partial Aggregate
Batch View
Serving Layer
Real Time View
Real Time Data
Merged View
(Hbase)
HBASE
FLUME
Speed Layer
Stream
Processing
Increment
View
Spark

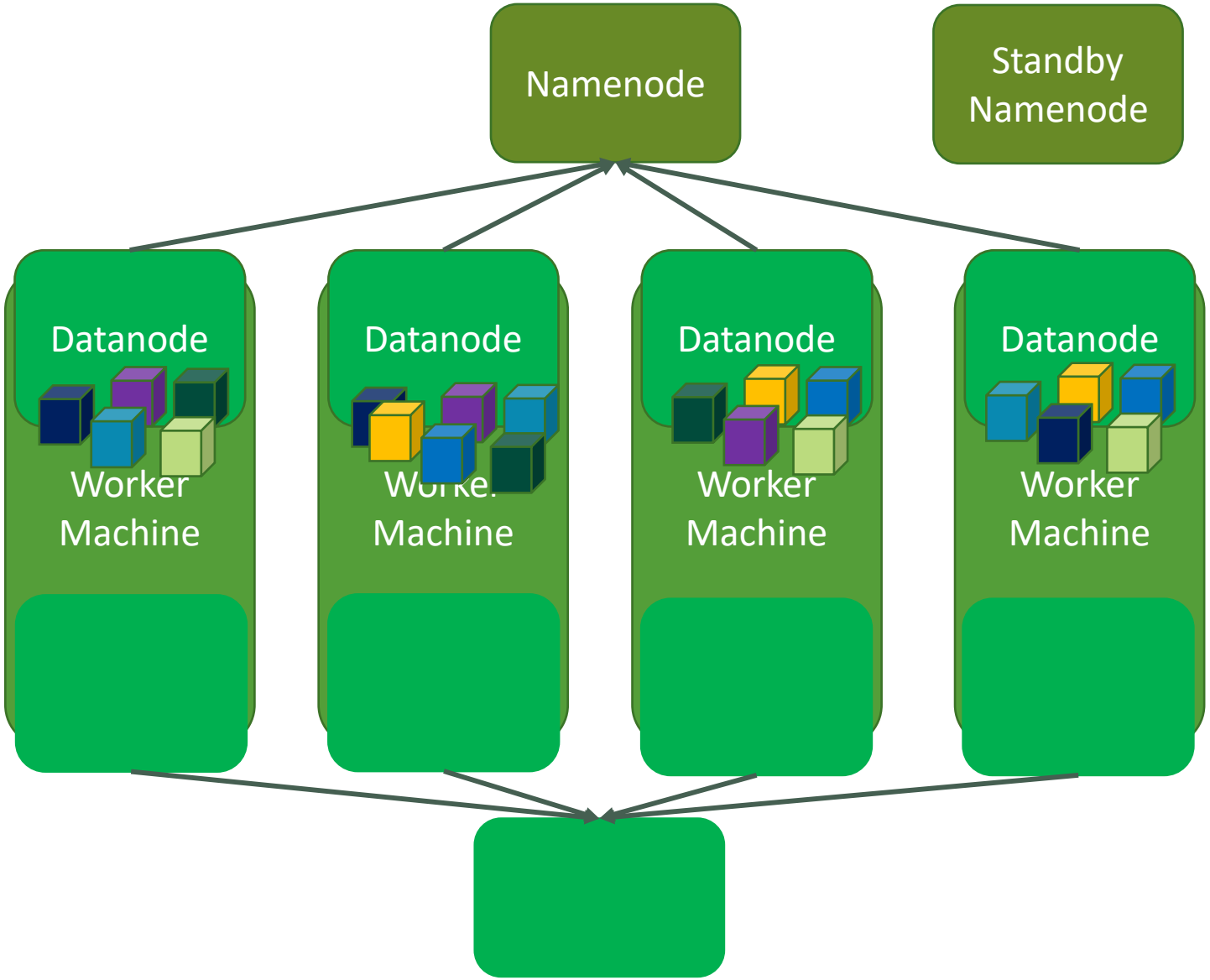
Hadoop EcoSystem							
Data Visualization							
	SAS Visual Analytics	Tableau	Qlick	SAP Lumira	R		
	D3.js	iCharts	Timeline JS	Apache Zeppelin	Pentaho		
System Deployment							
	Apache Ambari	Apache Mesos	Marathon	Apache Bigtop	HortonWorks HOYA	Deploop	
	Apache Eagle	Cloudera Manager	Brooklyn	Apache Helix	Buildoop	SequenceIQ Cloudbreak	
	Myriad						
Data Ingestion	**Service Programming**		**Scheduling & DR**	**Security**	**Frameworks**	**Metadata**	**Machine Learning**
Apache Flume	Apache Thrift	Apache Karaf	Apache Oozie	Apache Sentry	Jumbune	Metascope	Apache Mahout
Apache Sqoop	Apache Zookeeper	Twitter Elephant Bi	Linkedin Azkaban	Apache Knox Gatew	Spring XD	Apache Tika	WEKA
Facebook Scribe	Apache Avro	Linkedin Norbert	Apache Falcon	Apache Ranger	Cask Data App Platform		Cloudera Oryx
Apache Chukwa	Apache Curator		Shedoscope				Deeplearning4j
Apache Kafka							MADlib
Netflix Suro	**Distributed Programming**				**SQL on Hadoop**		H2O
Apache Samza	Apache Ignite	Apache Flink	Apache Twill	Kangaroo	Apache Hive	Facebook Presto	Sparkling Water
Cloudera Morphlin	Apache Mapreduce	Apache Apex	Damballa PARKOUR	TinkerPop	Apache Hcatalog	Datasalt Splout SQL	Apache SystemML
HIHO	Apache Pig	Netflix Pigpen	Apache Hama	Pachyderm Mapred	Apache Trafodion	Apache Tajo	
Apache Nifi	JAQL	AMPLAB SIMR	Datasalt Pangool	Apache Beam	Apache HAWQ	Apache MRQL	
Apache ManifoldCF	Apache Spark	Facebook Corona	Apache Tez		Apache Drill	Kylin	
	Apache Storm	Apache REEF	Apache DataFu		Cloudera Impala		
	NoSQL Databases					**NewSQL Databases**	**Deep Learning**
	Wide Column	**Document**	**Key-Value**	**Graph**	**Stream Data Model**	TokuDB	Caffe
	Apache Hbase	MongoDB	Redis	Giraph	EventStore	HandlerSocket	Microsoft CNTK
	Apache Cassandra	RethinkDB	Linkedin Voldemort	Neo4J		Akiban Server	Tensor Flow
	Hypertable	ArangoDB	RocksDB	TitanDB		Drizzle	Theano
	Apache Accumulo	CouchDB	OpenTSDB	OrientDB		Haeinsa	Torch
	Apache Kudu	DynamoDB				SenseiDB	Mxnet
	Apache Parquet	Gemfire				Sky	Chainer
						BayesDB	Keras
						InfluxDB	
						VoldDB	
						SAP HANA	
Distributed File System							
	Apache HDFS	Quantcast File Syste	Lustre File System	GridGain			
	Red Hat GlusterFS	Ceph File System	Alluxio	XtreemFS			


Ingestion
(Volume)
Streaming Data / Velocity
Batch
Store
Store
Model
Prediction / Prescription
Dashboard
Query
Data
Mining/
Analysis
Massive Compute Capacity
Refine/
ML
Massive Storage
Big Data
proc
mem
mem
HD
Dynamic Scaling (Cloud)


Master
Process/
Daemon
Process/
Daemon
Worker
Process/
Daemon
Worker
Process/
Daemon
Worker


YARN – Yet Another Resource Negotiator
HDFS – Hadoop Distributed File System

HDFS

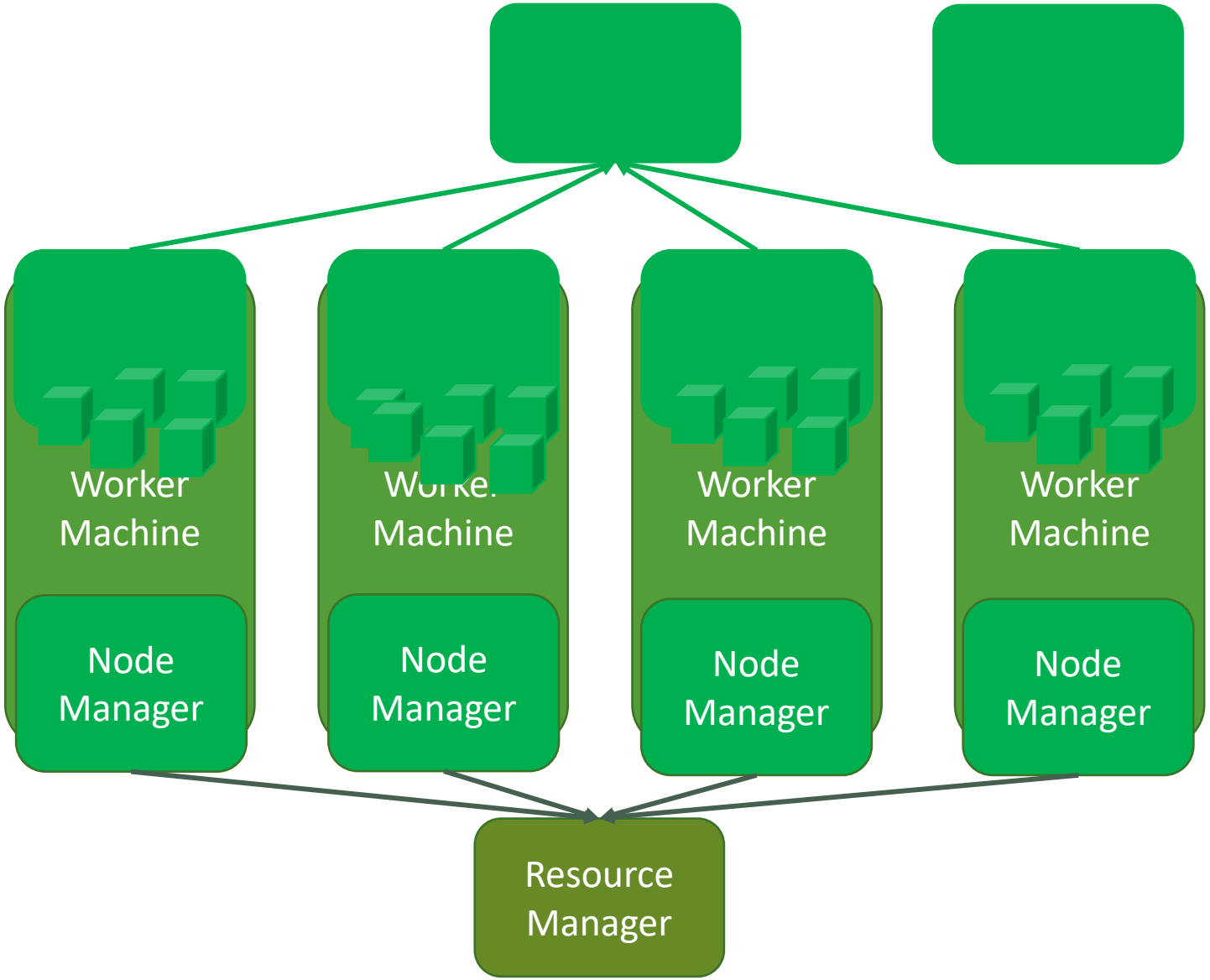


200 MB

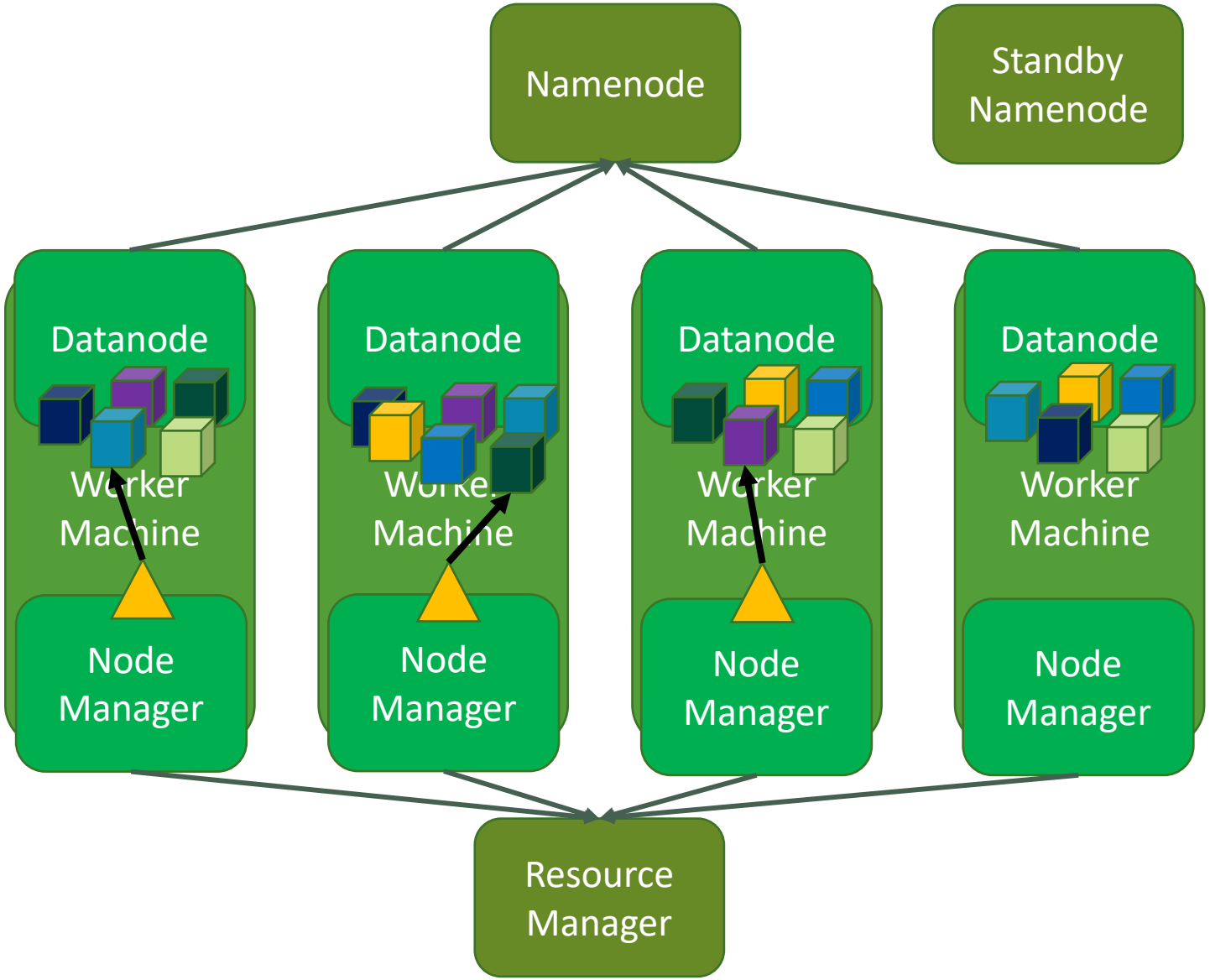
128 MB

72 MB

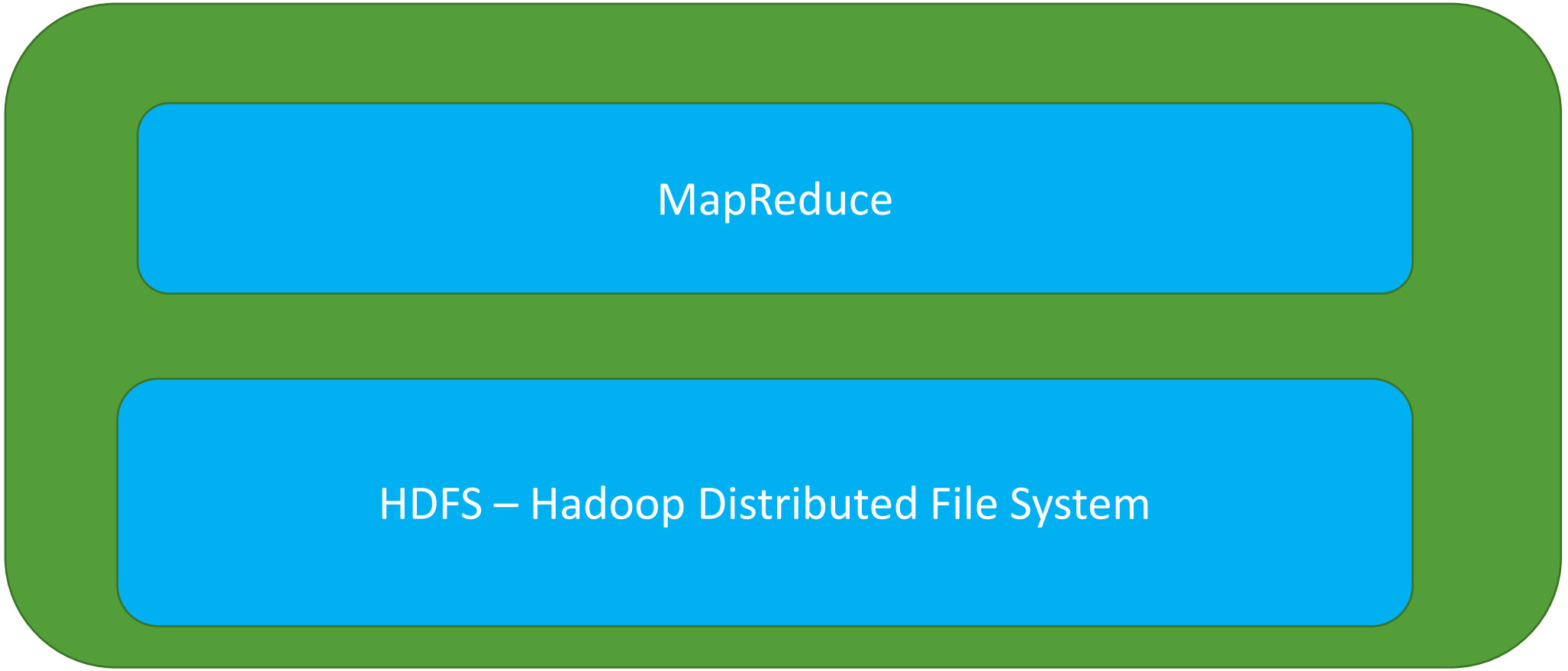
YARN

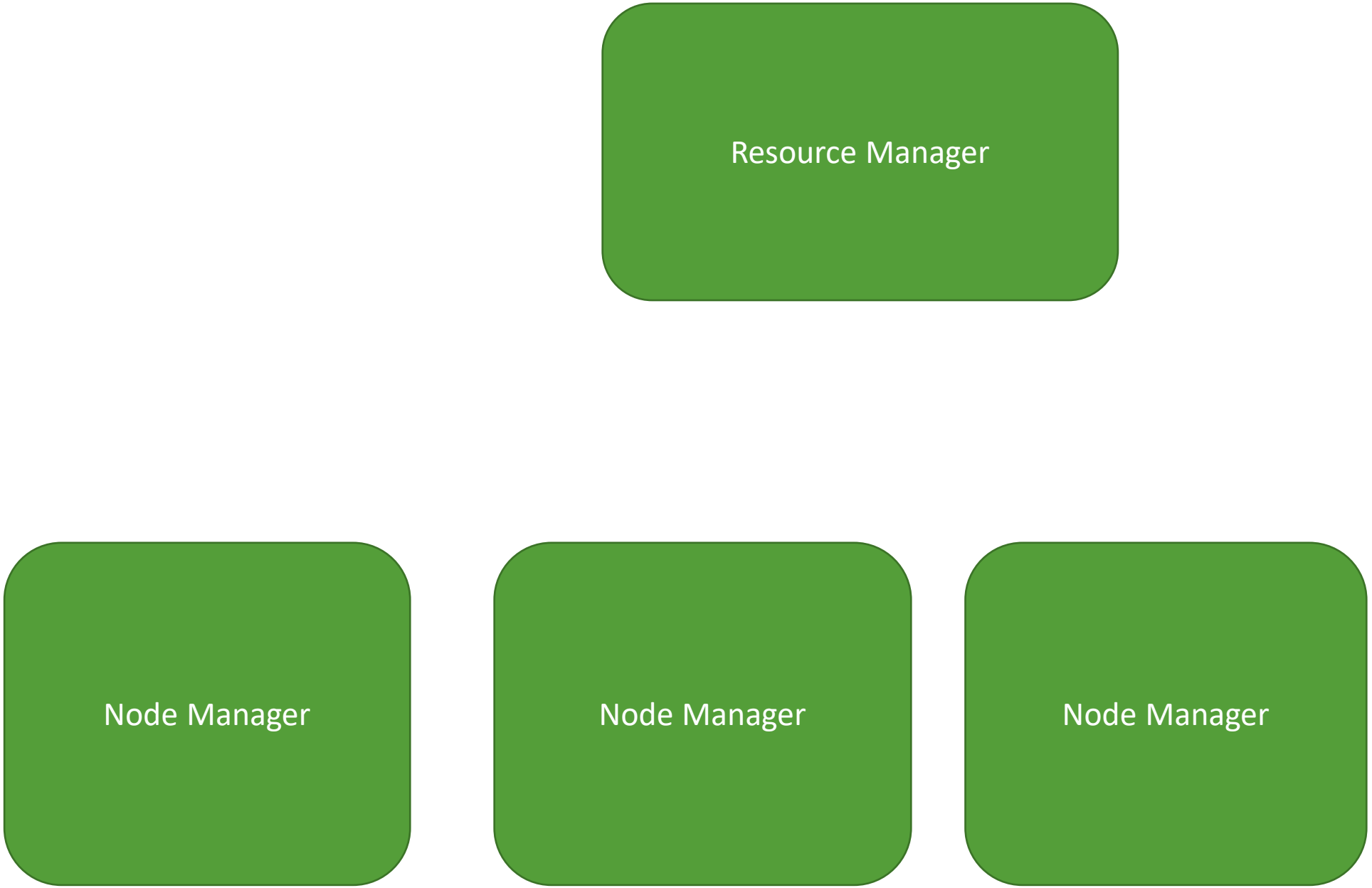

Worker Machine
Worker Machine
Worker Machine
Node Manager
Node Manager
Node Manager
Resource Manager

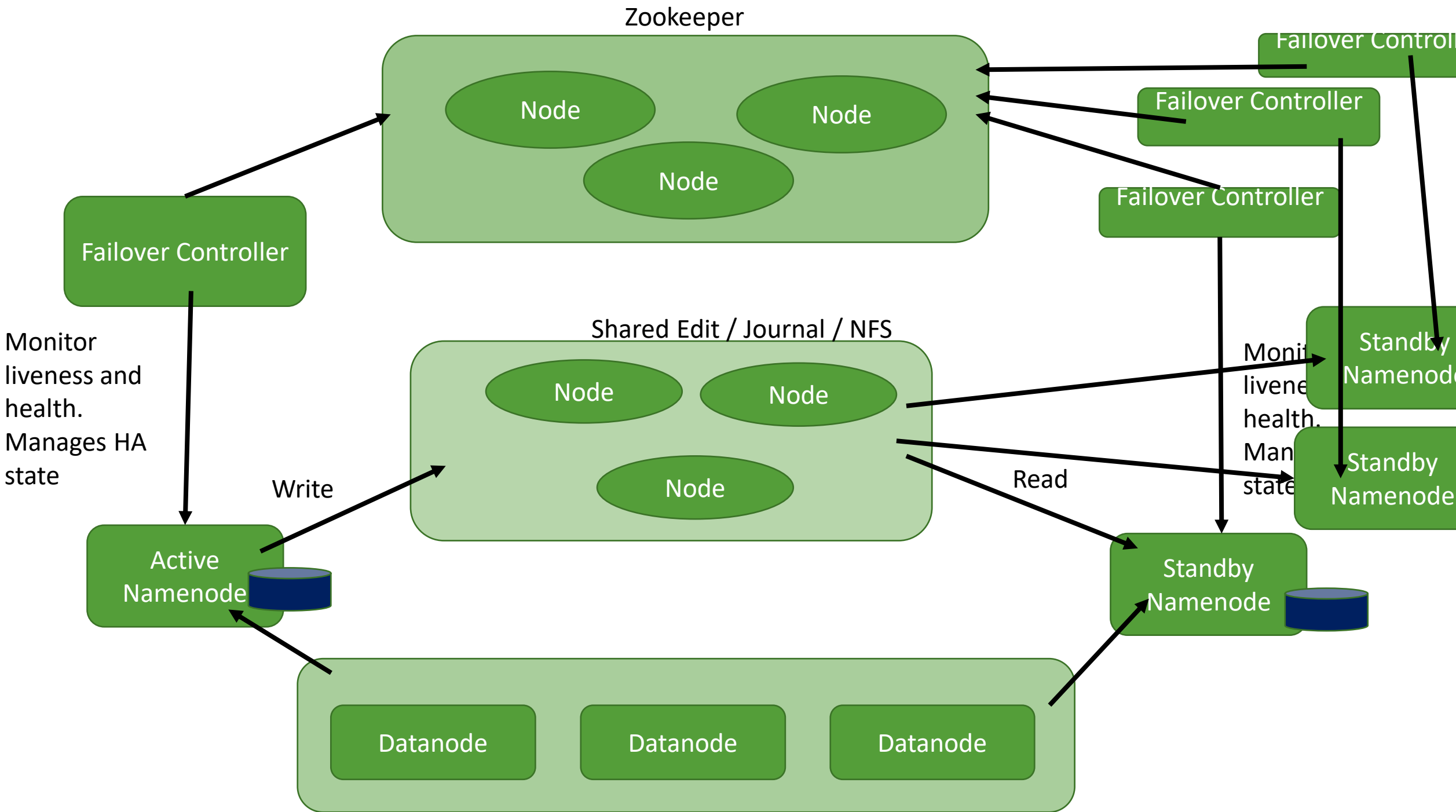
HDFS and YARN



Hadoop EcoSystem							
Data Visualization							
	SAS Visual Analytics	Tableau	Qlick	SAP Lumira	R		
	D3.js	iCharts	Timeline JS	Apache Zeppelin	Pentaho		
System Deployment							
	Apache Ambari	Apache Mesos	Marathon	Apache Bigtop	HortonWorks HOYA	Deploop	
	Apache Eagle	Cloudera Manager	Brooklyn	Apache Helix	Buildoop	SequenceIQ Cloudbreak	
	Myriad						
Data Ingestion	**Service Programming**		**Scheduling & DR**	**Security**	**Frameworks**	**Metadata**	**Machine Learning**
Apache Flume	Apache Thrift	Apache Karaf	Apache Oozie	Apache Sentry	Jumbune	Metascope	Apache Mahout
Apache Sqoop	Apache Zookeeper	Twitter Elephant Bi	Linkedin Azkaban	Apache Knox Gatew	Spring XD	Apache Tika	WEKA
Facebook Scribe	Apache Avro	Linkedin Norbert	Apache Falcon	Apache Ranger	Cask Data App Platform		Cloudera Oryx
Apache Chukwa	Apache Curator		Shedoscope				Deeplearning4j
Apache Kafka							MADlib
Netflix Suro	**Distributed Programming**				**SQL on Hadoop**		H2O
Apache Samza	Apache Ignite	Apache Flink	Apache Twill	Kangaroo	Apache Hive	Facebook Presto	Sparkling Water
Cloudera Morphlin	Apache Mapreduce	Apache Apex	Damballa PARKOUR	TinkerPop	Apache Hcatalog	Datasalt Splout SQL	Apache SystemML
HIHO	Apache Pig	Netflix Pigpen	Apache Hama	Pachyderm Mapred	Apache Trafodion	Apache Tajo	
Apache Nifi	JAQL	AMPLAB SIMR	Datasalt Pangool	Apache Beam	Apache HAWQ	Apache MRQL	
Apache ManifoldCF	Apache Spark	Facebook Corona	Apache Tez		Apache Drill	Kylin	
	Apache Storm	Apache REEF	Apache DataFu		Cloudera Impala		
	NoSQL Databases					**NewSQL Databases**	**Deep Learning**
	Wide Column	**Document**	**Key-Value**	**Graph**	**Stream Data Model**	TokuDB	Caffe
	Apache Hbase	MongoDB	Redis	Giraph	EventStore	HandlerSocket	Microsoft CNTK
	Apache Cassandra	RethinkDB	Linkedin Voldemort	Neo4J		Akiban Server	Tensor Flow
	Hypertable	ArangoDB	RocksDB	TitanDB		Drizzle	Theano
	Apache Accumulo	CouchDB	OpenTSDB	OrientDB		Haeinsa	Torch
	Apache Kudu	DynamoDB				SenseiDB	Mxnet
	Apache Parquet	Gemfire				Sky	Chainer
						BayesDB	Keras
						InfluxDB	
						VoldDB	
						SAP HANA	
Distributed File System							
	Apache HDFS	Quantcast File Syste	Lustre File System	GridGain			
	Red Hat GlusterFS	Ceph File System	Alluxio	XtreemFS			


MapReduce
HDFS – Hadoop Distributed File System

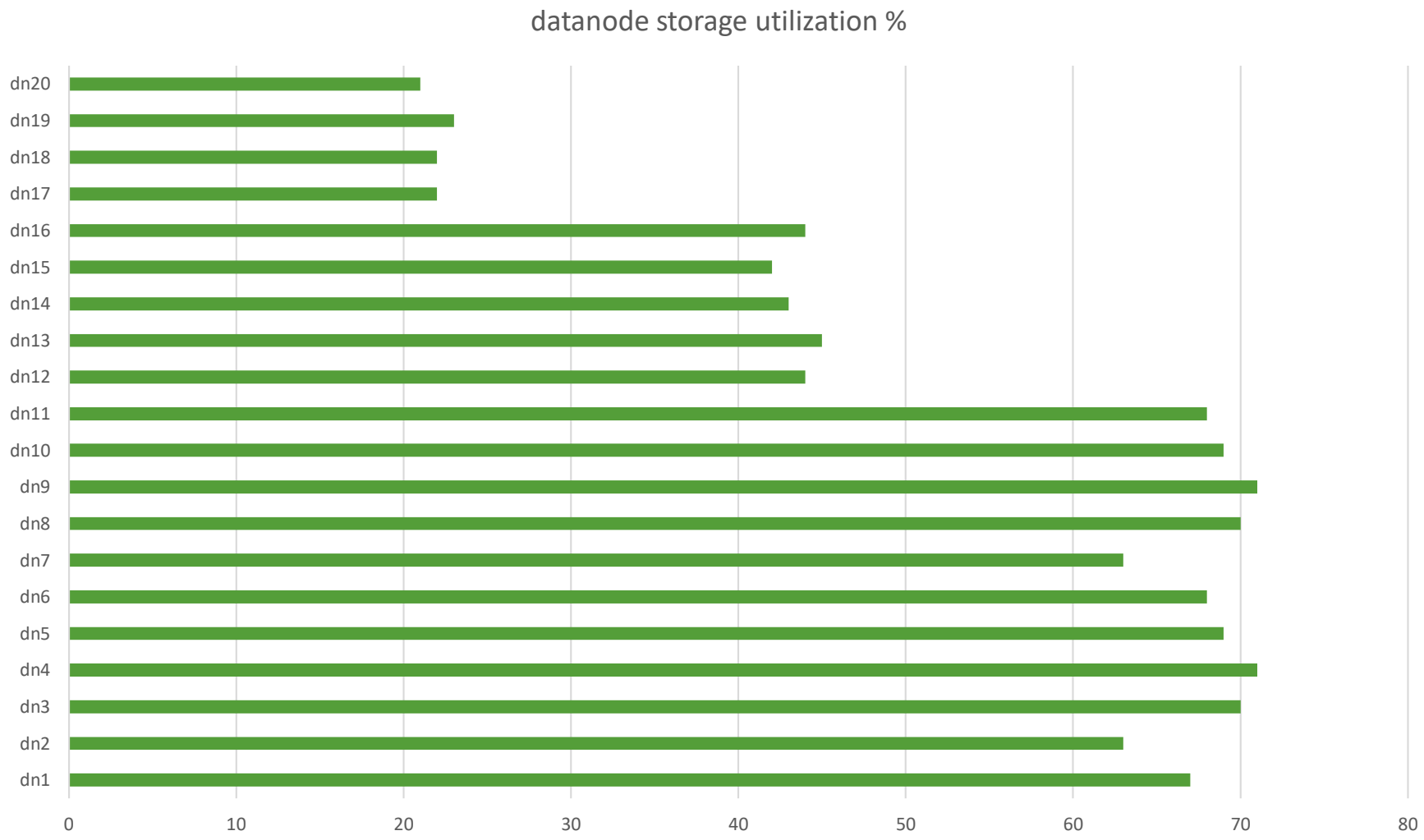

Resource Manager
Node Manager
Node Manager
Node Manager

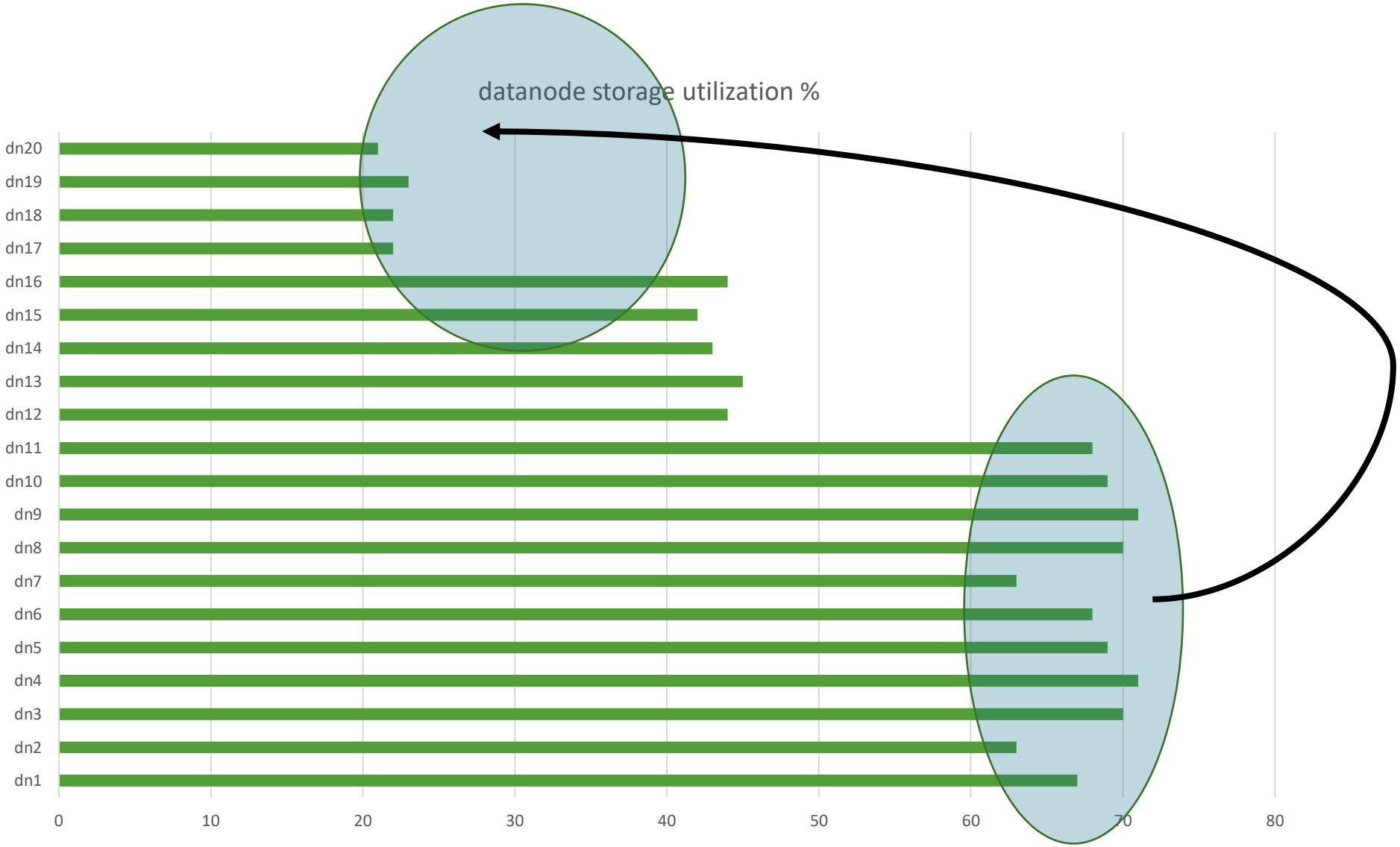

Zookeeper
Node
Node
Node
Failover Controller
Failover Controller
Failover Controller
Failover Controller
Monitor liveness and health. Manages HA state
Shared Edit / Journal / NFS
Node
Node
Node
Write
Read
Active Namenode
Standby Namenode
Standby Namenode
Standby Namenode
Datanode
Datanode
Datanode

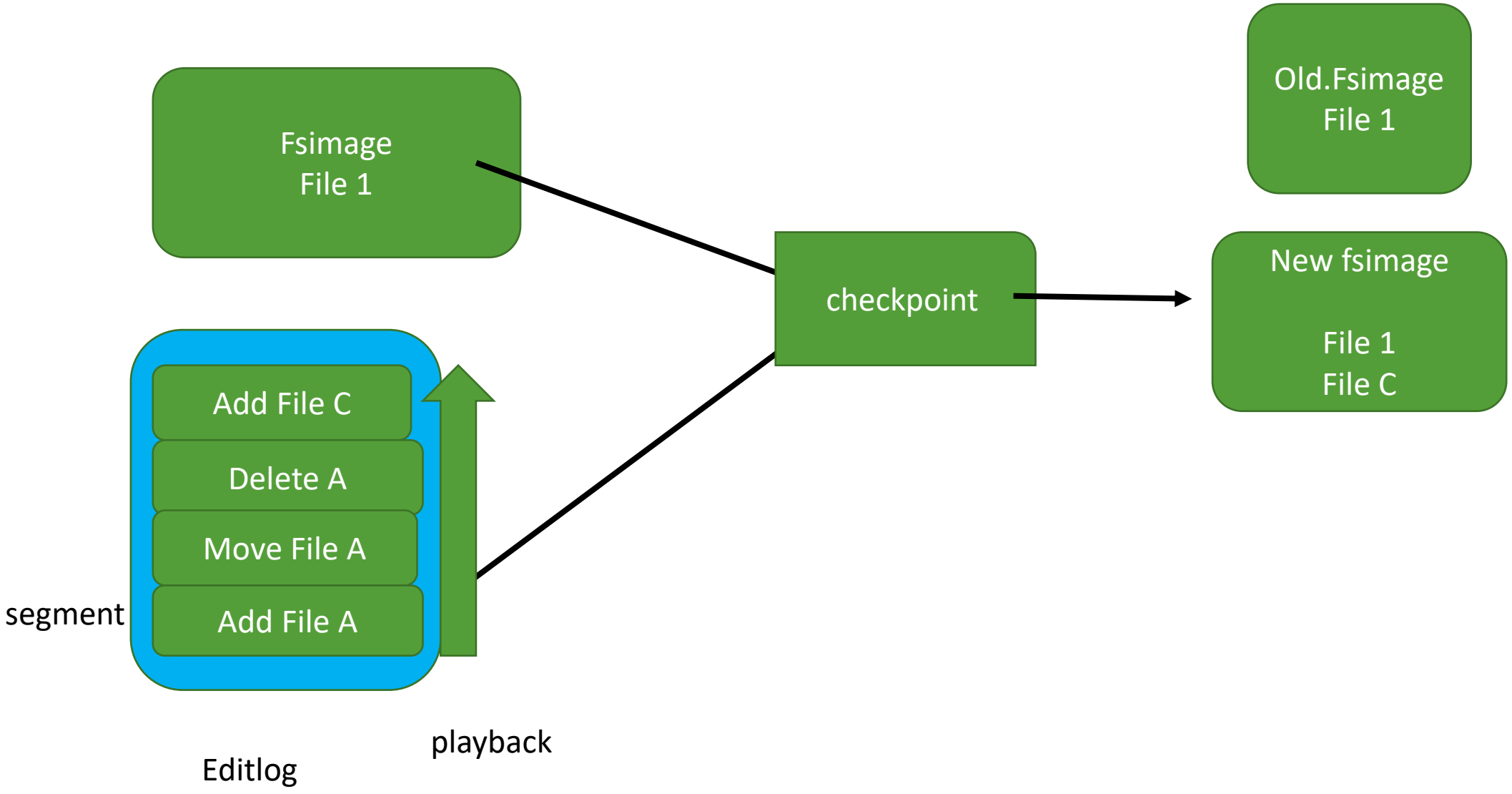
HDFS Roles

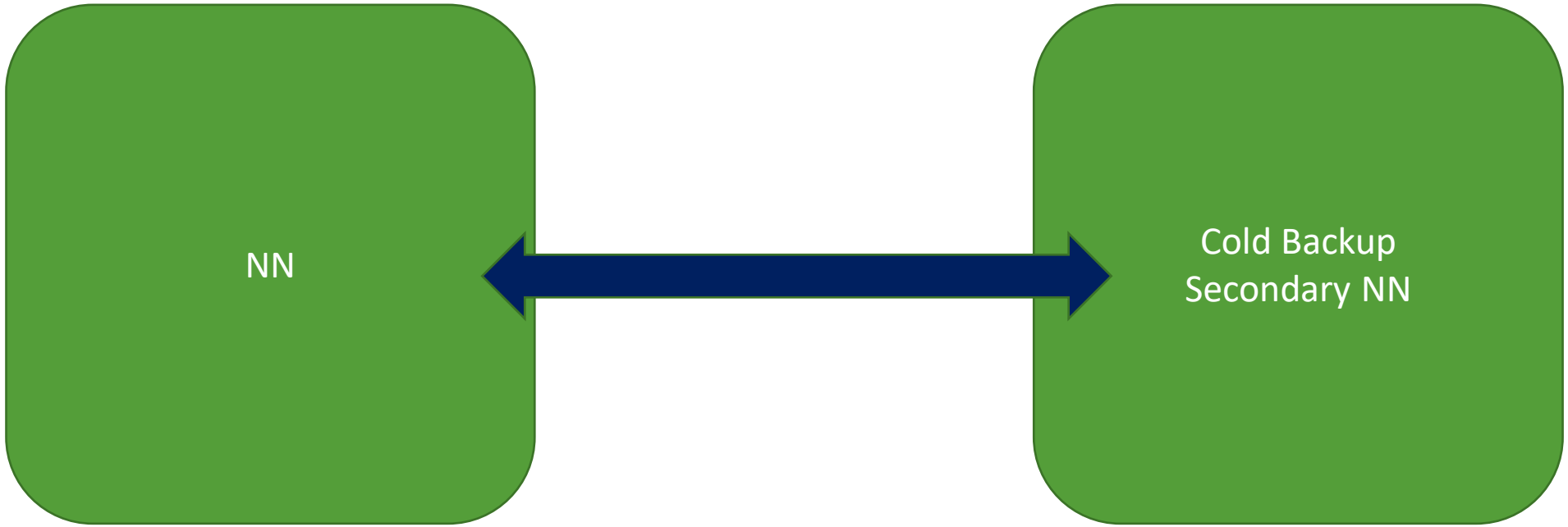
- Namenode
- Secondary Namenode
- Balancer
- HttpFS
- NFS Gateway
- Datanode
- Failover Controller
- Gateway
- JournalNode

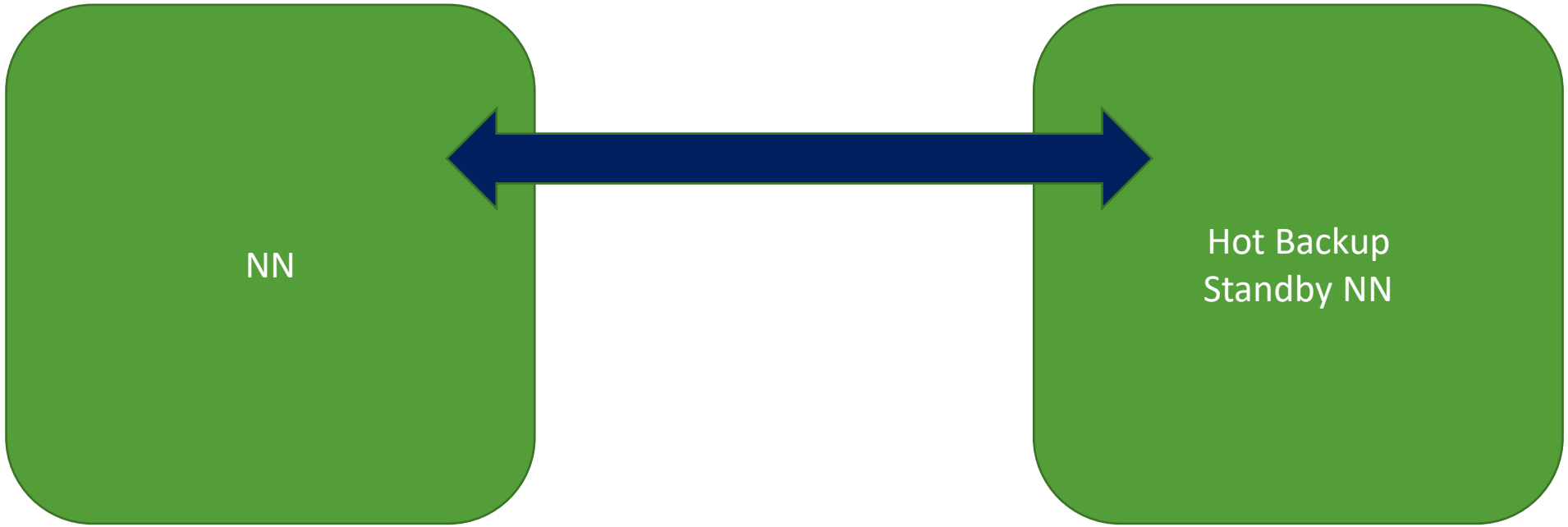
Hadoop Ecosystem

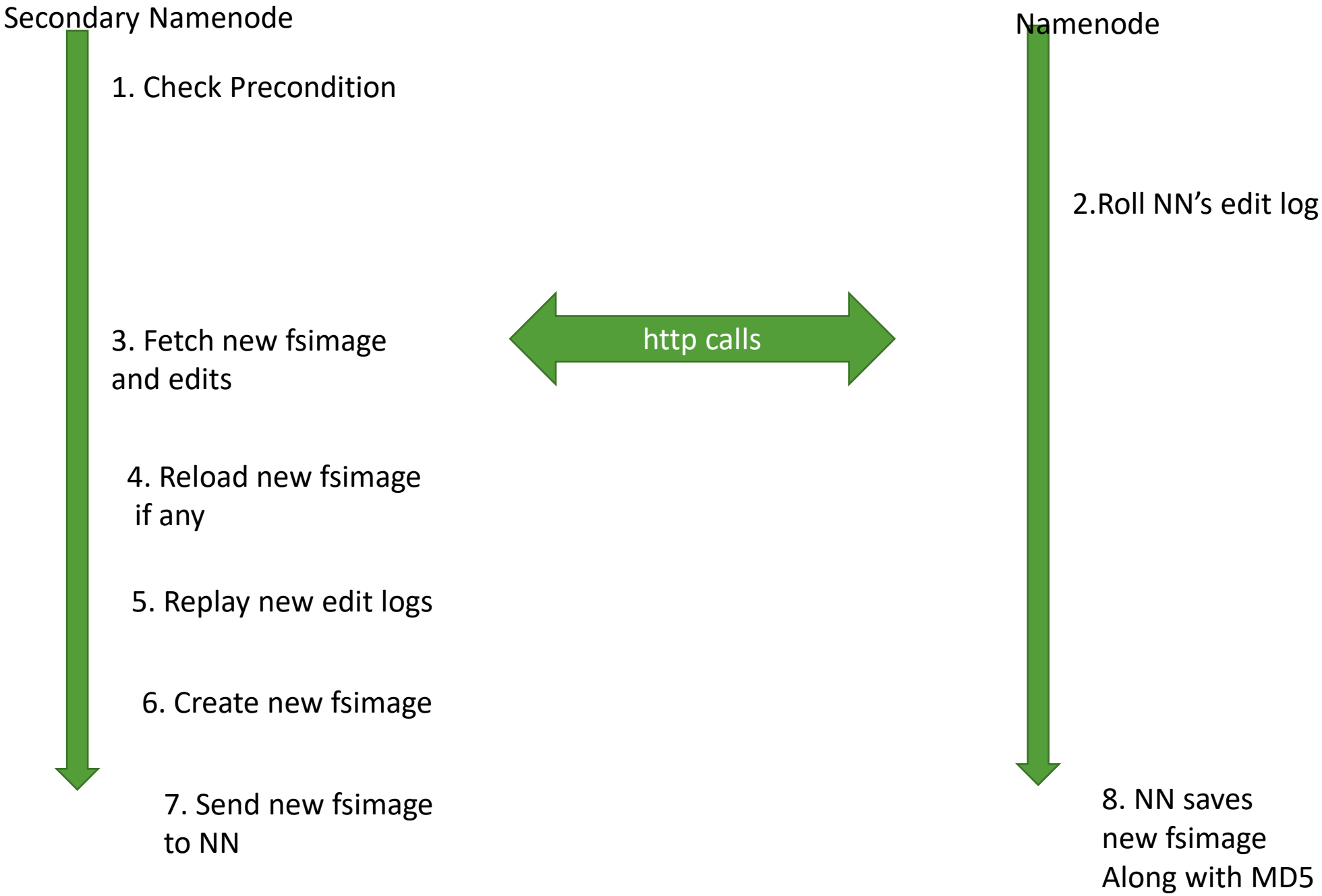

datanode storage utilization %
dn20
dn19
dn18
dn17
dn16
dn15
dn14
dn13
dn12
dn11
dn10
dn9
dn8
dn7
dn6
dn5
dn4
dn3
dn2
dn1
0
10
20
30
40
50
60
70
80

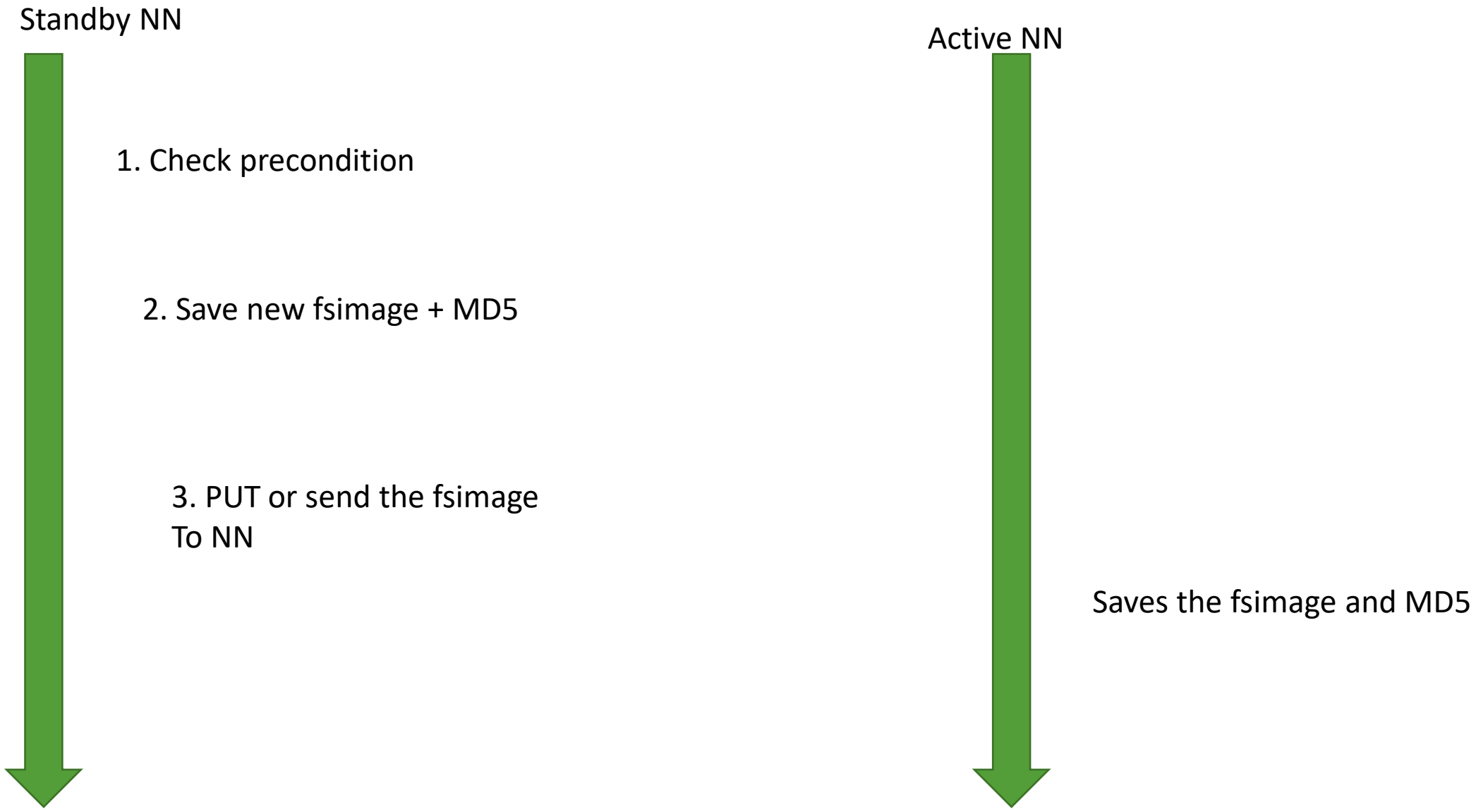

datanode storage utilization %
dn20
dn19
dn18
dn17
dn16
dn15
dn14
dn13
dn12
dn11
dn10
dn9
dn8
dn7
dn6
dn5
dn4
dn3
dn2
dn1
0
10
20
30
40
50
60
70
80

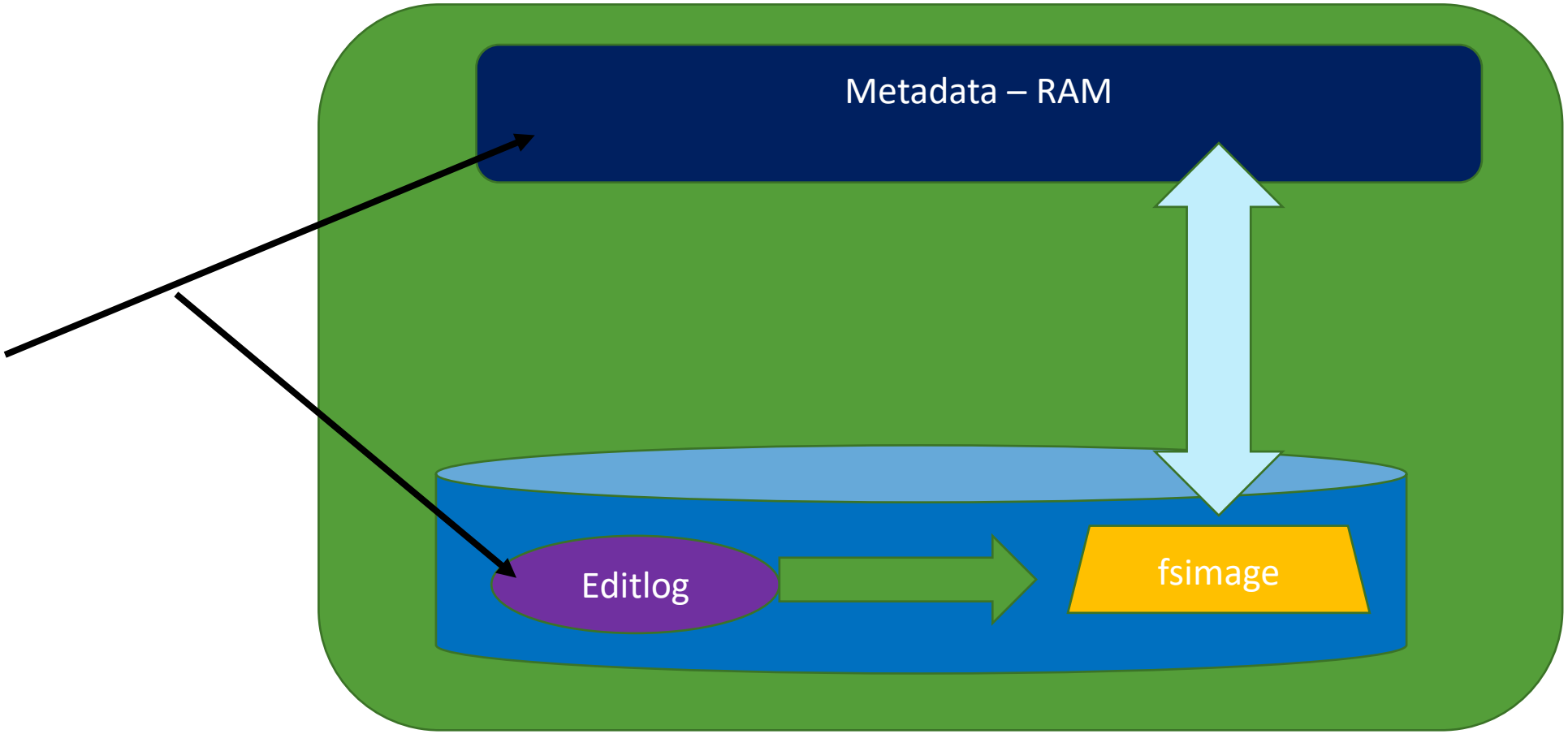

Fsimage
File 1
Old.Fsimage
File 1
checkpoint
New fsimage
File 1
File C
Add File C
Delete A
Move File A
Add File A
segment
Editlog
playback


NN
Cold Backup
Secondary NN


NN
Hot Backup
Standby NN


Secondary Namenode
1. Check Precondition
3. Fetch new fsimage
and edits
4. Reload new fsimage
if any
5. Replay new edit logs
6. Create new fsimage
7. Send new fsimage
to NN
http calls
Namenode
2.Roll NN's edit log
8. NN saves
new fsimage
Along with MD5

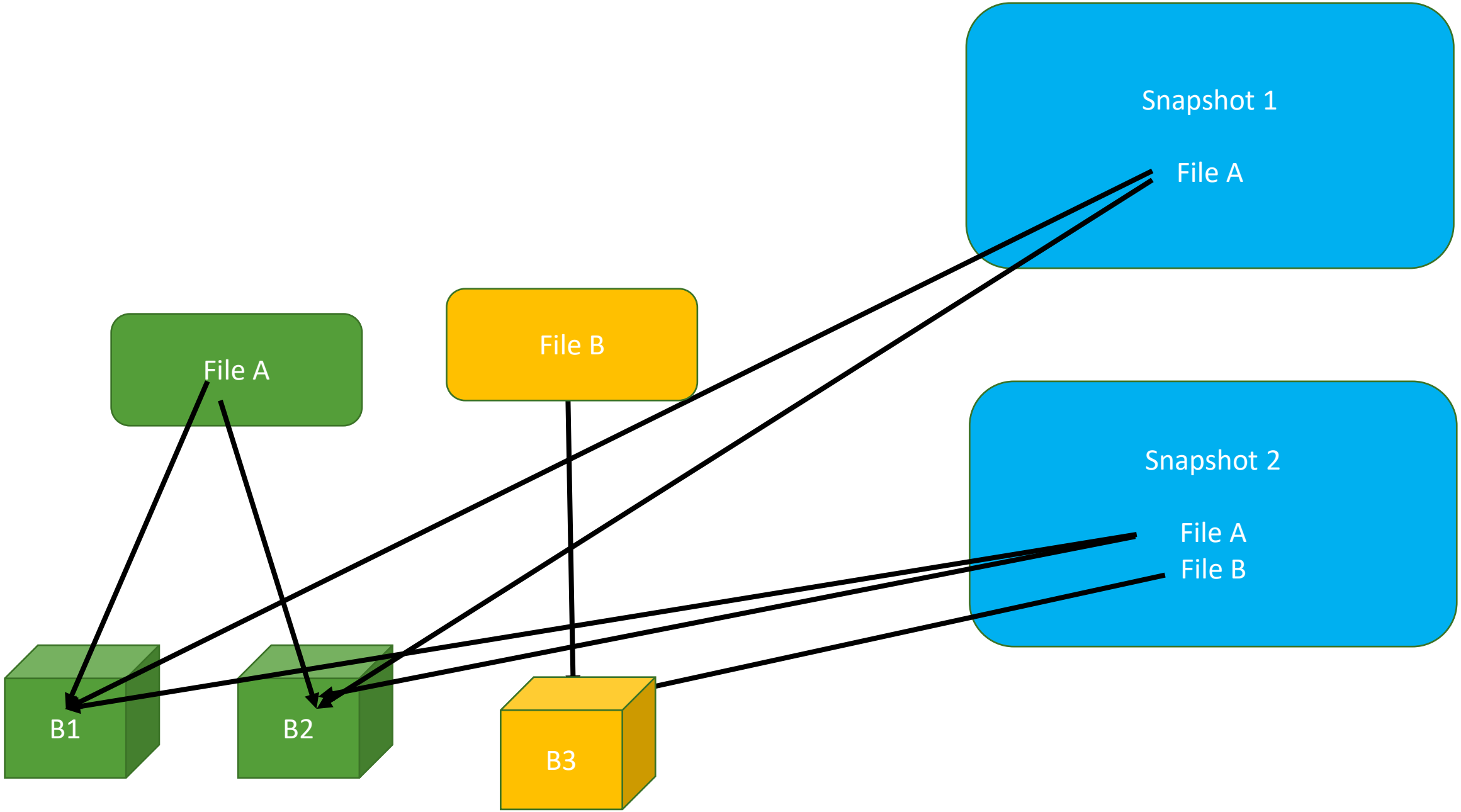

Standby NN
Active NN
1. Check precondition
2. Save new fsimage + MD5
3. PUT or send the fsimage
To NN
Saves the fsimage and MD5

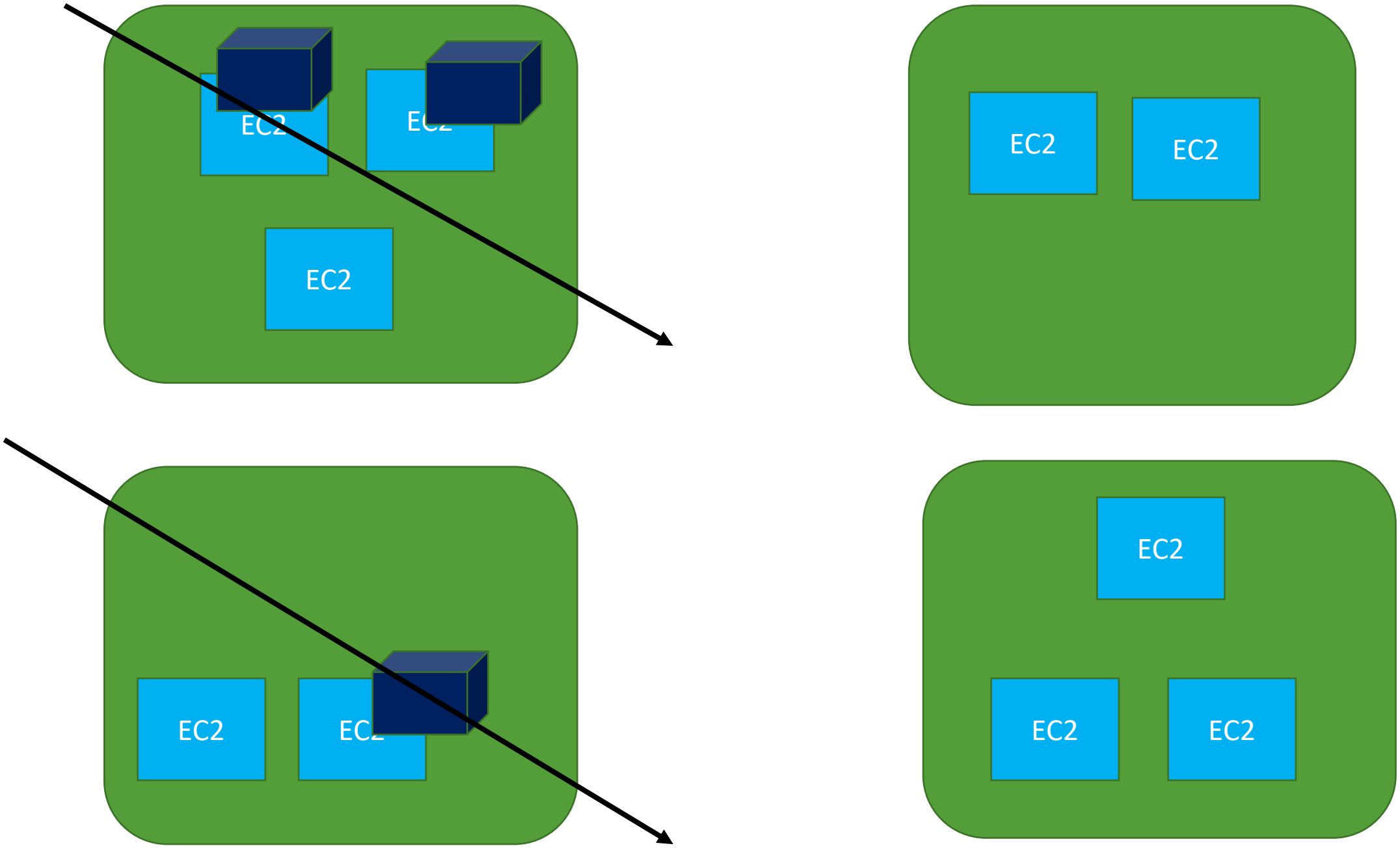

Metadata – RAM
Editlog
fsimage

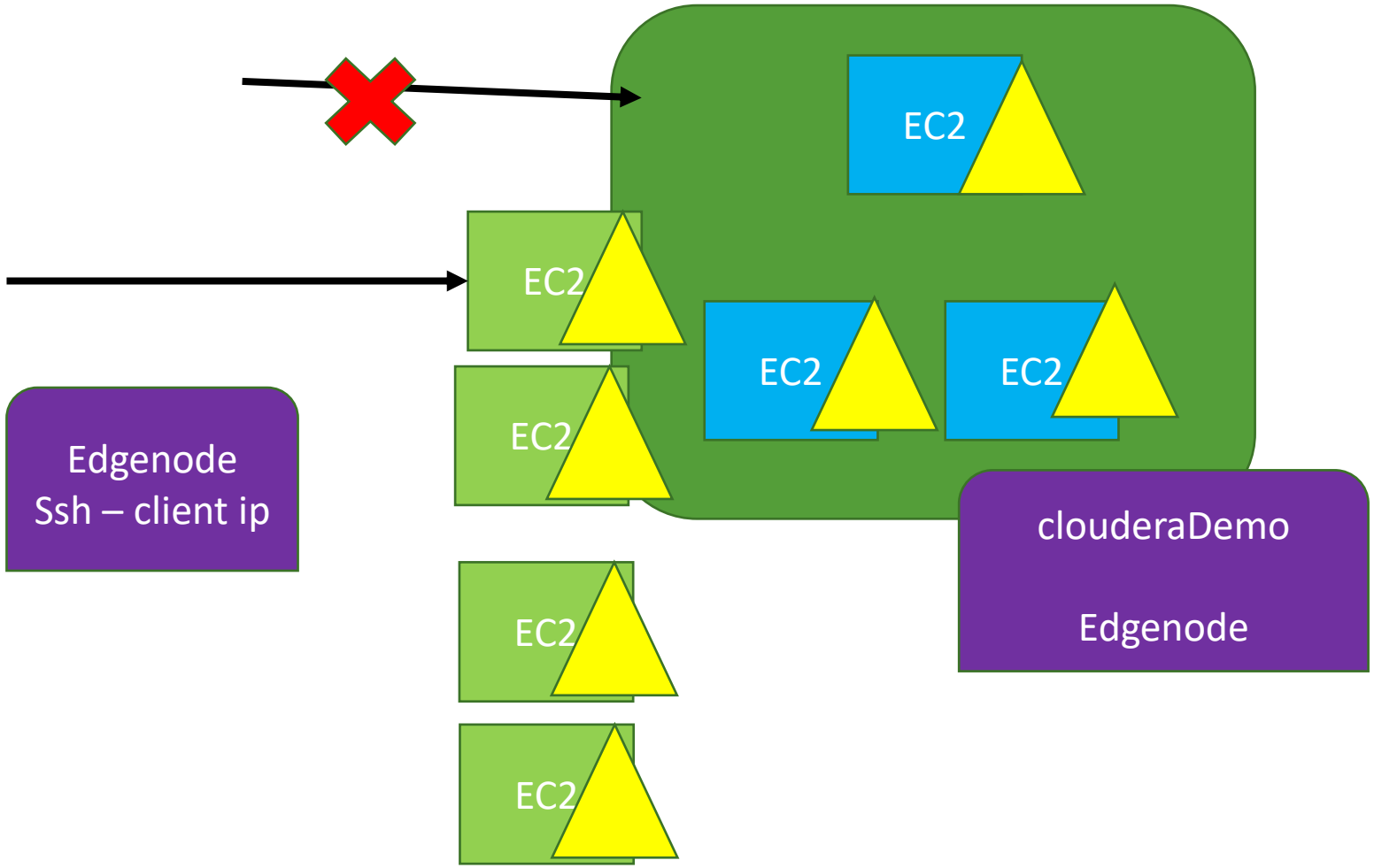
```
data/dfs/name
+-- current
¦ +-- VERSION
¦ +-- edits_0000000000000000001-0000000000000000007
¦ +-- edits_0000000000000000008-0000000000000000015
¦ +-- edits_0000000000000000016-0000000000000000025
¦ +-- edits_0000000000000000026-0000000000000000028
¦ +-- edits_0000000000000000029-0000000000000000030
¦ +-- edits_0000000000000000031-0000000000000000033
¦ +-- edits_inprogress_0000000000000000034
¦ +-- fsimage_0000000000000000030
¦ +-- fsimage_0000000000000000030.md5
¦ +-- fsimage_0000000000000000033
¦ +-- fsimage_0000000000000000033.md5
¦ +-- seen_txid
+-- in_use.lock
```

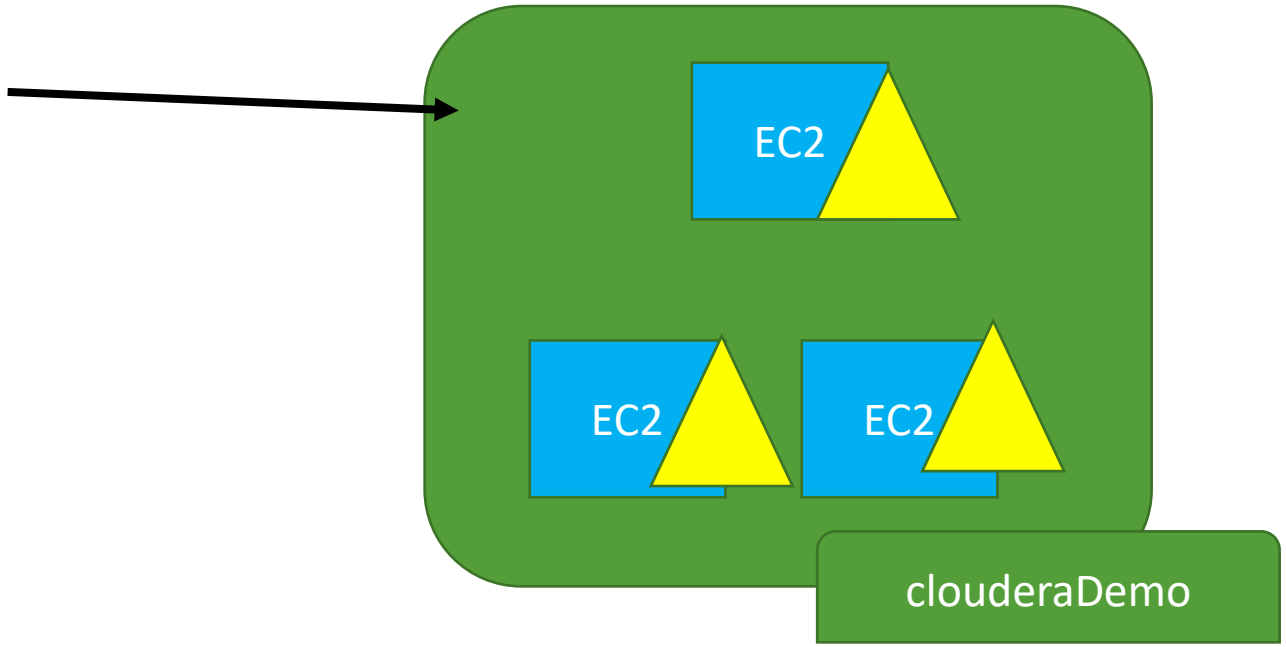
Property that controls checkpoint

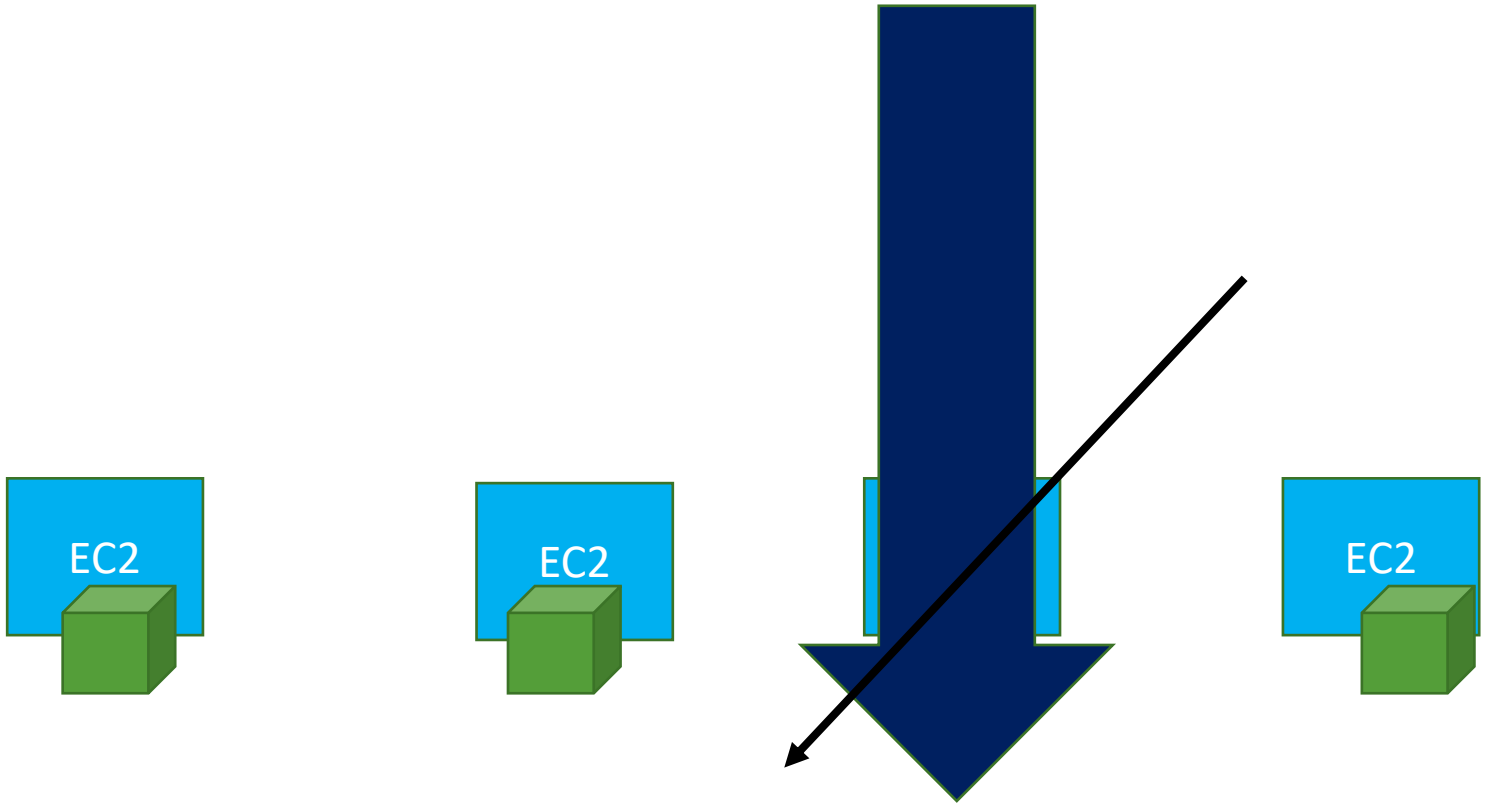
dfs.namenode.checkpoint.period - 3600
dfs.namenode.checkpoint.txns – 1000000
dfs.namenode.checkpoint.check.period - 60
dfs.namenode.num.checkpoints.retained - 1000000
dfs.namenode.num.extra.edits.retained - 2
dfs.namenode.max.extra.edits.segments.retained - 10000


Snapshot 1
File A
File A
File B
Snapshot 2
File A
File B
B1
B2
B3


EC2
EC2
EC2
EC2
EC2
EC2
EC2
EC2
EC2


EC2
EC2
EC2
EC2
EC2
Edgenode
Ssh – client ip
EC2
EC2
clouderaDemo
Edgenode


EC2
EC2
EC2
clouderaDemo

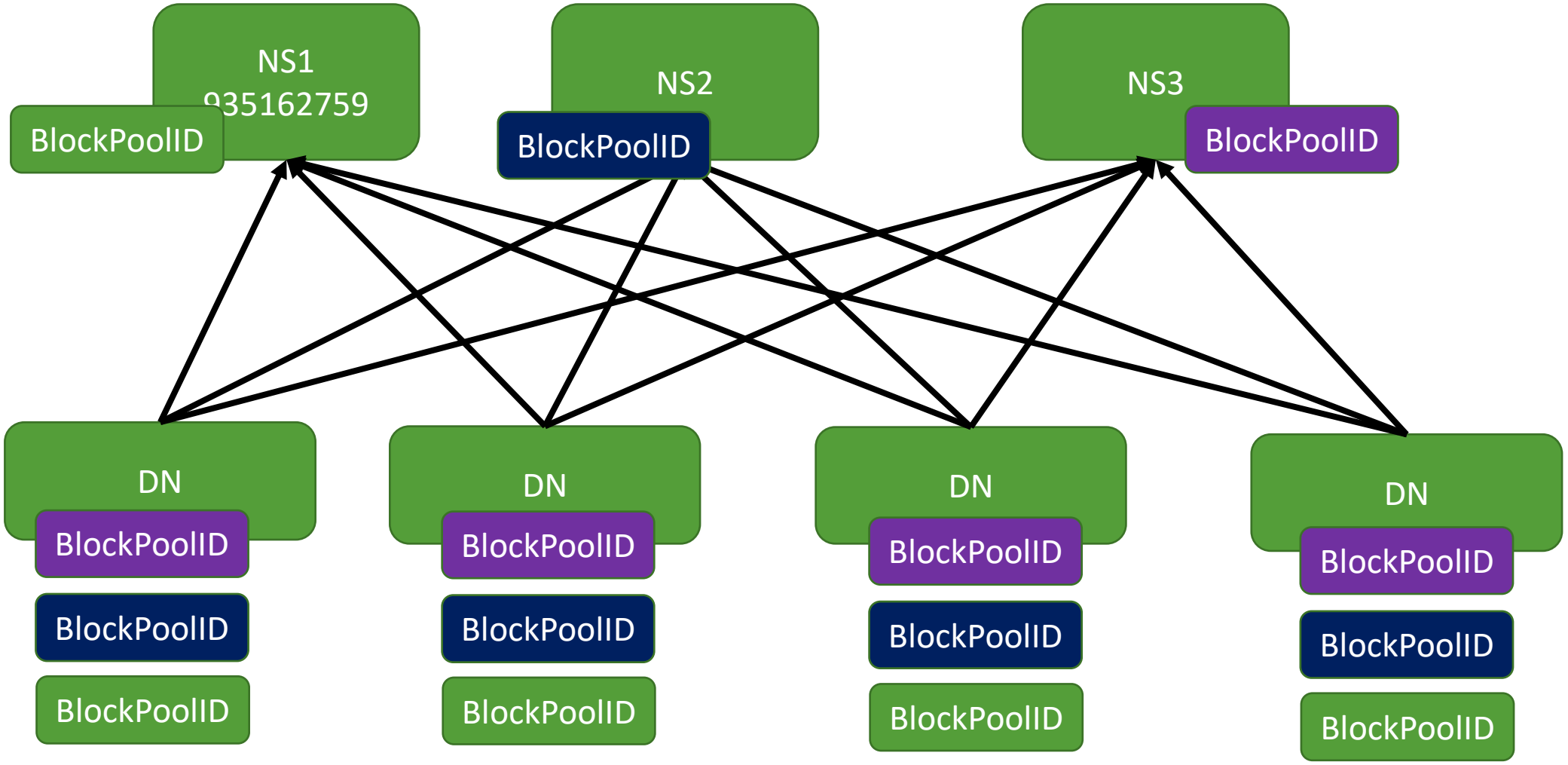

EC2
EC2
EC2

WebHDFS

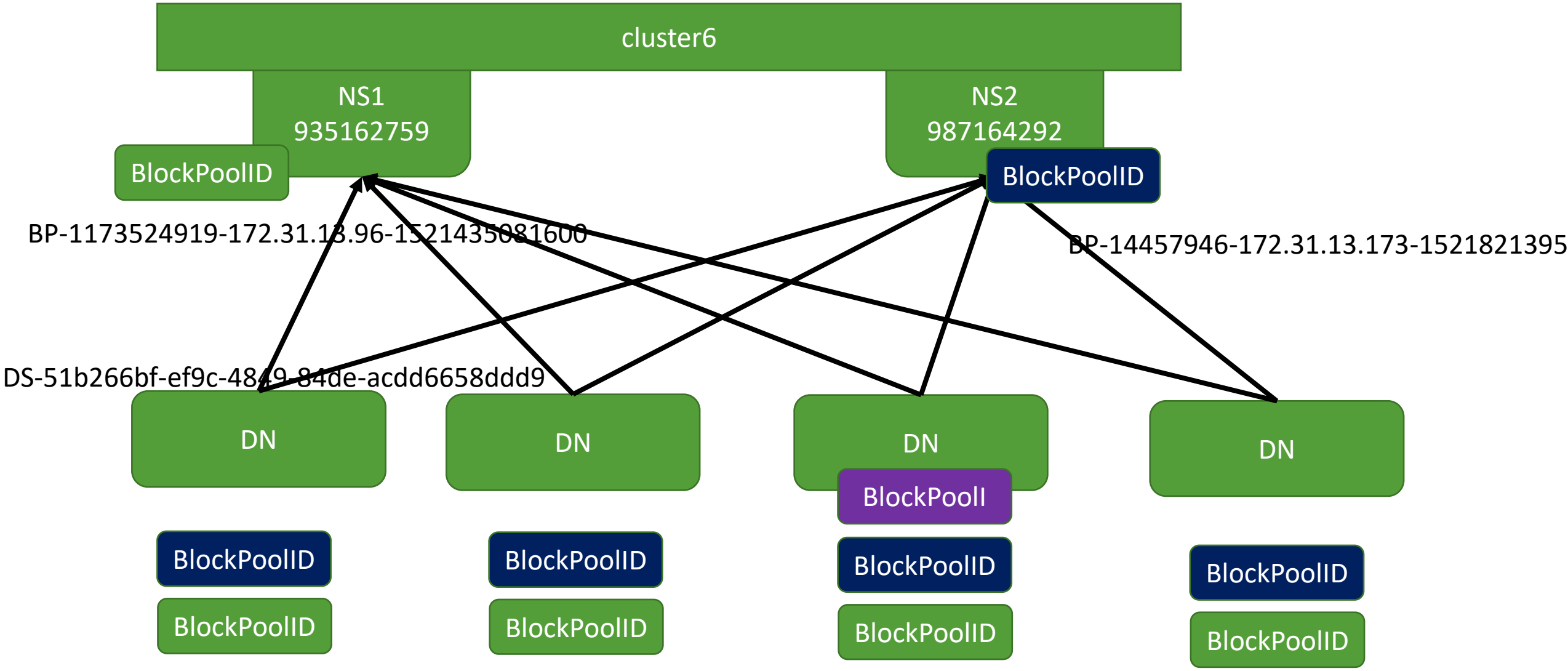
- Complete HDFS Interface (Read, Write, Directory Operations, Permissions)
- HTTP REST API
- Wire Compatibility – Can talk to different version of Hadoop
- Secure Authentication – Can be used along with Kerberos
- Data Locality – Calls will be redirected to Datanode for Read/Write
- HDFS Built-in Component

Federation

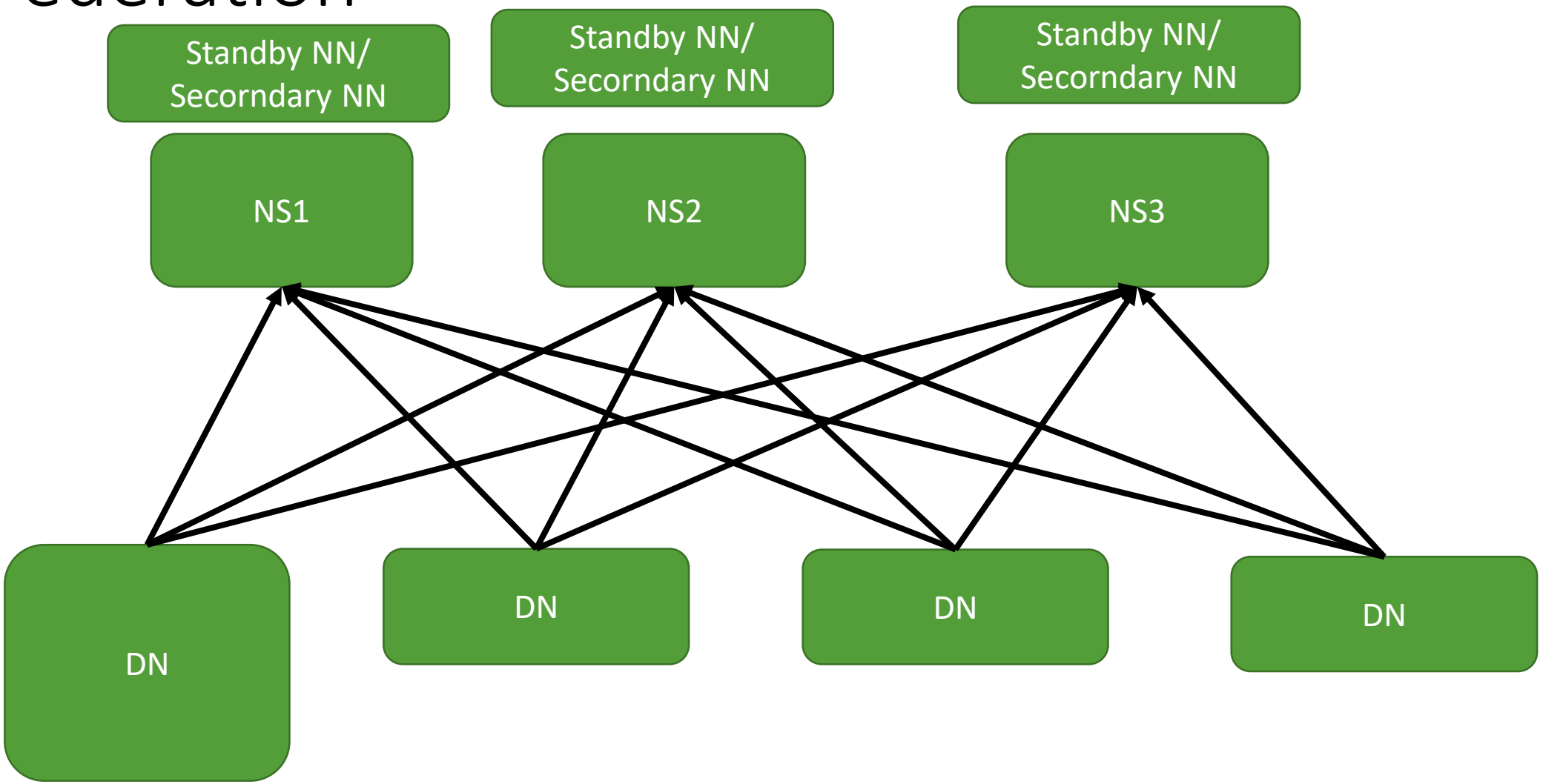
BP-1173524919-172.31.13.96-1521435081600



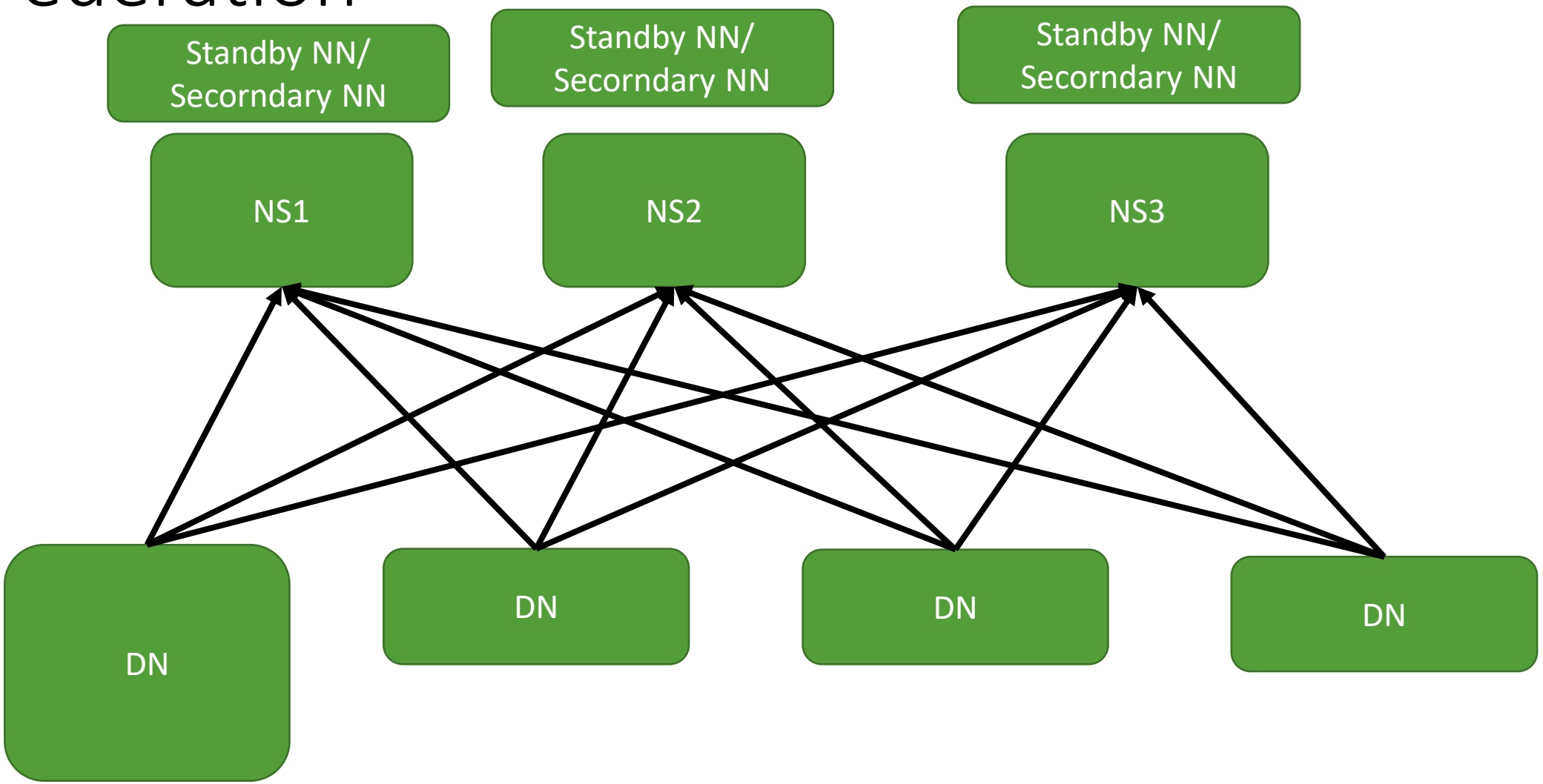
Federation


cluster6
NS1
935162759
NS2
987164292
BlockPoolID
BlockPoolID
BP-1173524919-172.31.13.96-1521435081600
BP-14457946-172.31.13.173-1521821395
DS-51b266bf-ef9c-4849-84de-acdd6658ddd9
DN
DN
DN
DN
BlockPoolI
BlockPoolID
BlockPoolID
BlockPoolID
BlockPoolID
BlockPoolID
BlockPoolID
BlockPoolID
BlockPoolID

Federation


Standby NN/
Secorndary NN
Standby NN/
Secorndary NN
Standby NN/
Secorndary NN
NS1
NS2
NS3
DN
DN
DN
DN

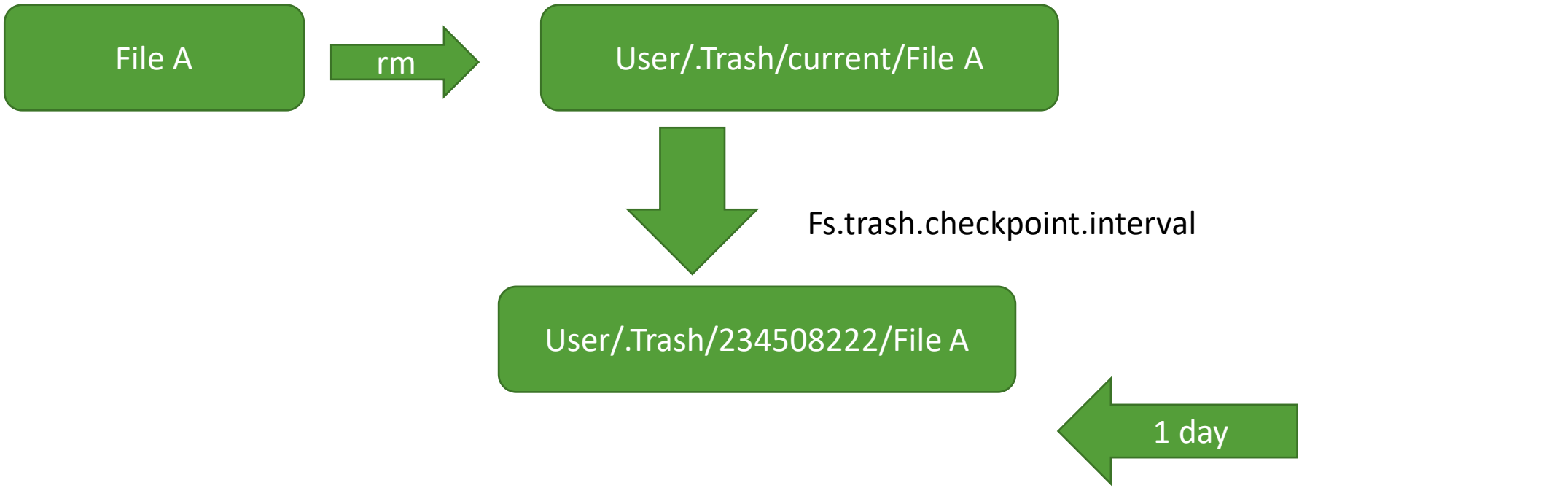
Federation

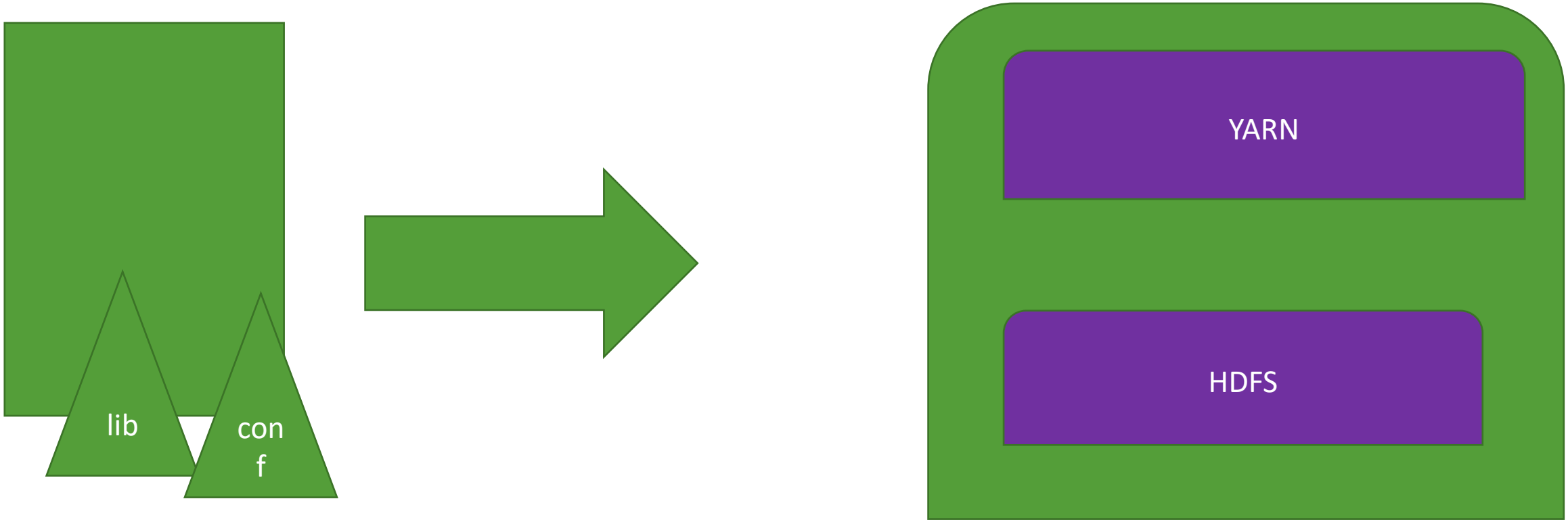

Standby NN/
Secorndary NN
Standby NN/
Secorndary NN
Standby NN/
Secorndary NN
NS1
NS2
NS3
DN
DN
DN
DN

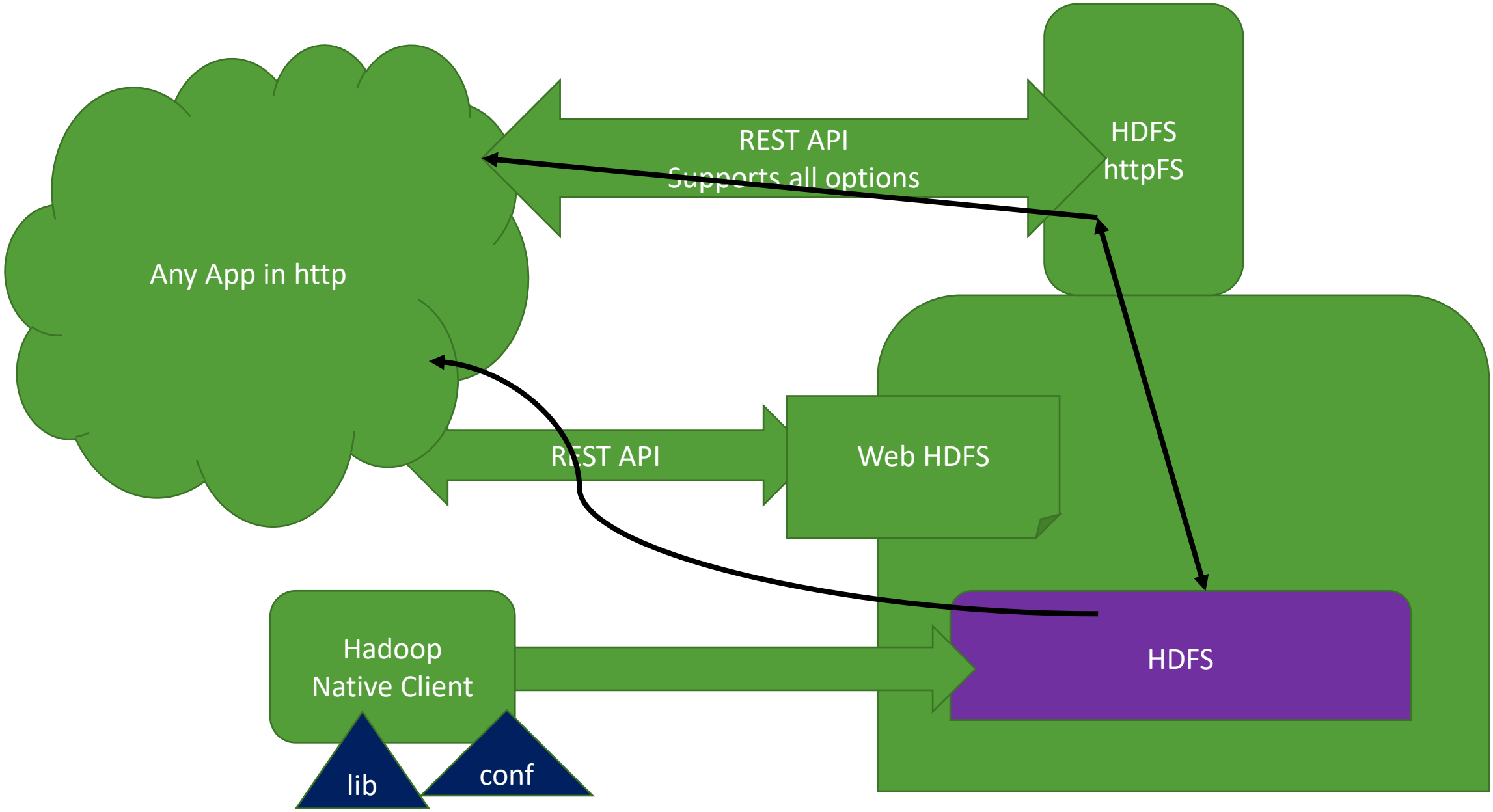
HDFS Trash

- HDFS trash overview
- -skipTrash
- Recover from trash
- Expunge
- Trash interval
- Trash checkpoint interval

Trash Process




lib
con
f
YARN
HDFS


Any App in http
REST API
Supports all options
HDFS
httpFS
REST API
Web HDFS
Hadoop
Native Client
lib
conf
HDFS

Web HDFS

- A Complete HDFS Interface
- HTTP REST API
- Secure Authentication
- Data Locality
- A HDFS Built-in Component
- WebHDFS needs access to all nodes of the cluster
- On read it is transmitted from that node directly

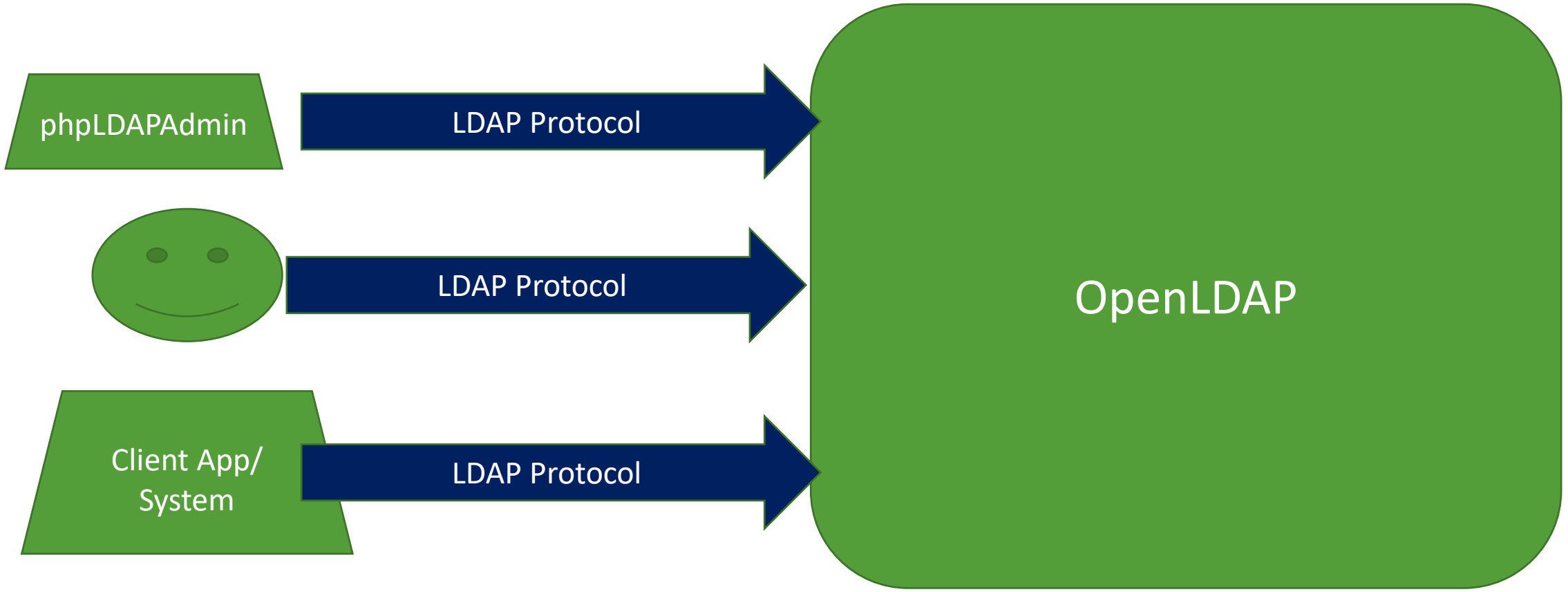
httpFS

- A singe node will act similar to a "gateway“
- A single point of data transfer to the client node
- Single Point of Failure
- Could be Over loaded during a large file transfer
- Reducers the footprint required to access HDFS
- Increases the security
- Preconfigured Tomcat bundled with HttpFS

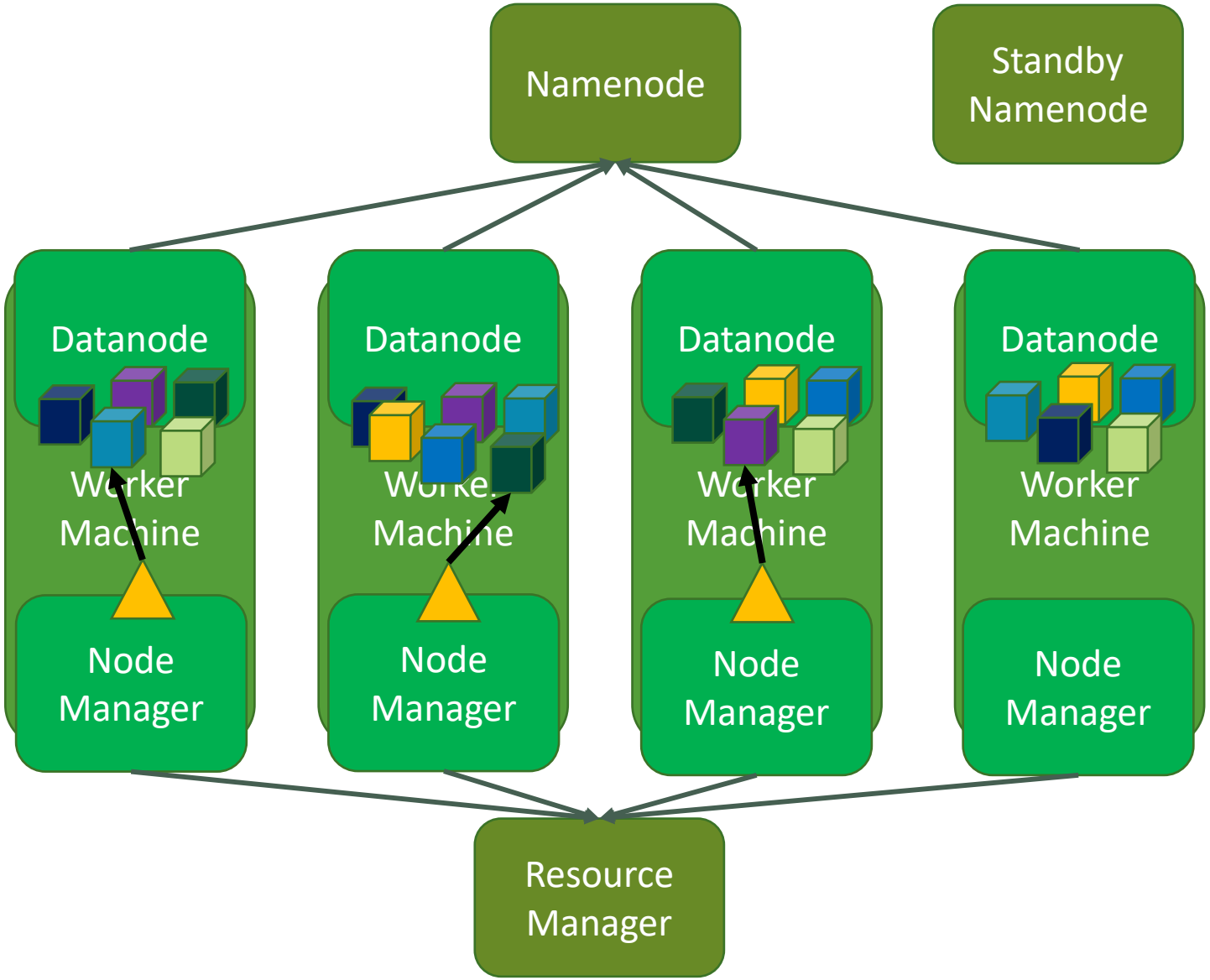
OpenLDAP

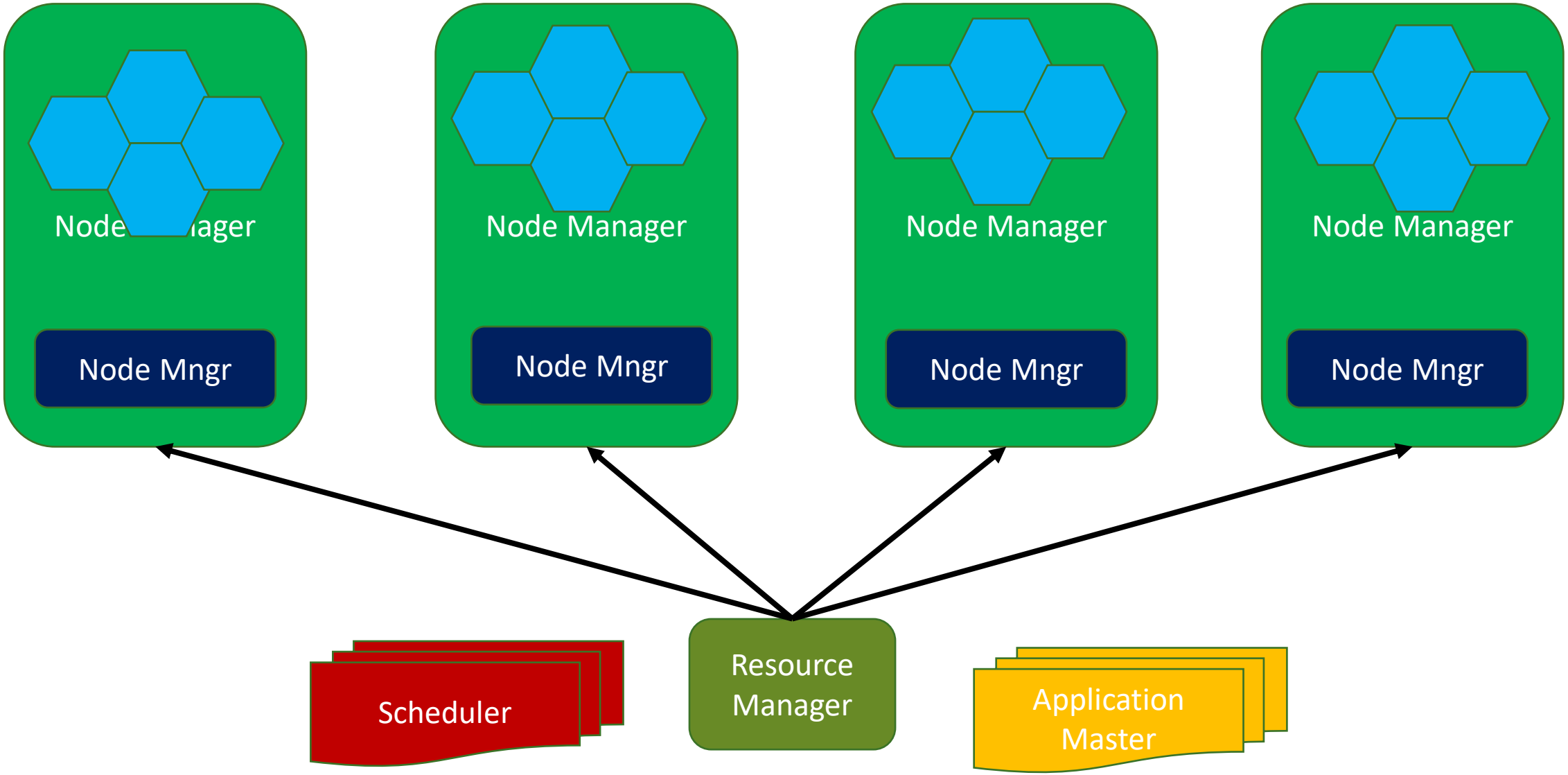
- Open-source and directory server
- organizational information stored on the server which can be used by client to search
- access, modify and manipulate entries in the directory.
- Client applications connect to OpenLDAP server using the Lightweight Directory Access Protocol (LDAP)
- User permissions can be controlled

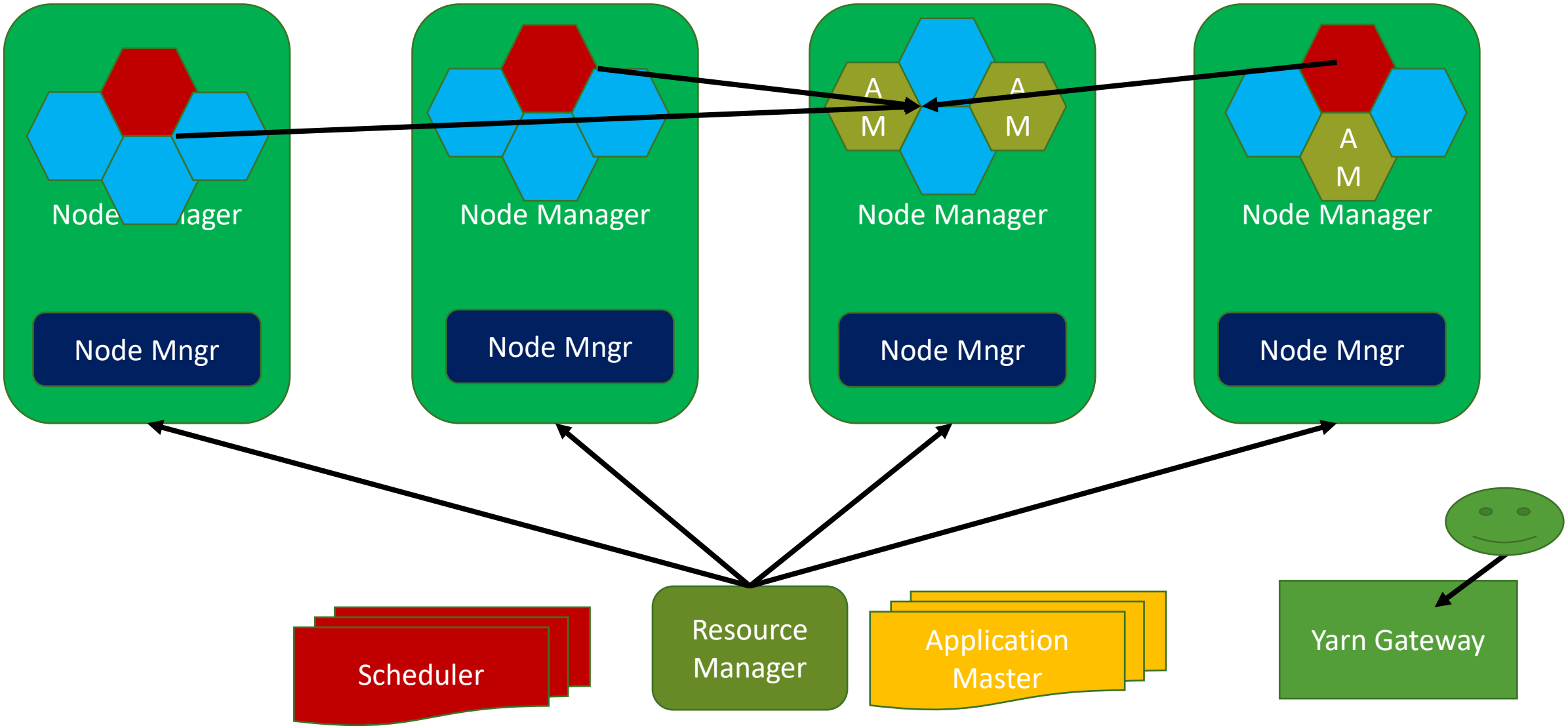
OpenLDAP

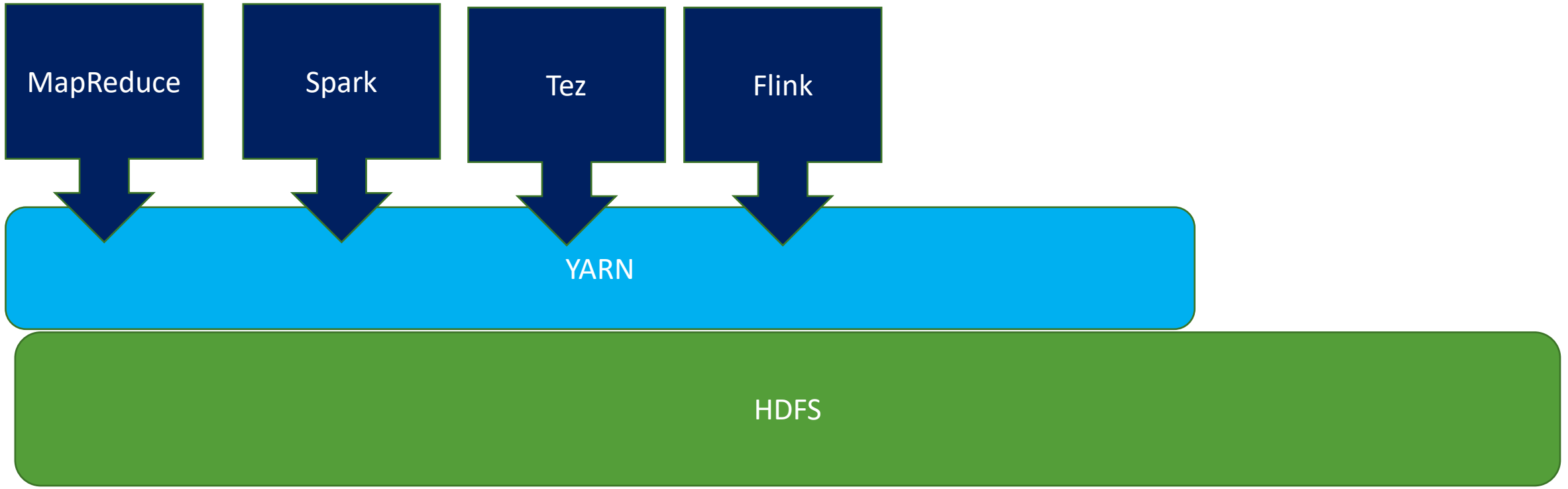

phpLDAPAdmin
LDAP Protocol
LDAP Protocol
Client App/
System
LDAP Protocol
OpenLDAP

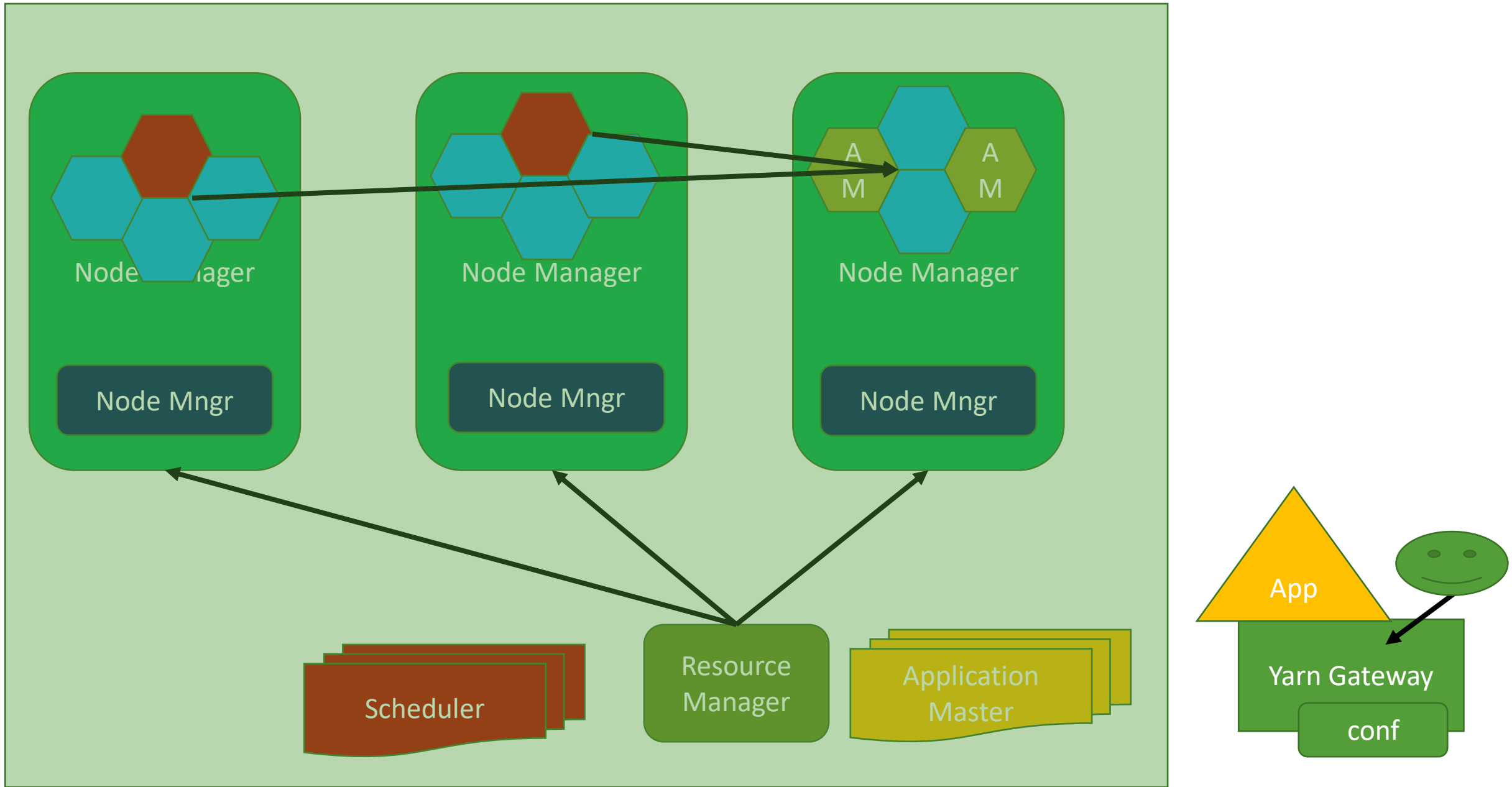
HDFS and YARN




Node Manager
Node Mngr
Node Manager
Node Mngr
Node Manager
Node Mngr
Node Manager
Node Mngr
Scheduler
Resource Manager
Application Master


Node Manager
Node Mngr
Node Manager
Node Mngr
A
M
A
M
Node Manager
Node Mngr
A
M
Node Manager
Node Mngr
Scheduler
Resource Manager
Application Master
Yarn Gateway

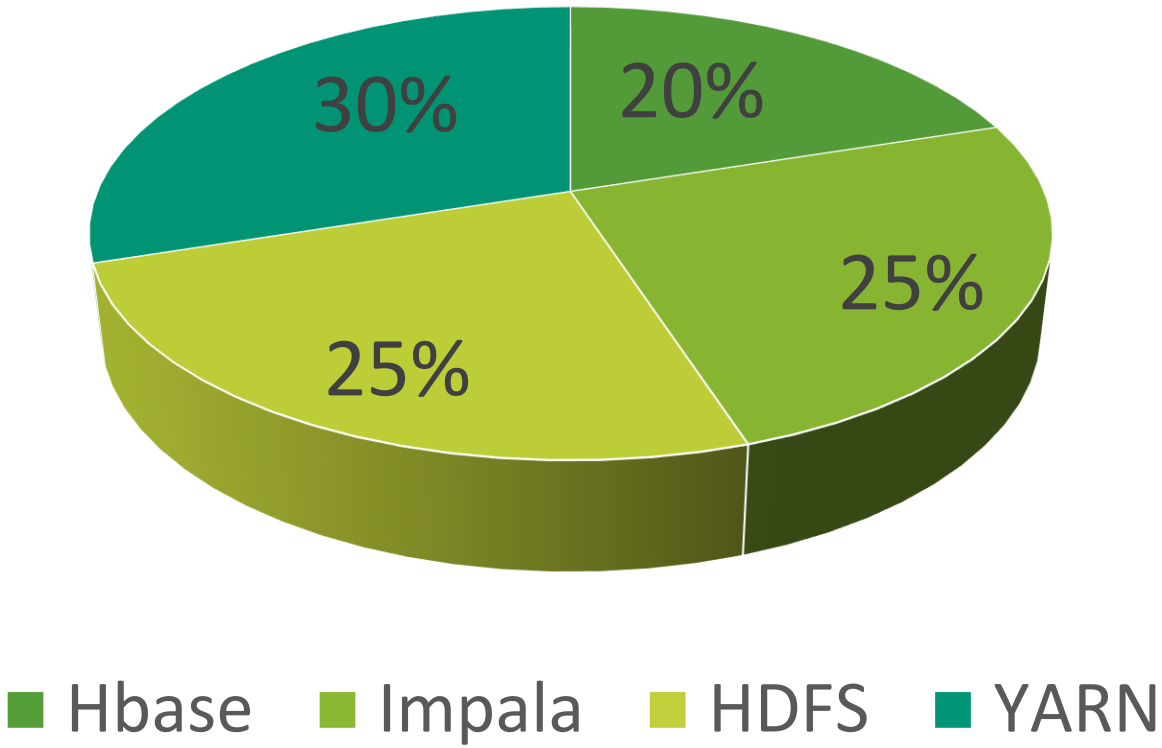

MapReduce
Spark
Tez
Flink
YARN
HDFS


Node Manager
Node Mngr
Node Manager
Node Mngr
A
M
A
M
Node Manager
Node Mngr
Scheduler
Resource Manager
Application Master
App
Yarn Gateway
conf

Resource Management

- Static Service Pools
- Dynamic Resource Pools
- Yarn (MRv2 and MapReduce (MRv1) Schedulers
- Resource Management for Impala (*Specific to Cloudera)

Static Service Pool



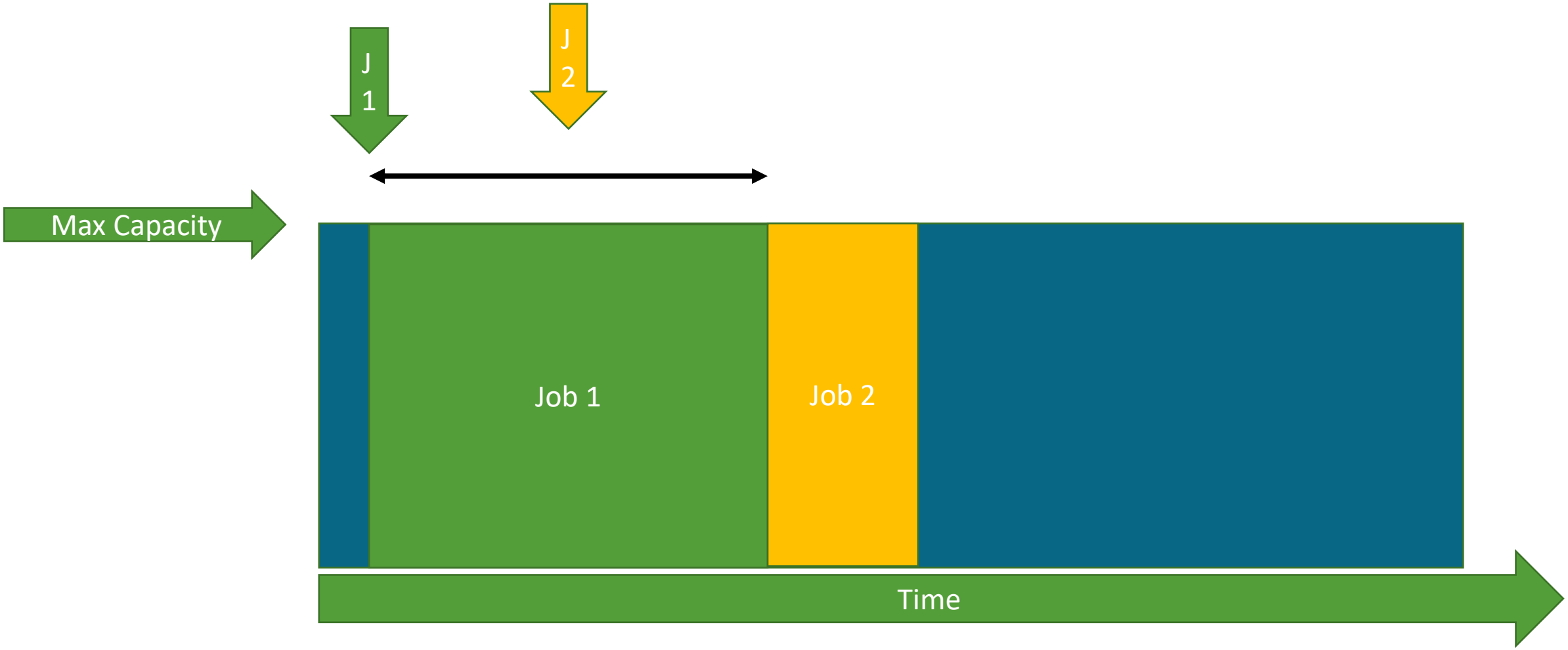
Schedulers

- FIFO Schedulers (Apache Hadoop)
- Capacity Schedulers (Hortonworks)
- Fair Share Schedulers (Cloudera Default)

FIFO Scheduler

- Simple and original scheduling algorithm from MRv1
- Initial Scheduler by Apache Hadoop
- Pulls the oldest jobs first from the queue
- Had no concept of the priority or size of the job
- Suitable for small size cluster

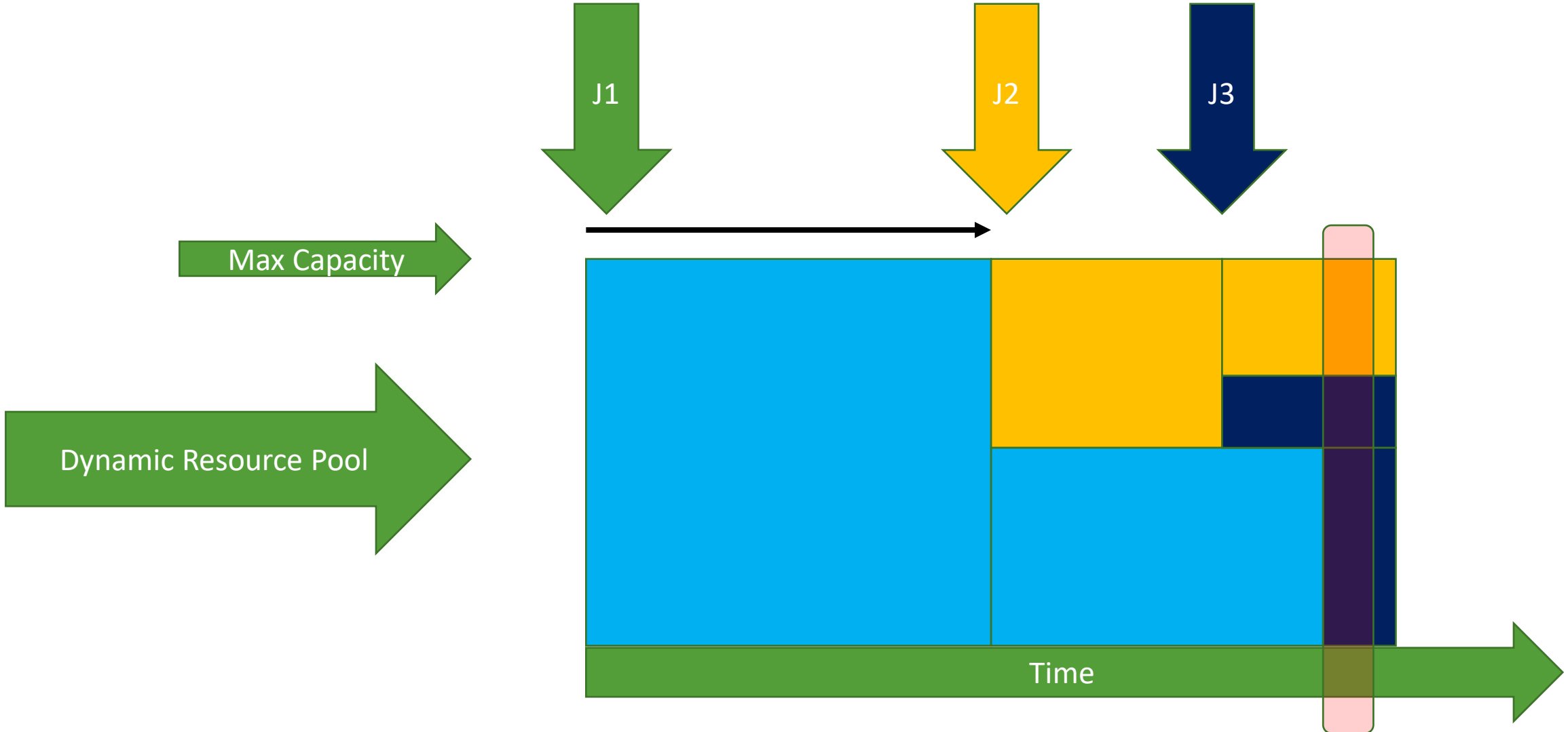
FIFO Scheduler



Fair Share Scheduler

- Jobs get average, equal share of resources over time.
- When single job running in entire cluster, it consumes all resources.
- Any new job(s) submitted, the free up container will allocated.
- Pools and Priority can be set to Jobs.
- If minimum share is not met for some period of time, the scheduler optionally supports *preemption* (Terminating any running jobs)
- Configurations in yarn-site.xml and fair-scheduler.xml

Fair Share



Yarn-site.xml – For every container request

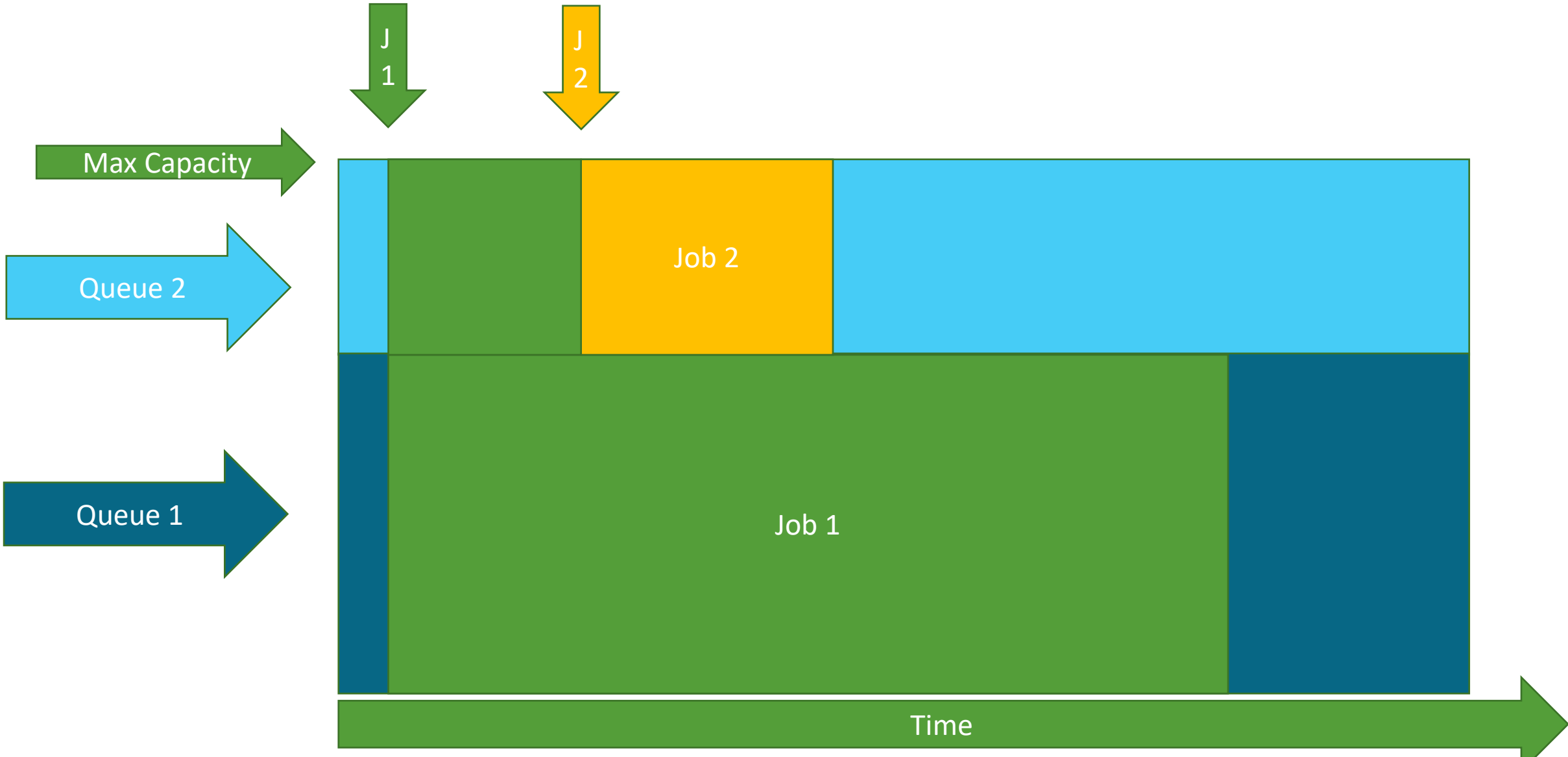
- Yarn.scheduler.minimum-allocation-mb - 1024
- Yarn.scheduler.maximum-allocation-mb - 8192
- Yarn.scheduler.minimum-allocation-vcores - 1
- Yarn.scheduler.maximum-allocation-vcores – 32

- yarn.scheduler.capacity.<queue-path>.maximum-capacity
- yarn.scheduler.capacity.<queue-path>.capacity

Capacity Scheduler

- Elasticity over queues to share capacity
- Access control
- Max capacity per queue to enforce capacity
- User limits within queue
- Queues with priority

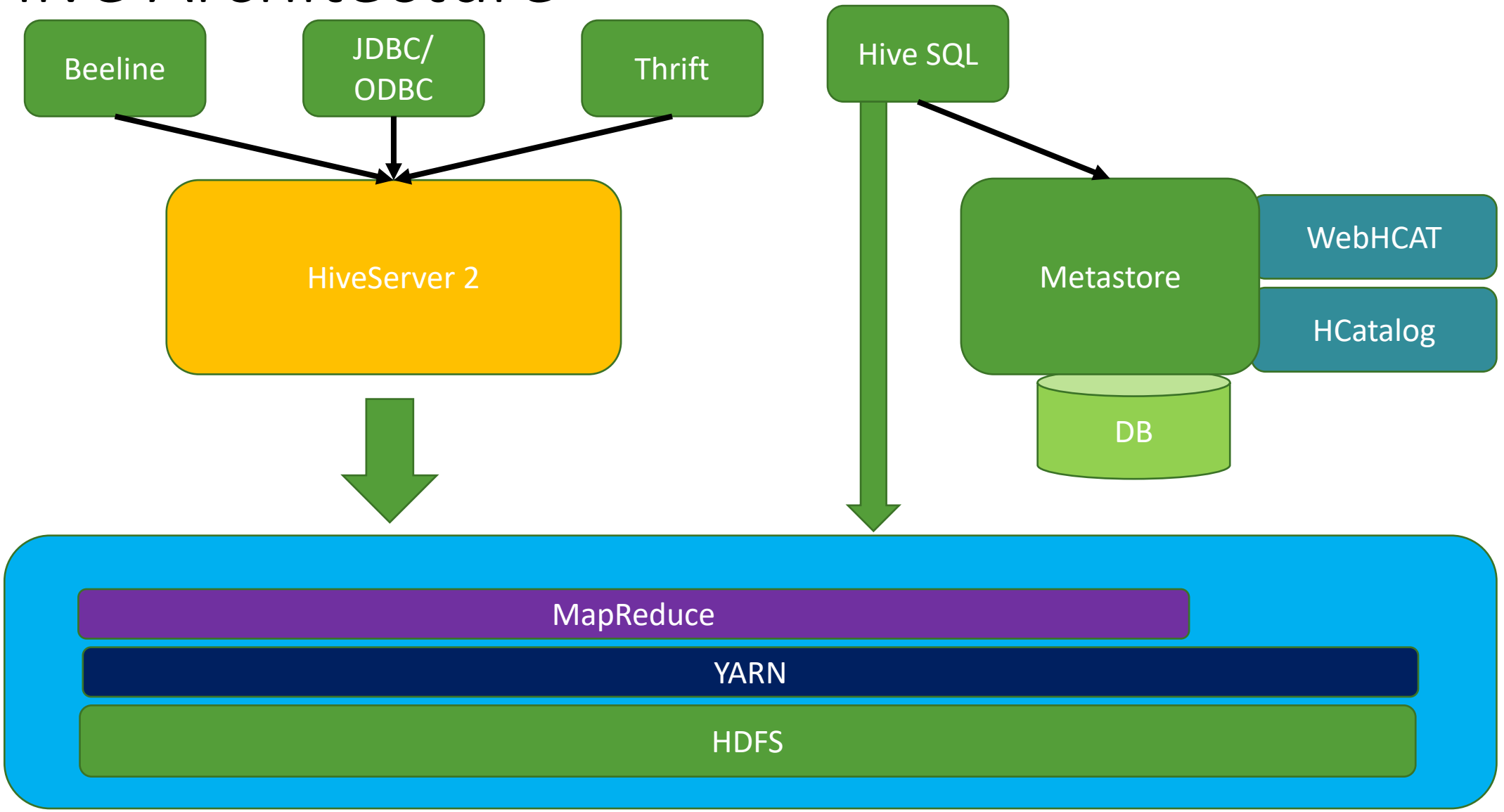
Capacity Scheduler



Hive

- Petabyte-scale data warehouse offering
- Enables reading, writing and managing large datasets in distributed environment
- Uses HiveQL, similar to SQL
- HiveQL transformed into sequence of MR jobs and executed on Hadoop Cluster through MapReduce and Spark
- Works with structured data only

Hive Architecture



Metastore database

- Holds the metadata about databases, tables, columns, partitions and Hadoop specific details like files and block location in HDFS
- Shared by other components like Impala, Pig, Hcatalog
- Runs on its own JVM
- Metadata gets stored in a dedicated RDBMS

HiveServer2

- Server interface which facilitates remote clients to submit queries
- Supports multi-client concurrency, capacity planning, security, authorization, etc.,
- Container for Hive execution engine.
- Supports JDBC clients such as Beeline CLI, ODBC clients.
- Client connection is established with Thrift API based Hive service.
- Can leverage and use Spark and HBase

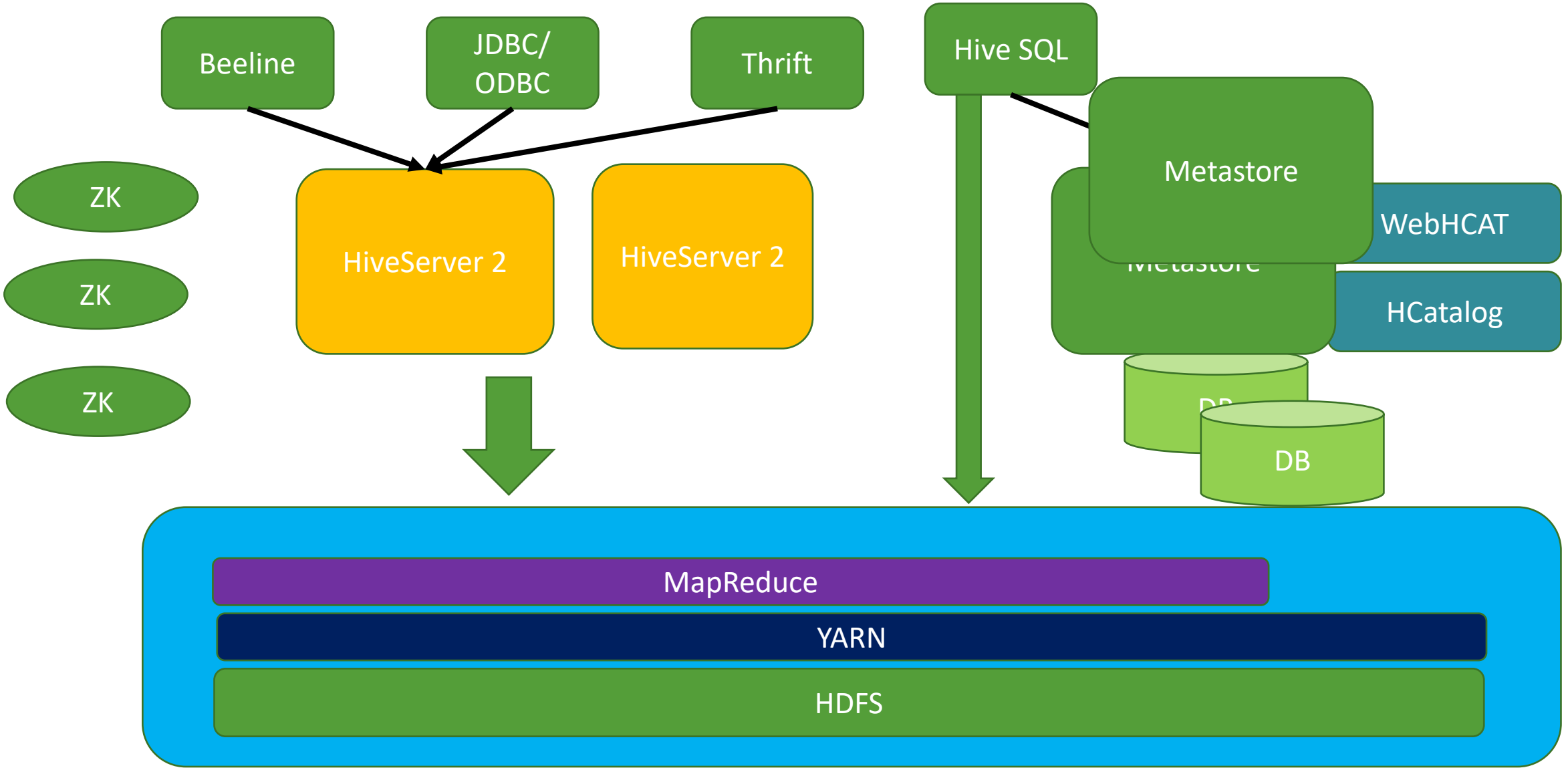
Hue (Hadoop User Experience)

- Open source web based interactive query editor
- Developed by Cloudera
- Browser, Editor, Workflow Editor, Shell, Solr Searches

Oozie

- Scheduler system for Hadoop Jobs from Apache
- Can create and manage collection dependency jobs as workflow
- Scheduler can trigger the workflow at regular interval
- Workflows are managed as DAG cycle

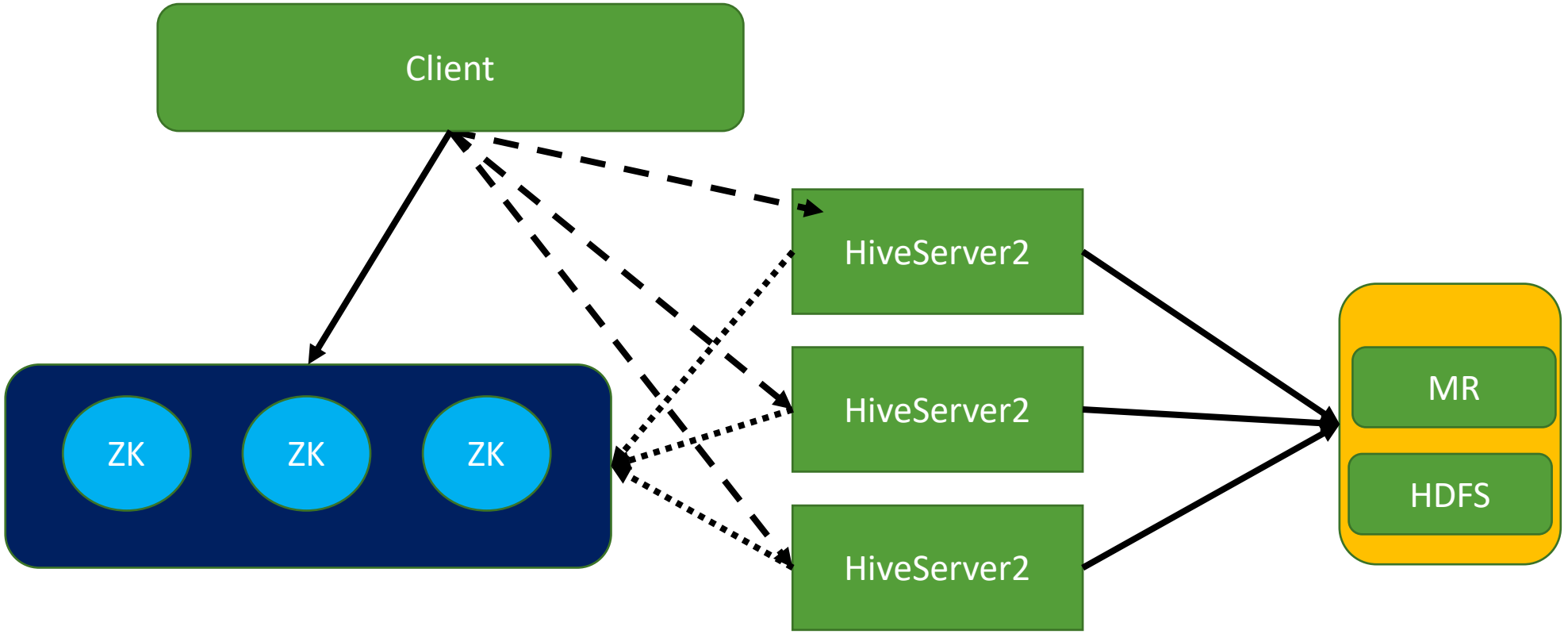
Hive High Availability



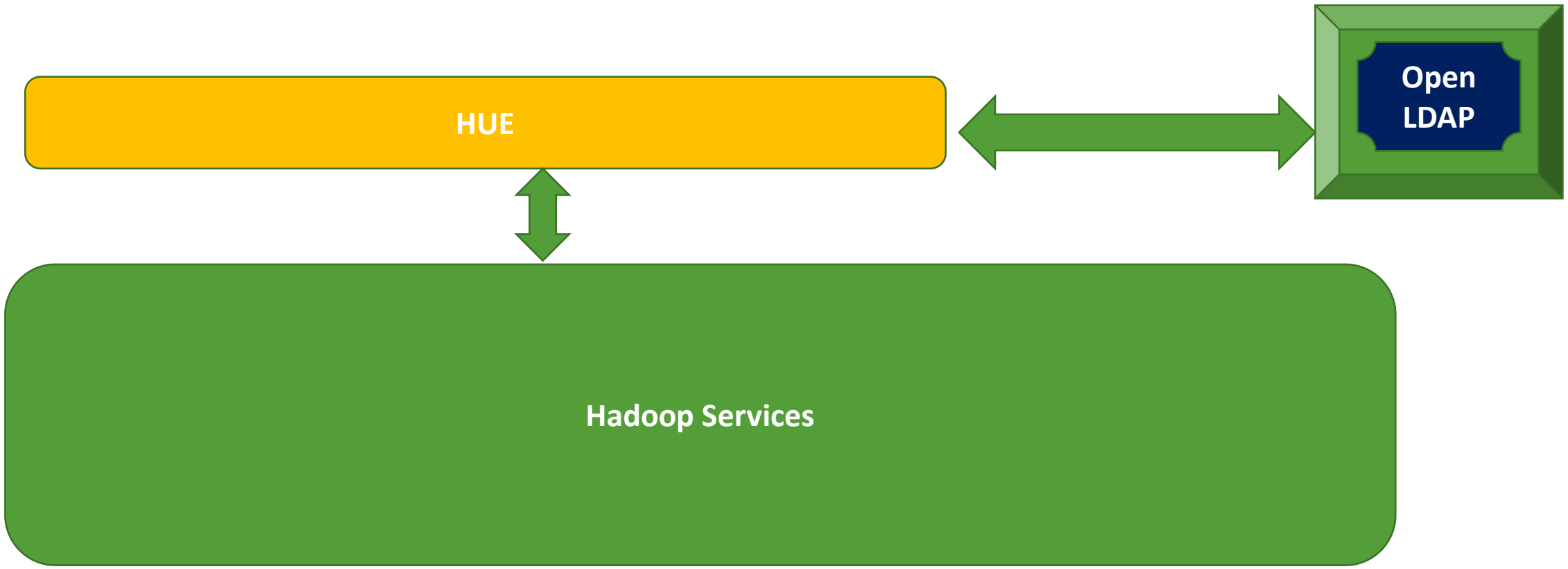
Point of Failure

- RDBMS holding the metastore
- **Hive Metaserver**
- **HiveServer2**
- Zookeeper to coordinate high availability

HiveServer2 HA



HUE Open LDAP Configuration



HUE Configuration Property

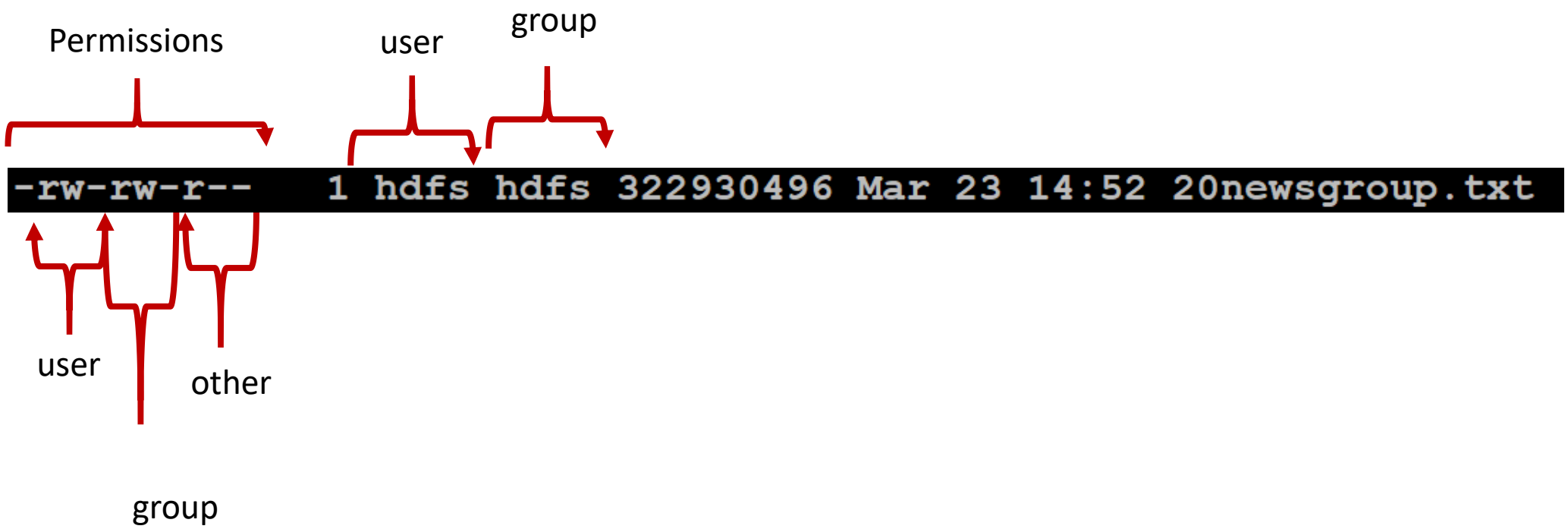
- Authentication Backend -> desktop.auth.backend.LdapBackend
- ldap_url -> ldap://<<hostname>>:389
- ldap_username_pattern -> uid=<username>,**ou=users,dc=muthu4all,dc=com**
- search_bind_authentication -> Select (True)
- use_start_tls -> True
- create_users_on_logon -> True
- base_dn -> **dc=muthu4all,dc=com**
- bind_dn -> **cn=admin,dc=muthu4all,dc=com**
- bind_password -> Provide as per LDAP configuration
- user_filter -> **objectClass=***
- user_name_attr -> **uid**
- group_filter -> **objectClass=posixGroup**

HDFS ACL

- Linux Permissions

- What if User1 and User2 needs to be owner of the file?

Linux Permissions

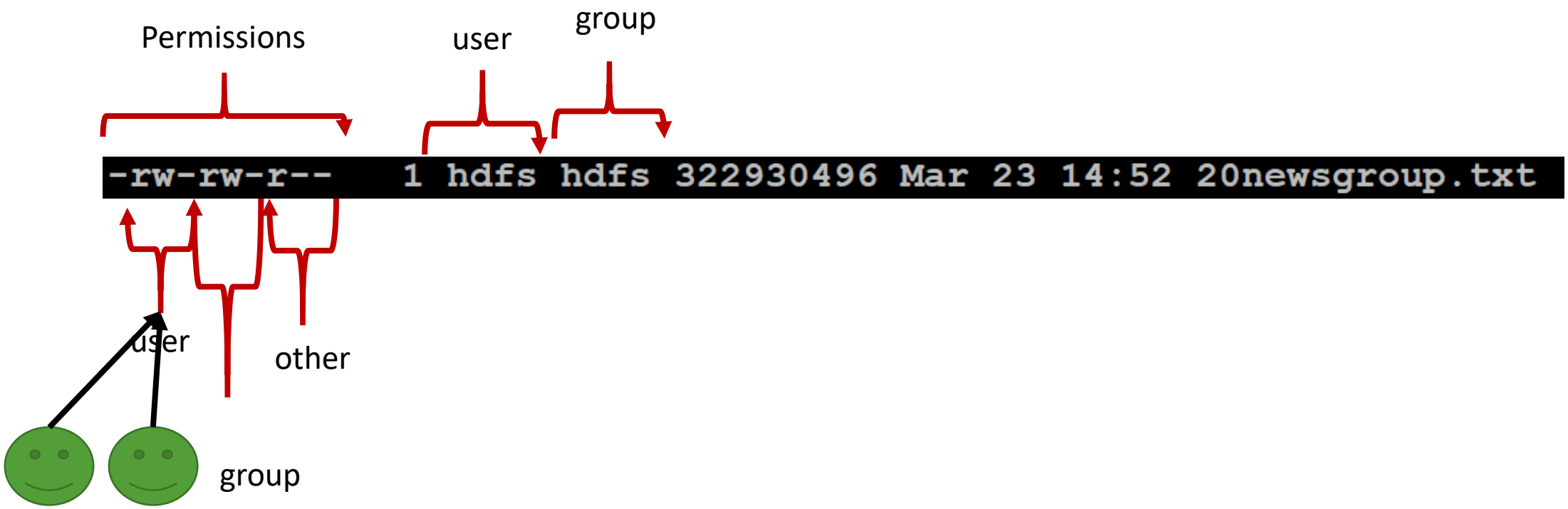


Scenario

- User Kumar should have read and write permission
- Kumar part of marketing team.
- All others in marketing team will have read permission.
- Others outside marketing team should not have neither read nor write.

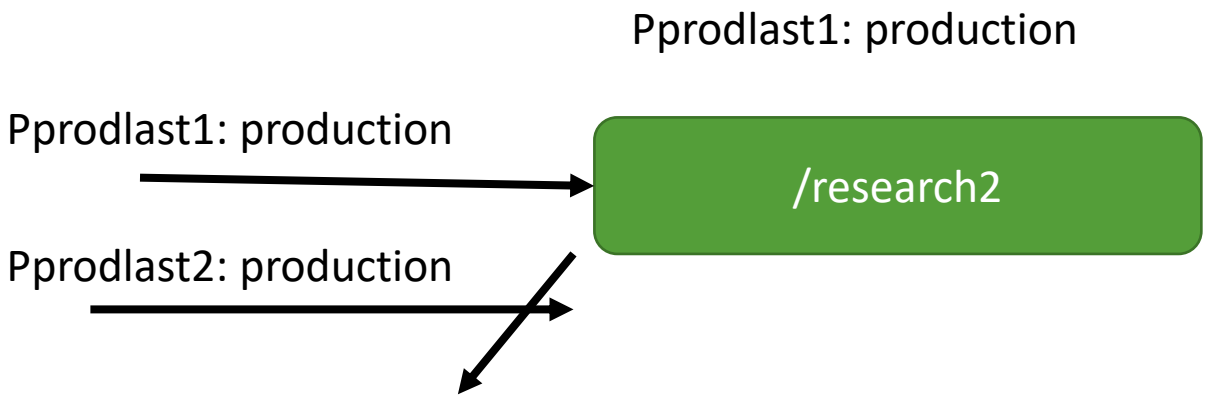
- RW_+R__+___ Owner will be Kumar and group will be marketing
- What if another user in marketing needs write permission?

Linux Permissions



Properties - ACL

- dfs.permissions.enabled = true
- dfs.permissions.superusergroup = supergroup
- dfs.namenode.acls.enabled = true



ACL Options

- -R: List ACLs recursively.
- -b: Revoke all permissions except the base ACLs for user, groups and others.
- -k: Remove the default ACL.
- -m: Add new permissions to the ACL.
- -x: Remove only the ACL specified.
- <acl_spec>: Comma-separated list of ACL permissions.
- --set: Completely replace the existing ACL. Previous ACL entries will no longer apply.

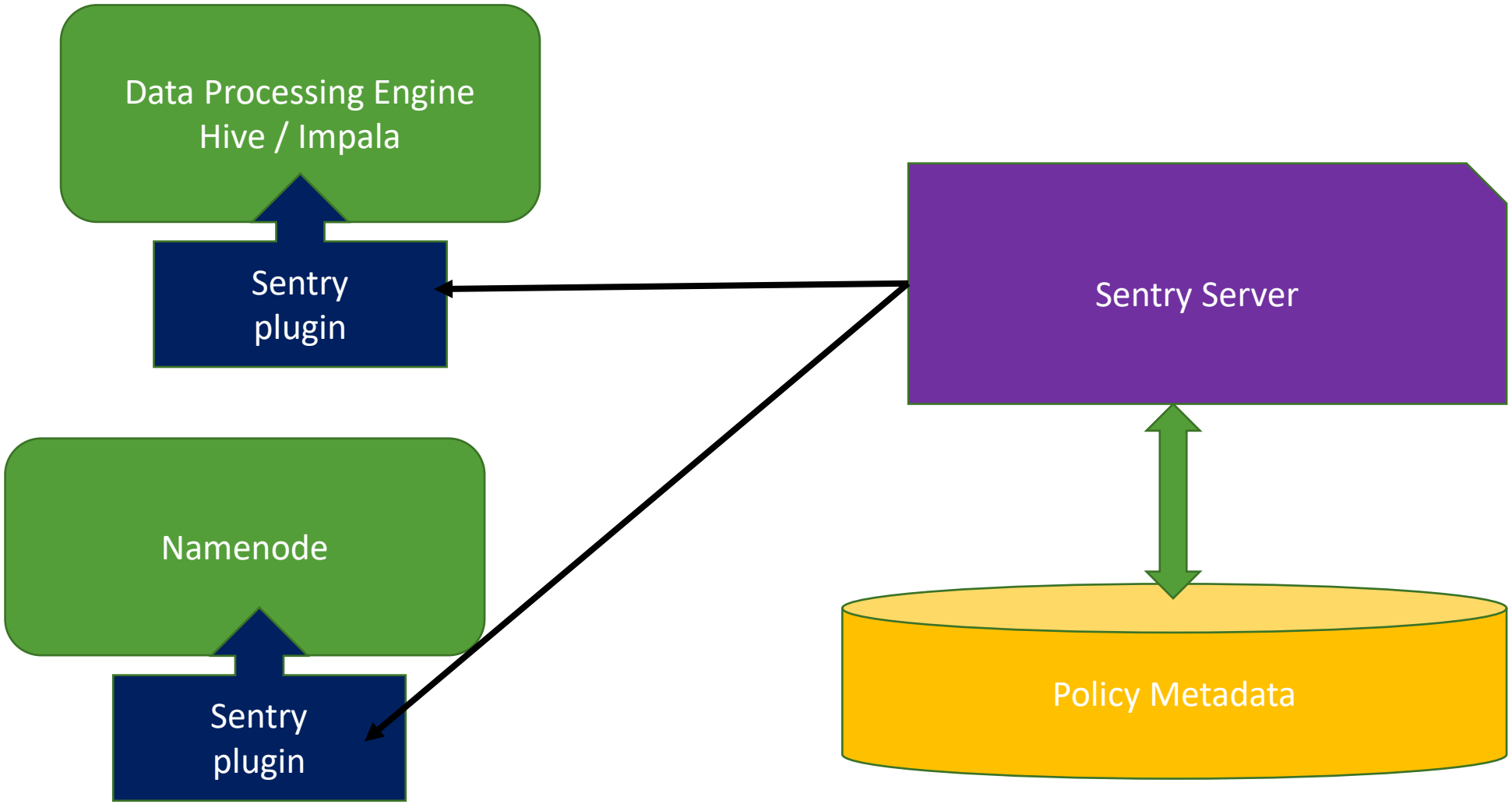
Order of evaluation of ACL Entries

- User is file owner – Owner permission bits are enforced
- Named user ACL entry.
- Member of file’s group
- Named group in ACL entry (Union of previous entry)
- If none then other permission bits are enforced

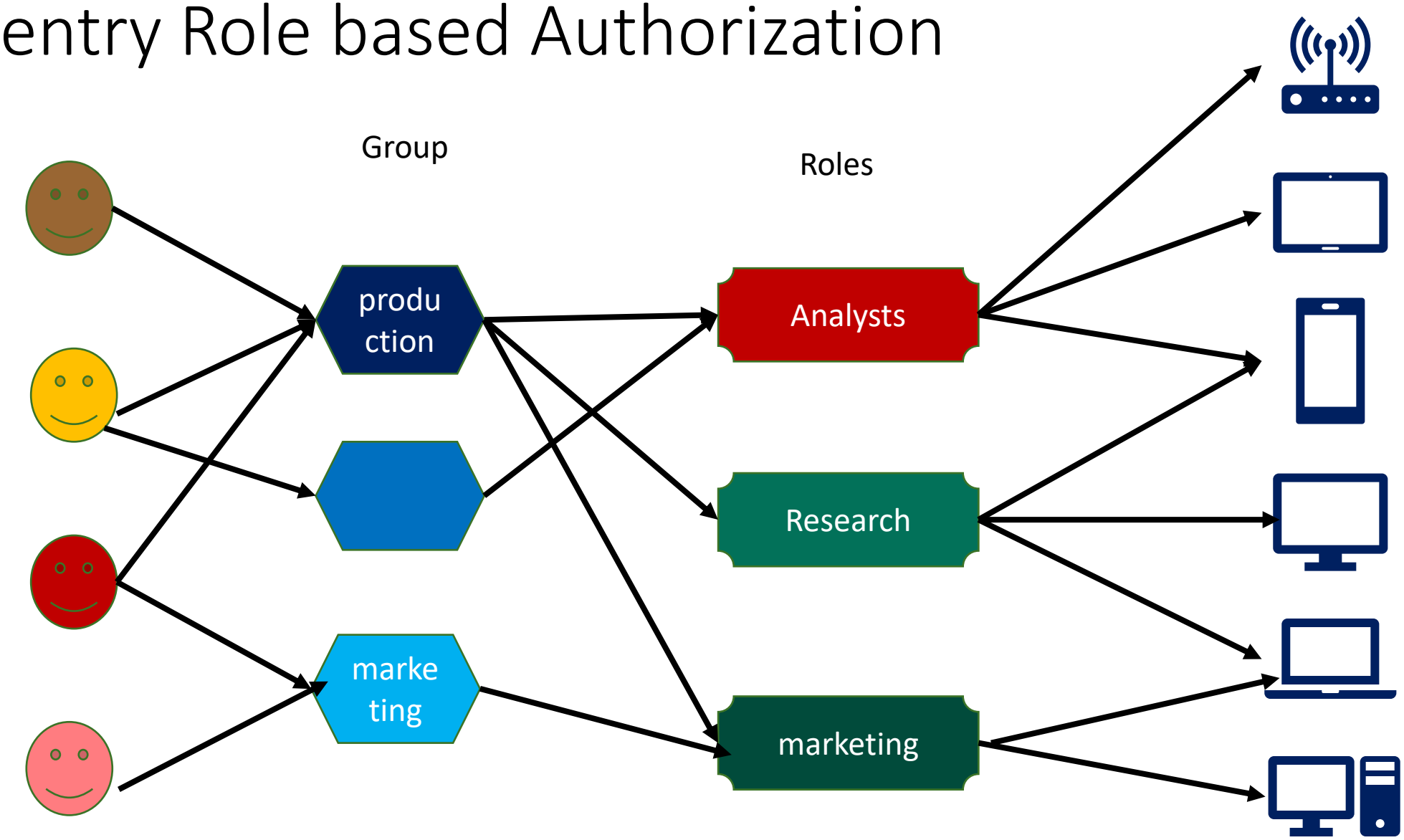
Apache Sentry

- Enforces find grained authorization
- Role based authorization to data and metadata
- Integrates with Hive and HDFS
- Supports Impala and many more components
- Developed by Cloudera
- Collection level, document level authorization could be provided

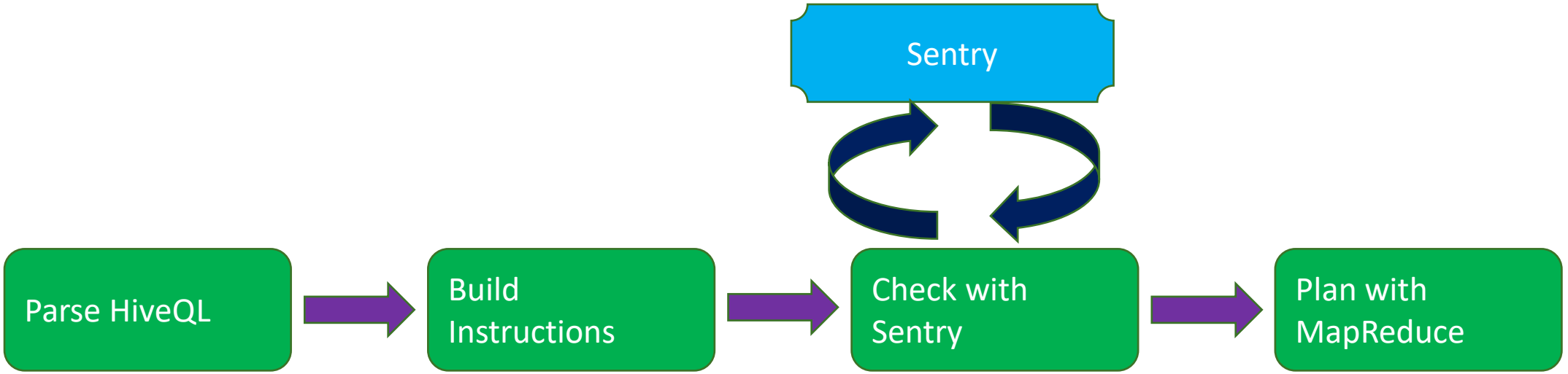
Sentry Components



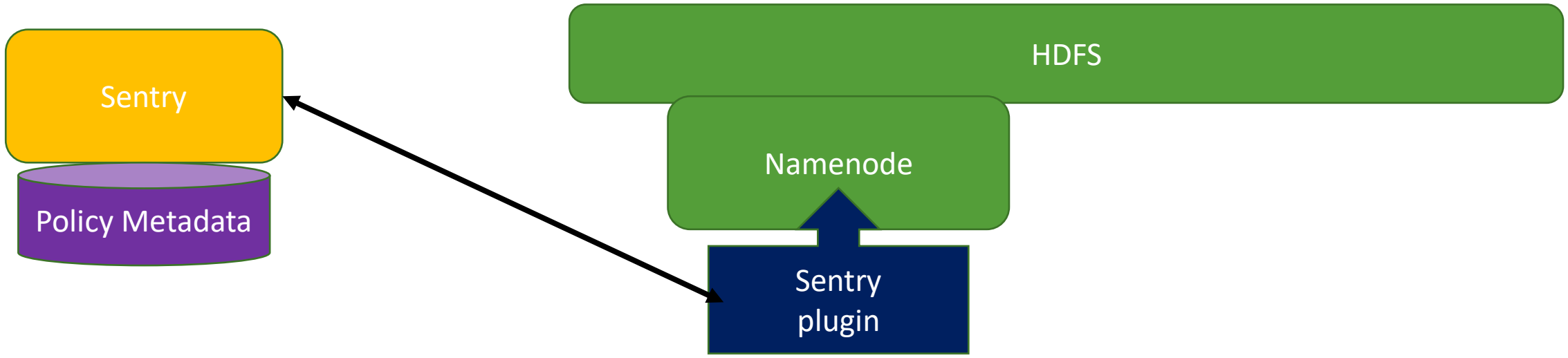
Sentry Role based Authorization



Hive and Sentry



HDFS and Sentry



Impala

- Distributed Massive Parallel Processing (MPP) database engine
- Doesn’t build on MapReduce
- Works with its own execution engine
- Native analytic database
- Developed by Cloudera

Impala Vs Hive

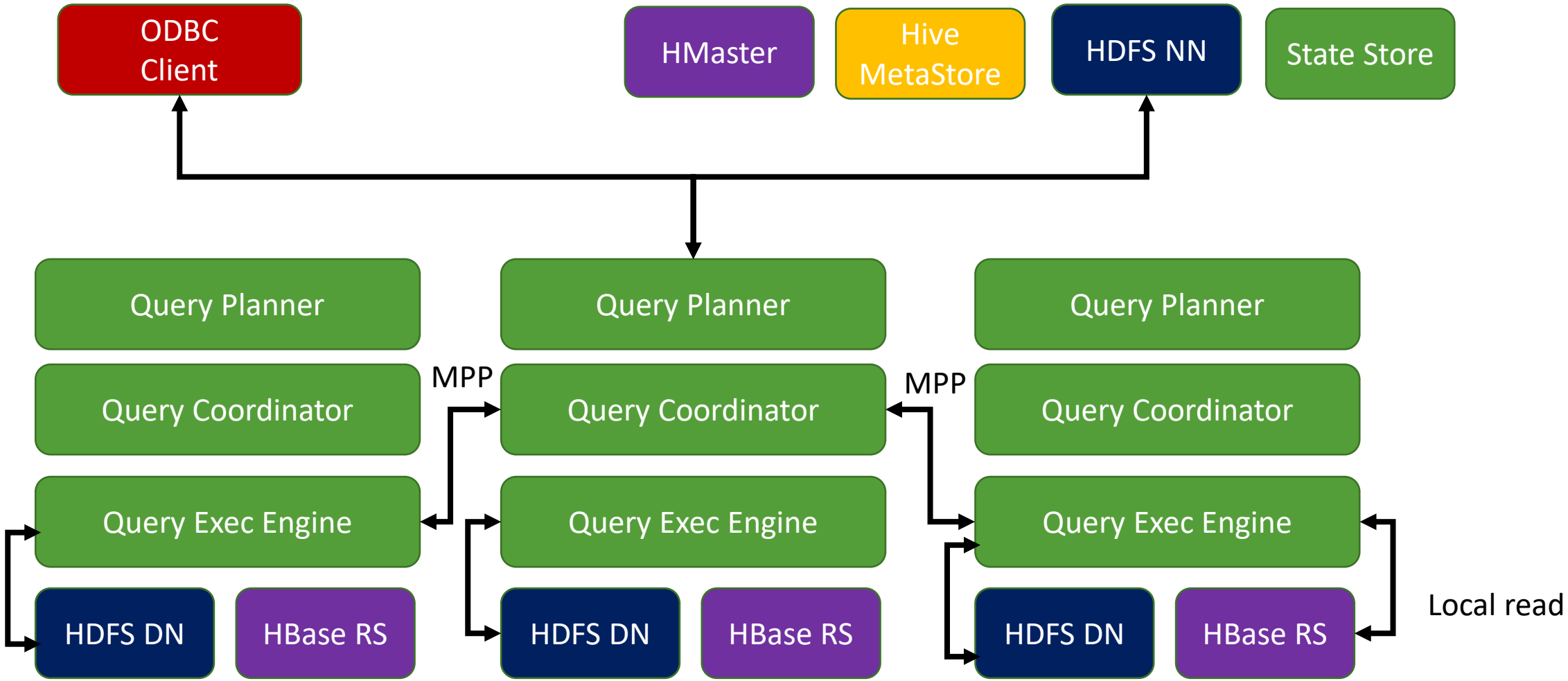
Impala

- Runtime code generation
- Daemons already running. Startup overhead is less
- Integrates with Sentry easily
- Latency is less
- Relies on Massive Parallel Processing

Hive

- Generates at compile time
- Cold start – On execution of query
- Relies on Hive metastore for security
- Latency is high
- Relies on MapReduce

Impala Architecture



Impala Introduction

- Unified Storage, Metastore, Security, Client Interface
- Real time SQL queries, natively distributed query engine

Impala Roles

- Impala Daemon (impalad)
  - Runs on each Datanode
  - Reads and writes to data files
  - Accepts queries transmitted from the shell
- Impala StateStore (statestored)
  - Need only one and not mandatory
  - Required when any impala daemon goes down to resume its functions.
  - Main function is to provide high availability and load balancing of impalad
- Impala Catalog Server (catalogd)
  - Mostly runs along with statestored
  - Avoids the need to refresh or invalidate the metadata when it gets changed

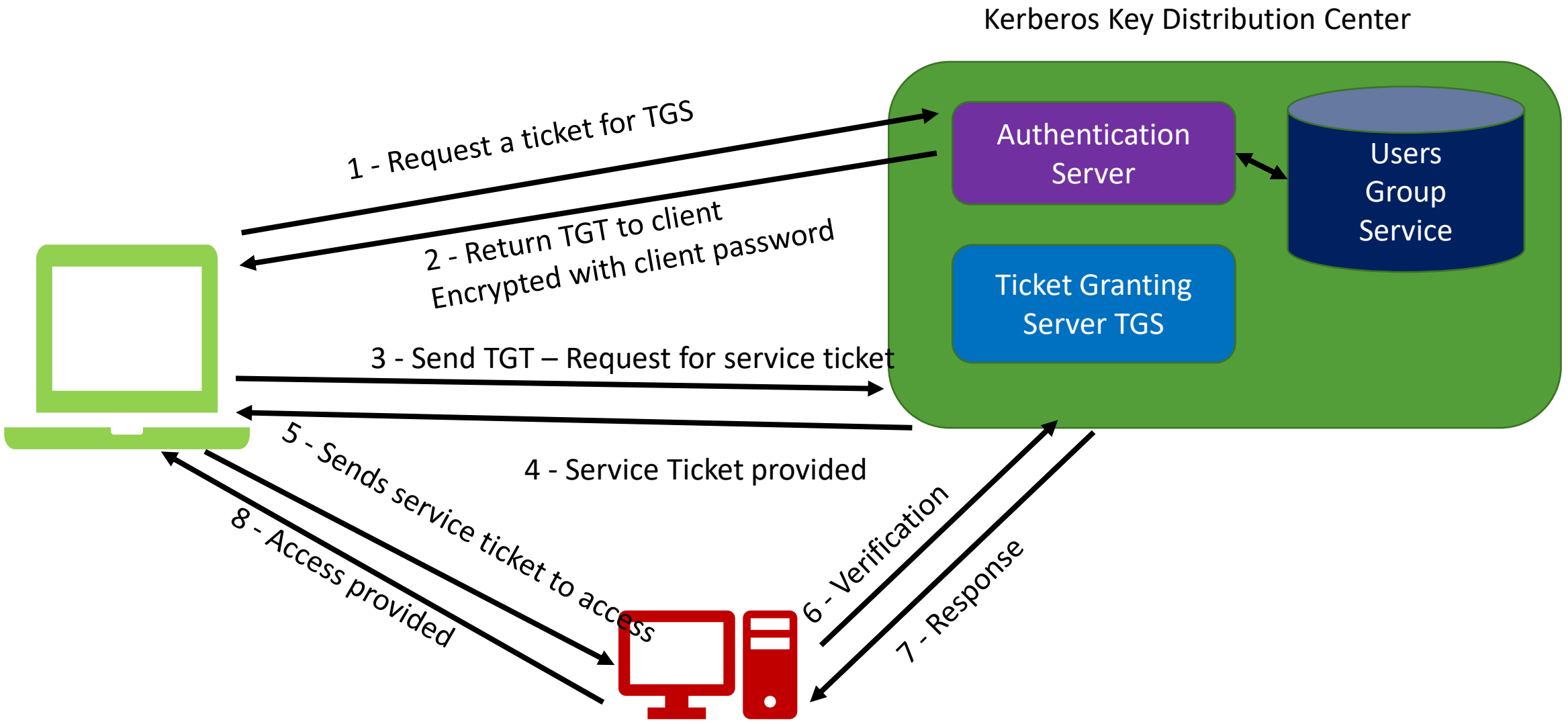
Cloudera Manager – OpenLDAP Integration

- Authentication Backend Order → External Then Database
- LDAP url ldap://<OpenLDAP hostname>:389
- LDAP bind user dn → cn=admin,dc=muthu4all,dc=com
- ldap bind password
- ldap user search filter → (uid={0})
- ldap user search base → ou=clouderausers, dc=muthu4all,dc=com
- ldap group search filter → objectClass=posixGroup
- group search base→ ou=clouderausers,dc=muthu4all,dc=com
- ldap distinguished name pattern → uid={0},ou=clouderausers,dc=muthu4all,dc=com

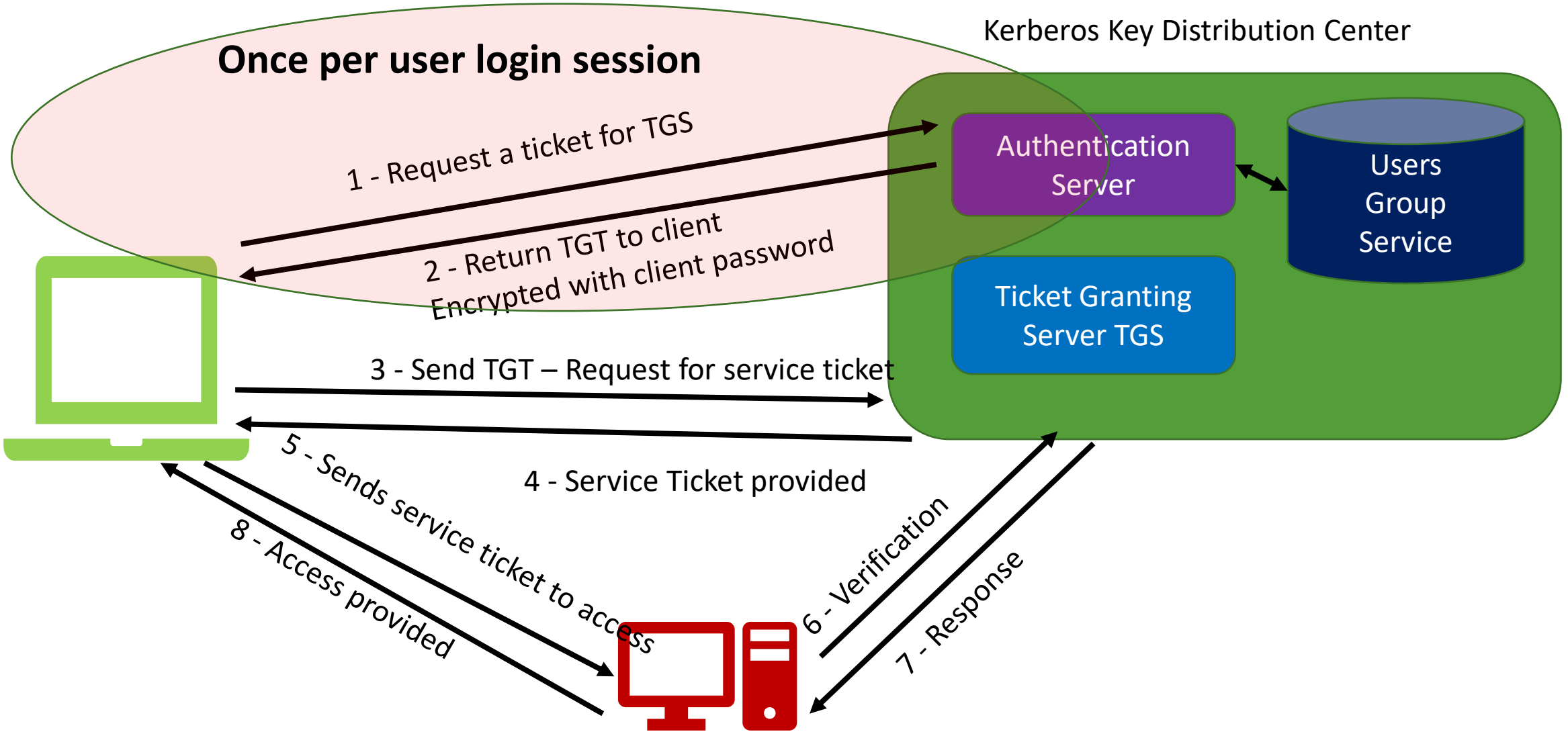
Kerberos Introduction

- Network authentication protocol
- Works based on tickets and key exchange mechanism
- Provides strong security on a non secure network
- Developed by MIT
- Only provides protocol.
- Should be integrated with external LDAP Servers like AD or OpenLDAP

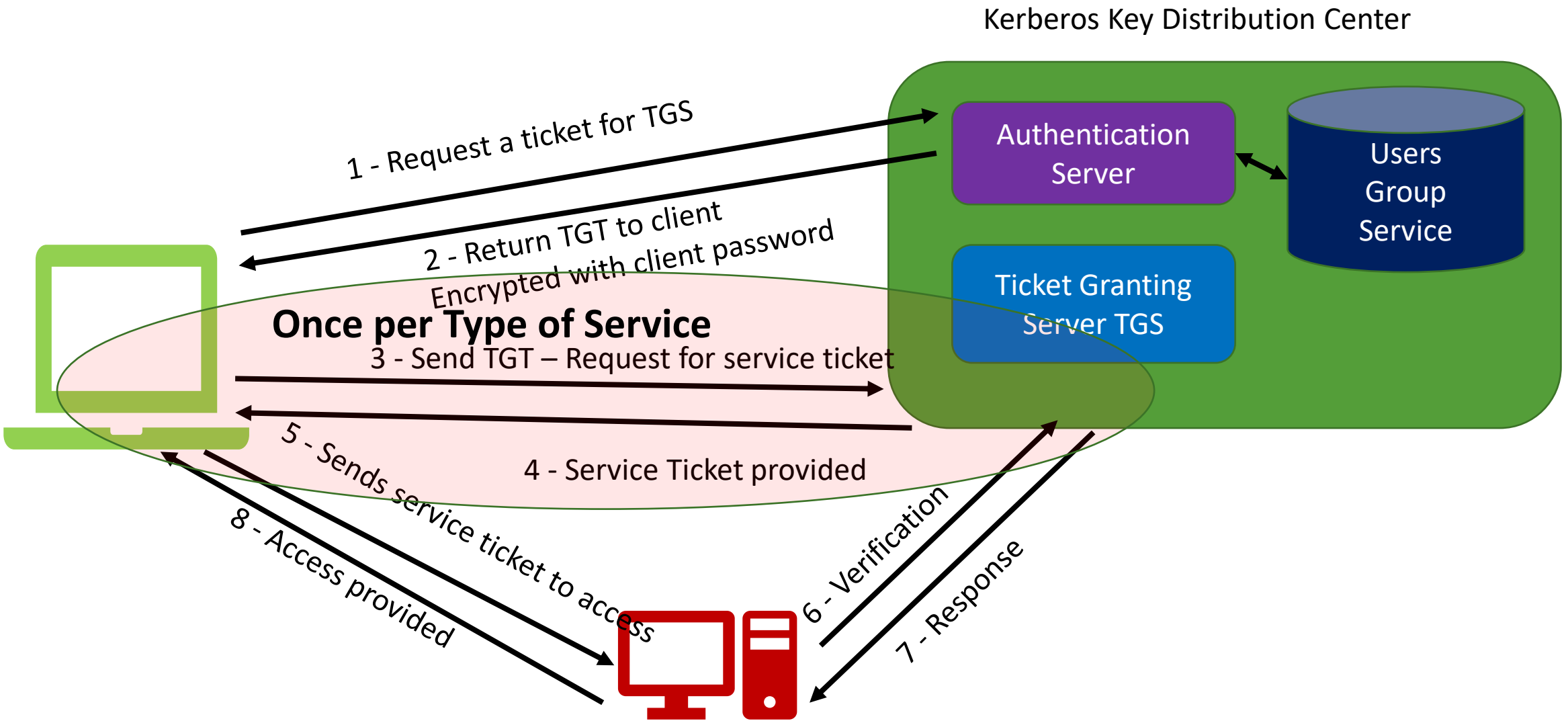
Kerberos Architecture



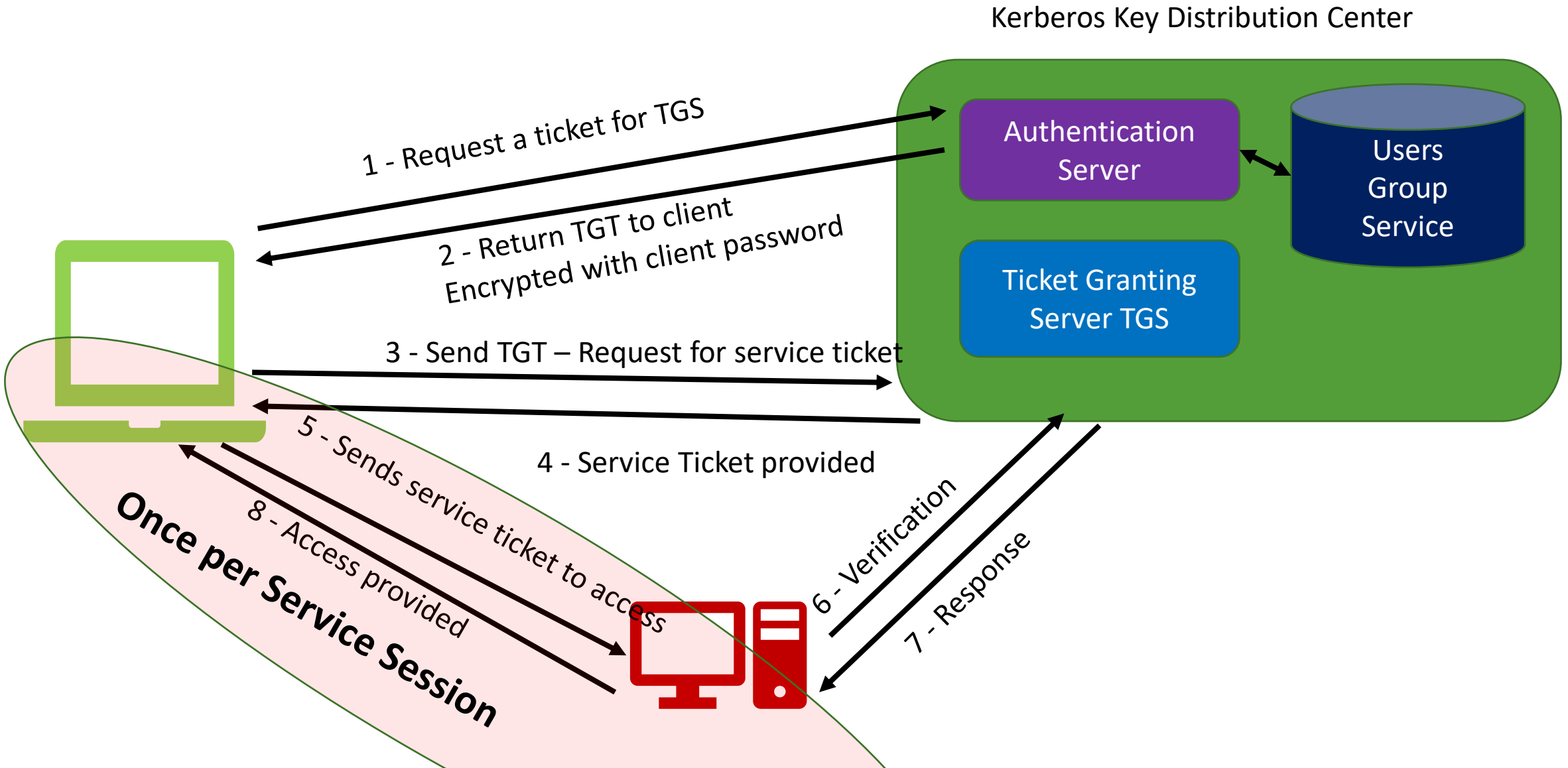
Kerberos Architecture



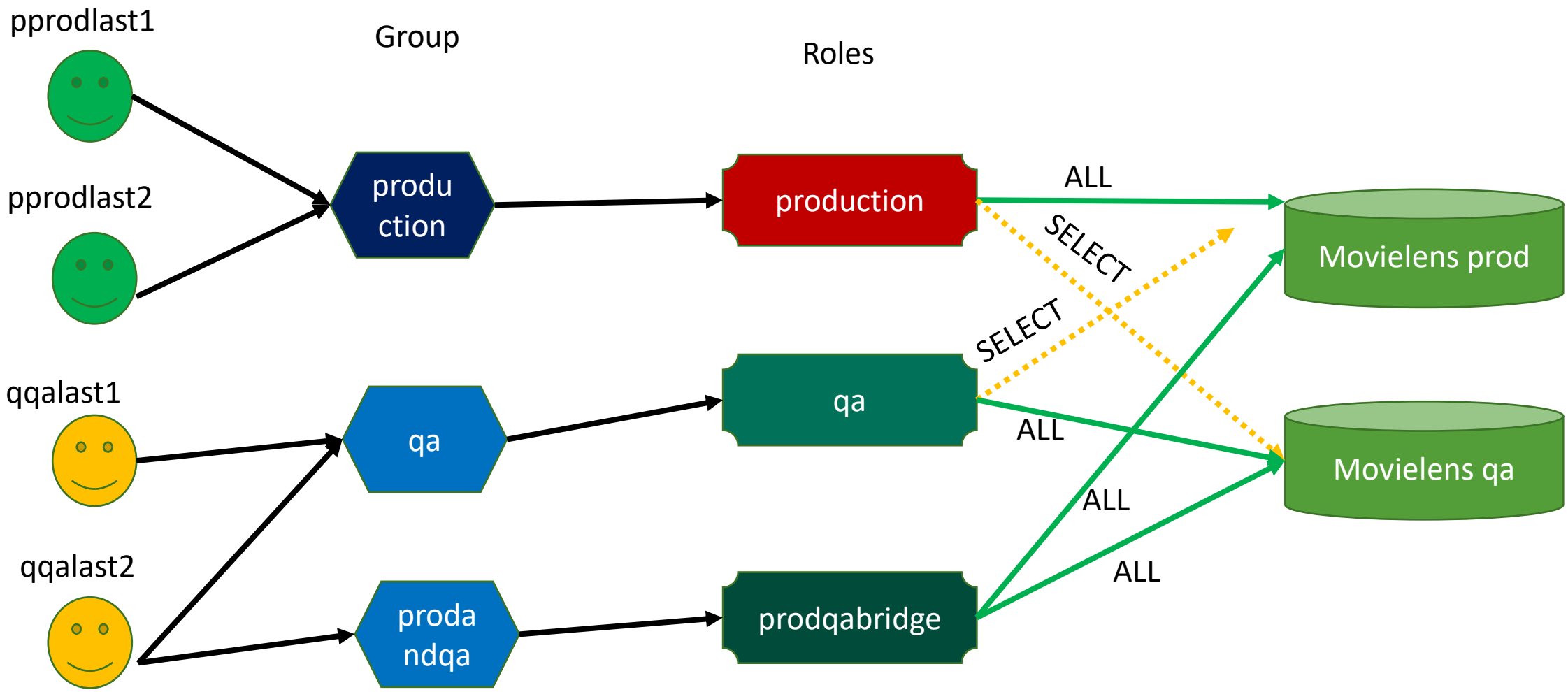
Kerberos Architecture



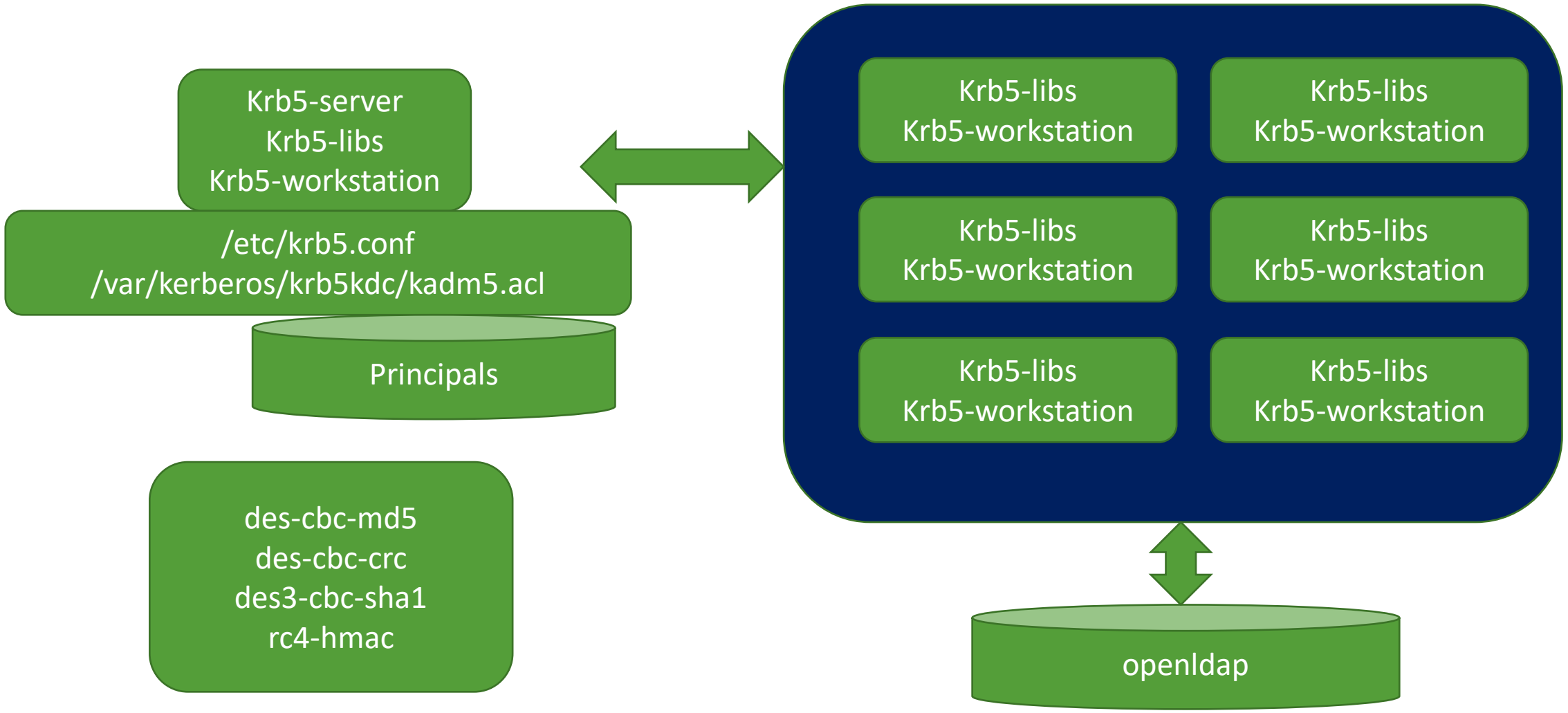
Kerberos Architecture



Sentry Role based Authorization



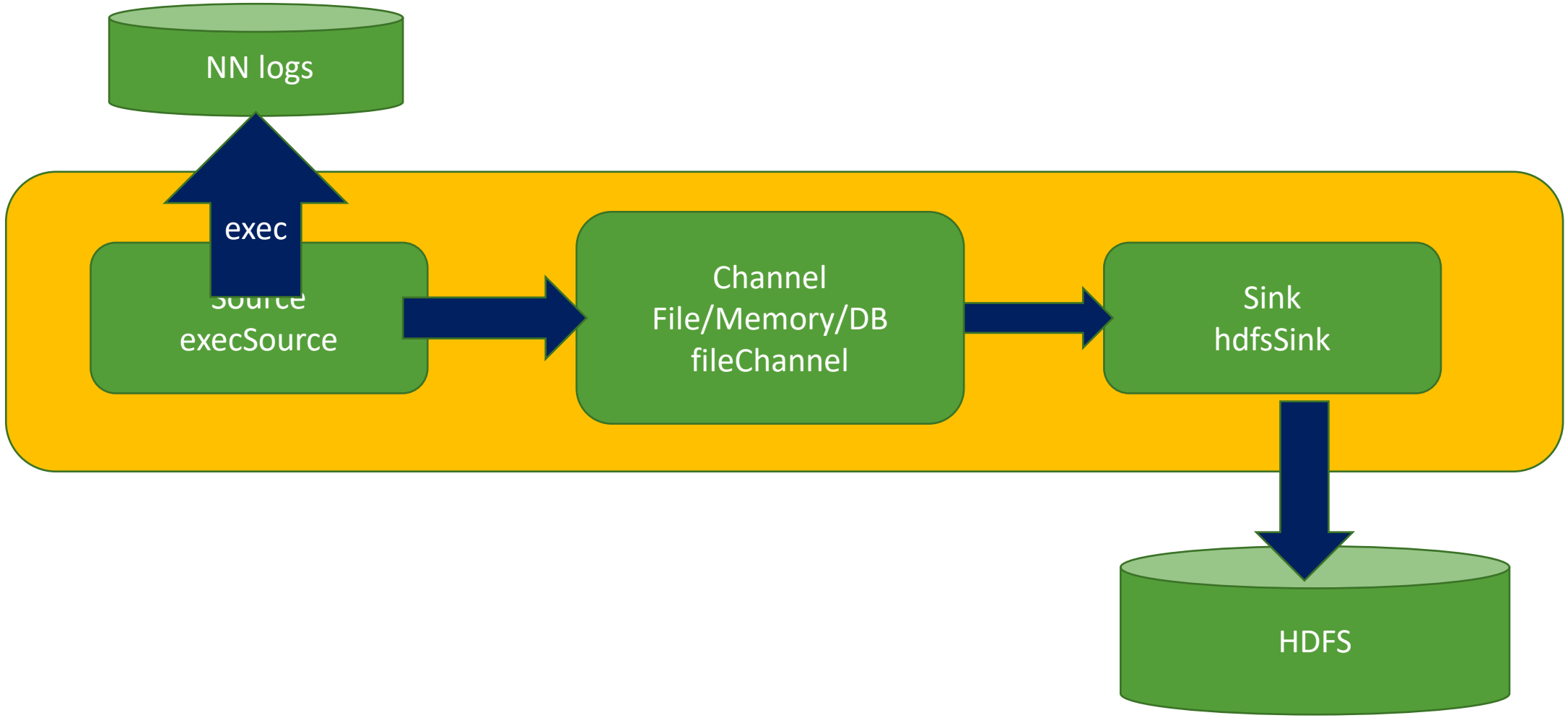
Kerberize Hadoop Cluster



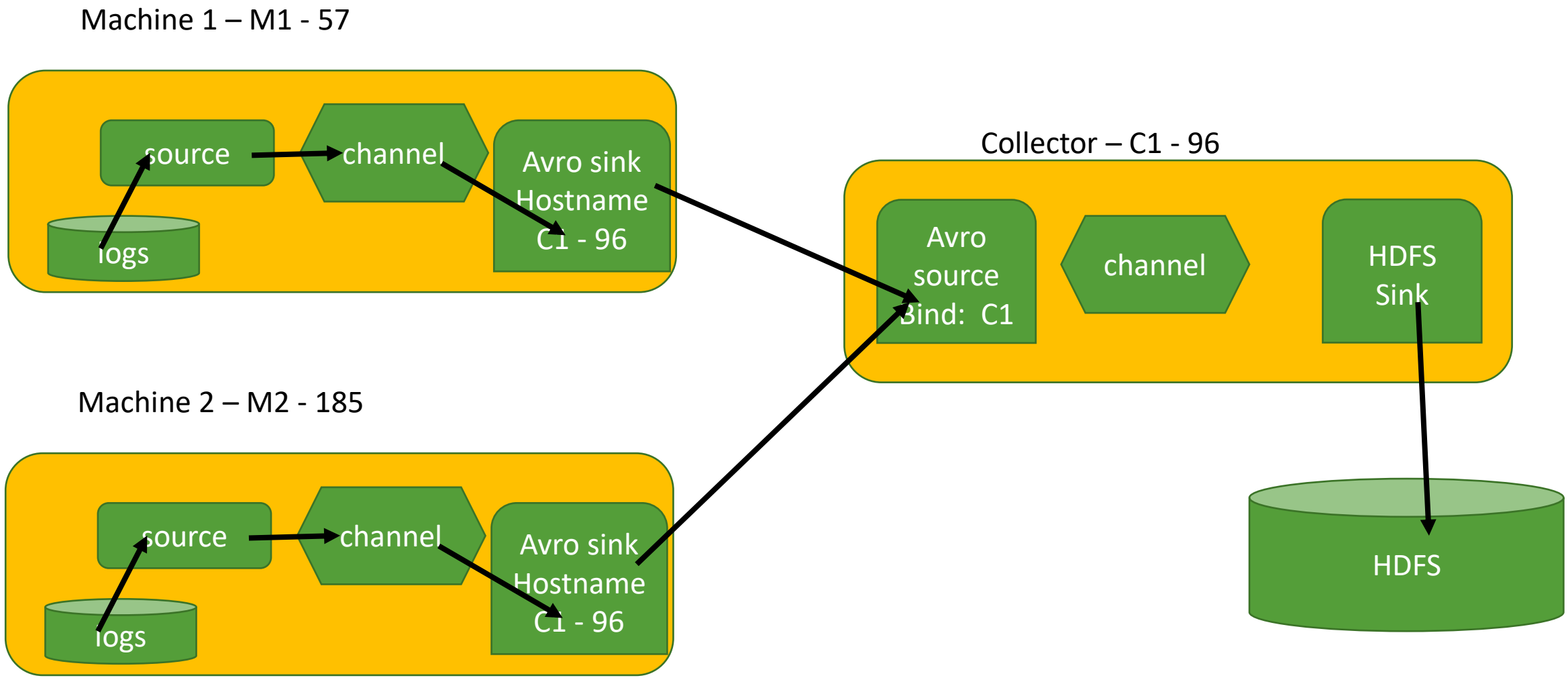
Flume Introduction

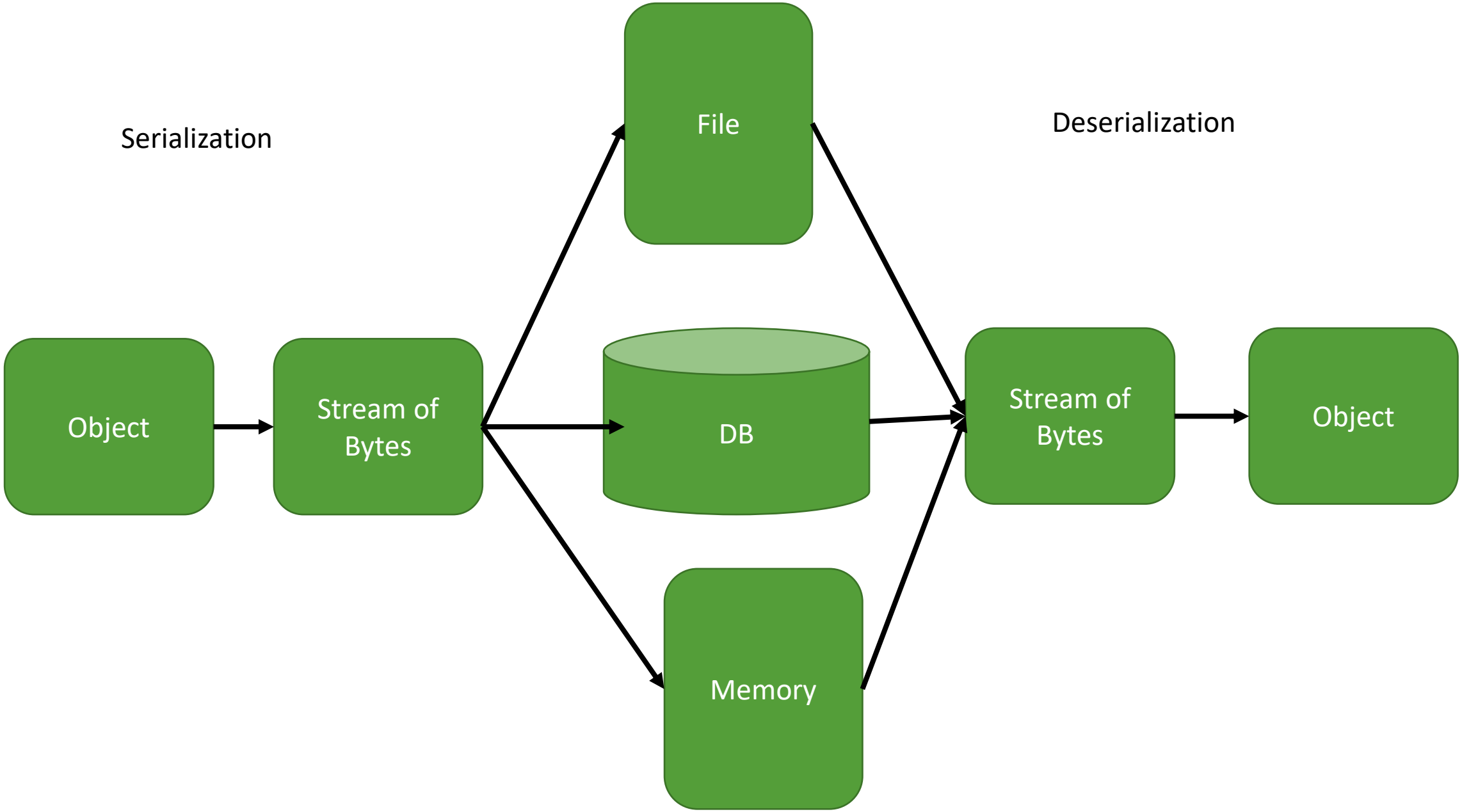
- Data ingestion tool for collecting/aggregating/transporting data
- Works with streaming data as logs/Facebook events/ twitter tweets
- Aggregates/sinks data to a single location
- Reliable, Fault tolerant, Scalable, Manageable

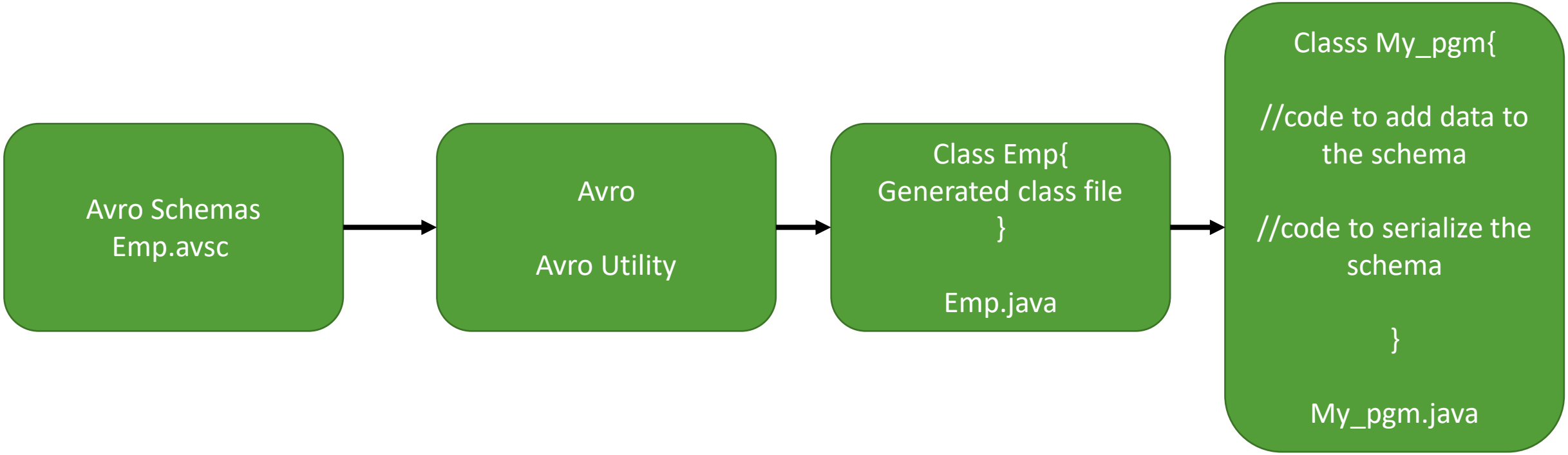
Flume


NN logs
exec
execSource
Channel
File/Memory/DB
fileChannel
Sink
hdfsSink
HDFS

Flume Multi Source



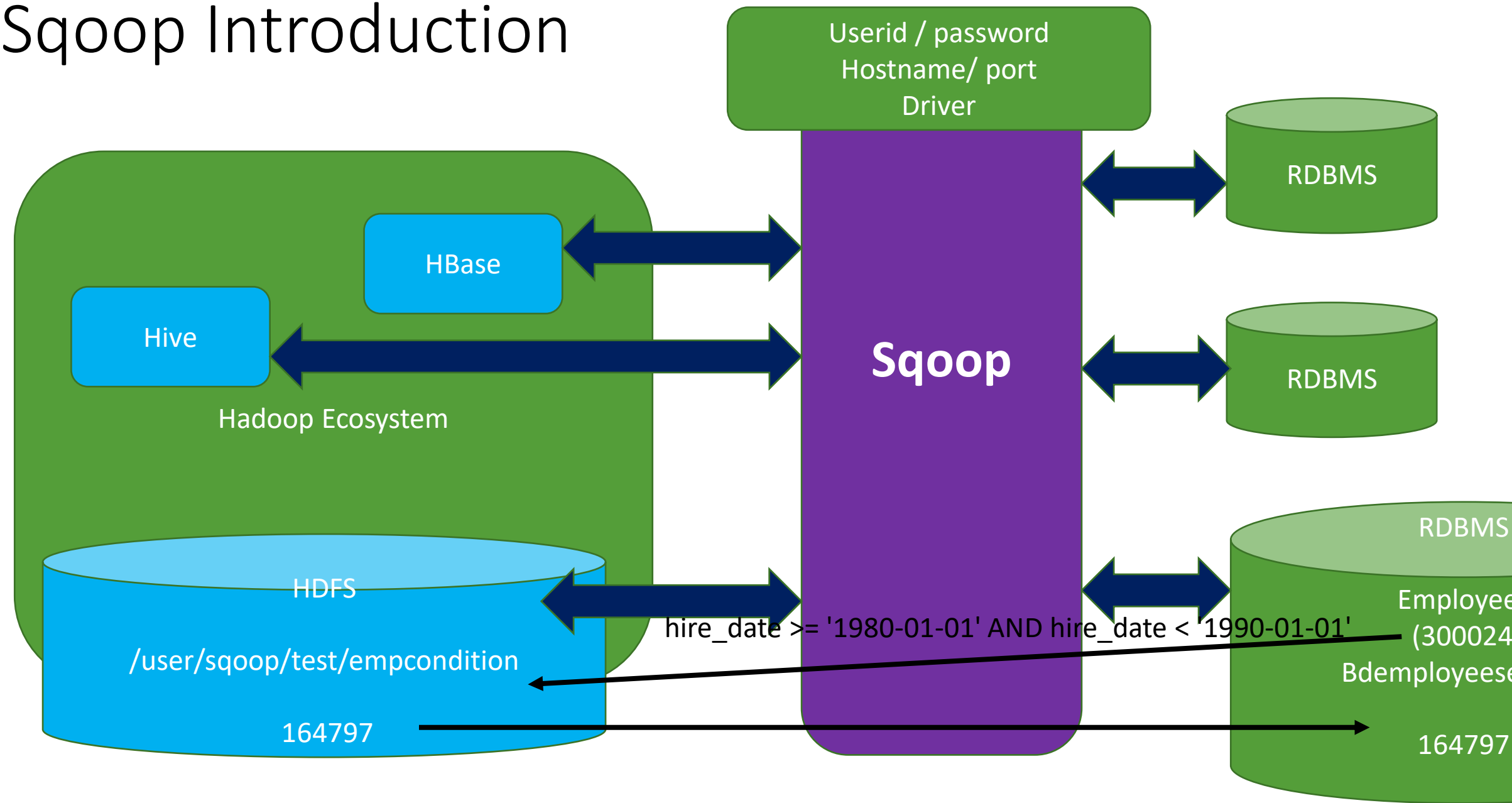

Serialization
Deserialization
File
Object
Stream of Bytes
DB
Stream of Bytes
Object
Memory


Avro Schemas
Emp.avsc
Avro
Avro Utility
Class Emp{
Generated class file
}
Emp.java
Classs My_pgm{
//code to add data to the schema
//code to serialize the schema
}
My_pgm.java

Sqoop2

- Works on client server model
- Connectors for major RDBMS not yet available
- Data transfer from RDBMS to Hive and Hbase not supported
- Data transfer from Hive and Hbase to RDBMS not supported
- Not yet integrated with HUE

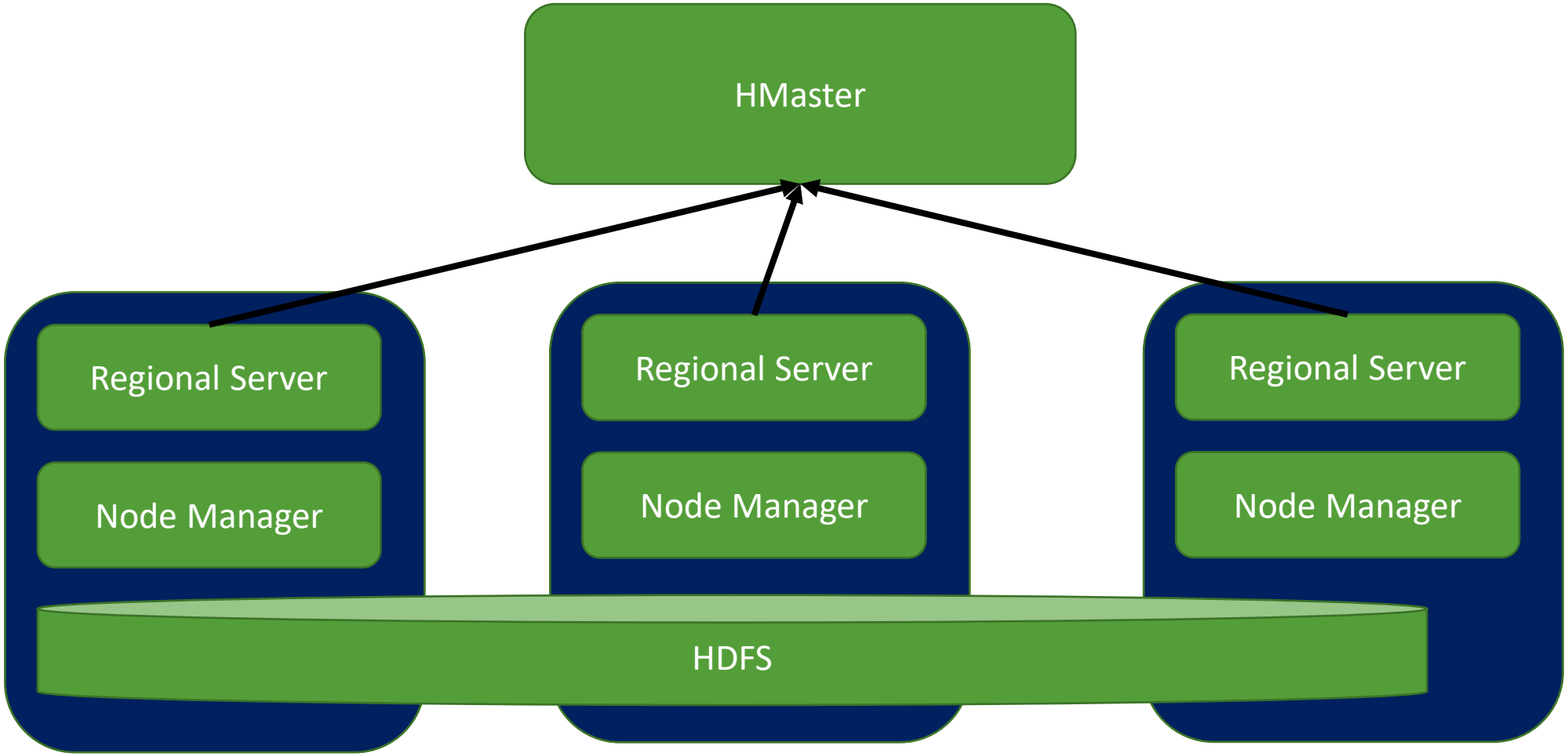
Sqoop Introduction



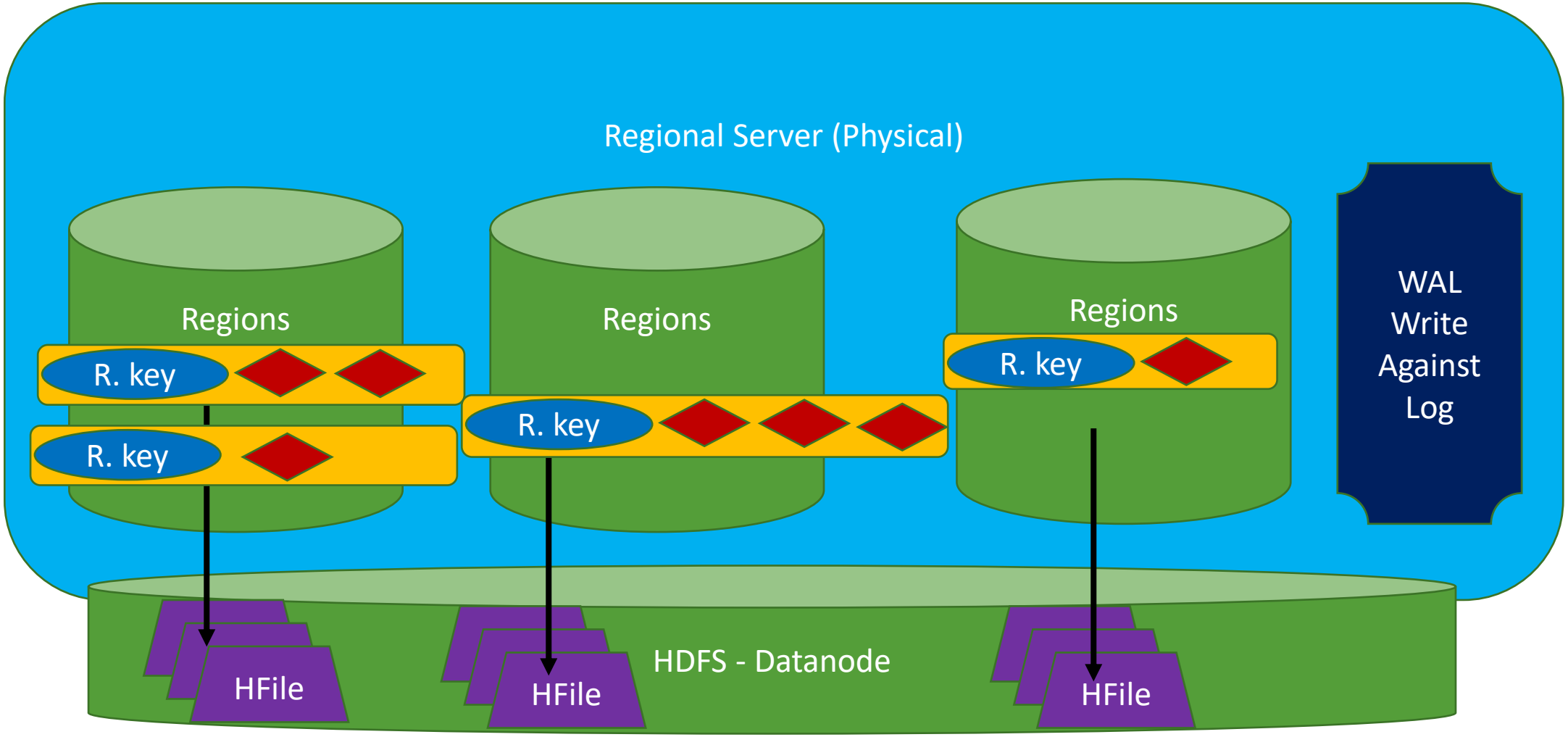
HBase

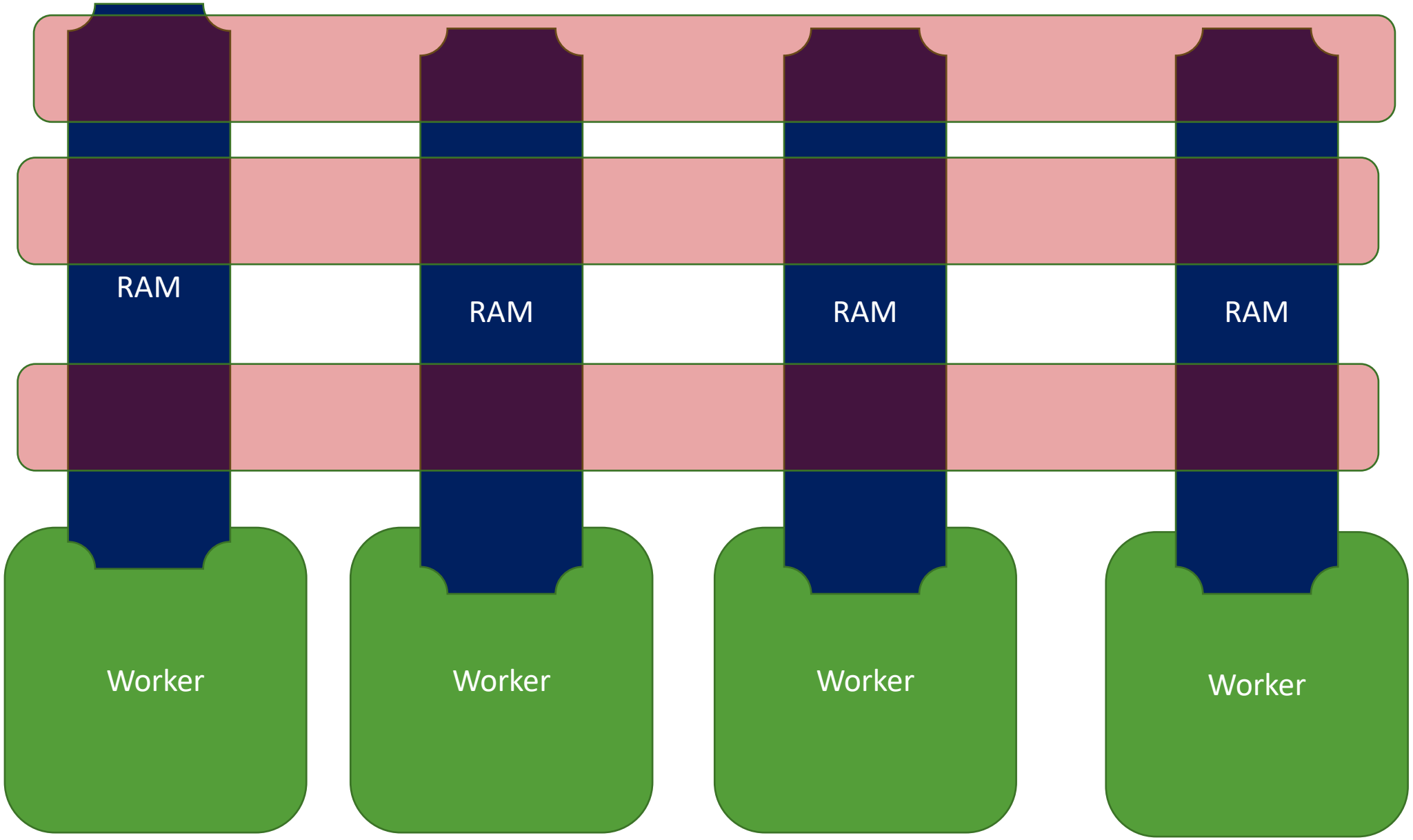
- Distributed Column oriented NoSQL Database
- Works on top of HDFS
- Similar to Google’s big table
- Data exists as key value pair.
- Group of key value pair together exists as column family
- N number of key value pair makes a row.
- Rowid, column family, key makes unique combination of any cell
- Works on master worker architecture

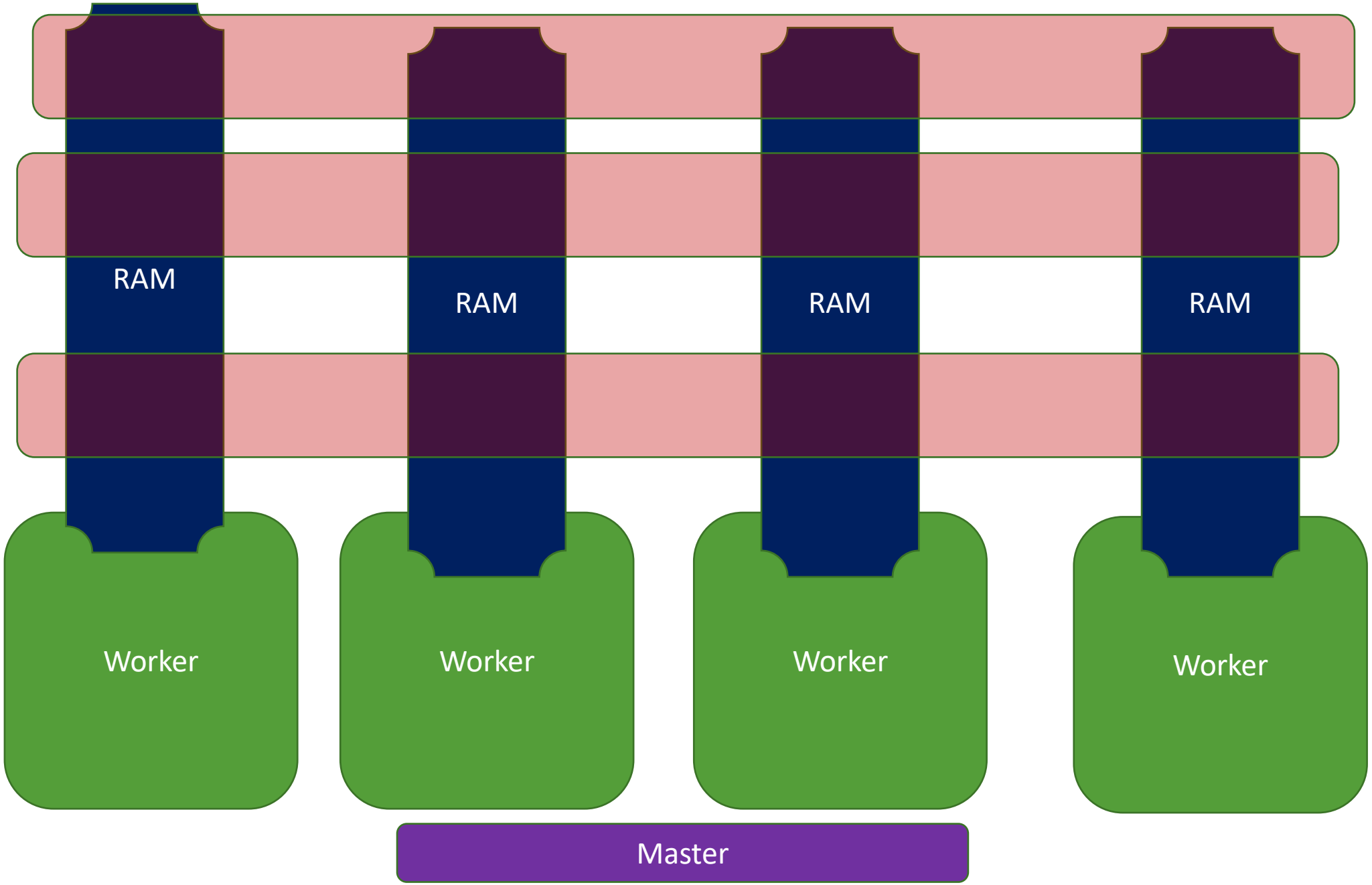
Hbase Architecture



Regional Server




RAM
RAM
RAM
RAM
Worker
Worker
Worker
Worker


RAM
RAM
RAM
RAM
Worker
Worker
Worker
Worker
Master