- ZooNavigator: UI to navigate Zookeeper Data

- Kafka Manager: UI to explore Kafka Metadata and perform basic administration tasks

- Kafka Monitor: UI that runs a producer/consumer and ensures Kafka is available and working

Zero Copy

- Zero Copy is a Linux optimization, which allows to avoid context switches and copying data around for applications that transfer a lot of data
- Overall, it's complex to explain, and there are two good blogs that explain it:
  - http://searene.me/2017/07/09/Why-is-Kafka-so-fast/
  - https://www.ibm.com/developerworks/library/j-zerocopy/index.html
- Kafka has great performance, mostly thanks to zero copy.
- **Conceptually**, it allows producers to write directly to the Kafka disk and consumers to read directly from the disk.

Zero Copy

- Overall, the zero copy optimization is lost in the following cases:
  - SSL is enabled: that means you need to assign more RAM to Kafka
  - An older producer / consumer runs against a broker with a new log message format

- We're worried about the second case, because we perform Kafka upgrades.
- The log format doesn't change often, the last time being 0.11

- In that case, it was advised to upgrade clients first before upgrading the Kafka log format version.

Rolling Restart of Brokers

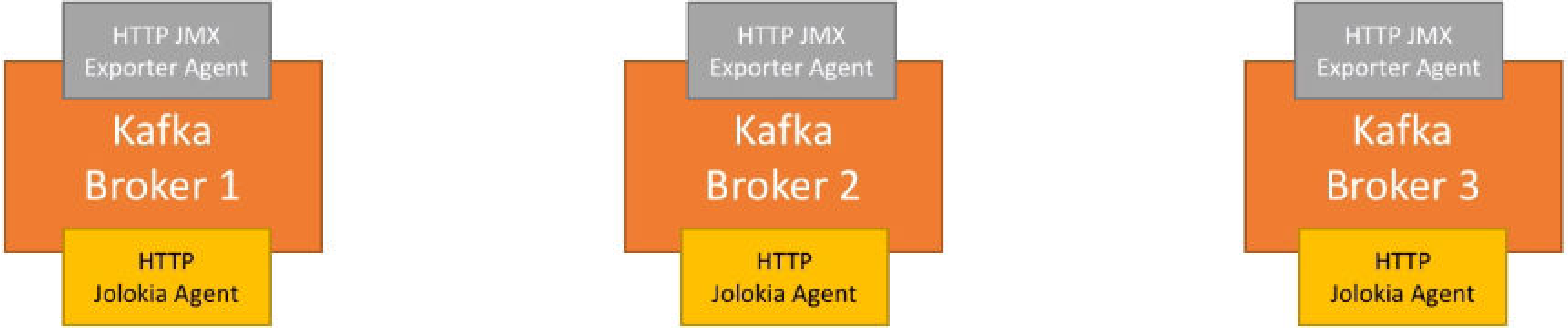
- For most of the Kafka changes you will operate, you will need to perform a roll restart of your Kafka Brokers.

- Doing a manual Roll Restart means:
  - Shutdown down a broker
  - Updating that broker
  - Starting the broker
  - Wait for the cluster to be stable
  - Repeat

- It would be better if we could automate this!

Roll Restart of Brokers
Installing Jolokia Agent

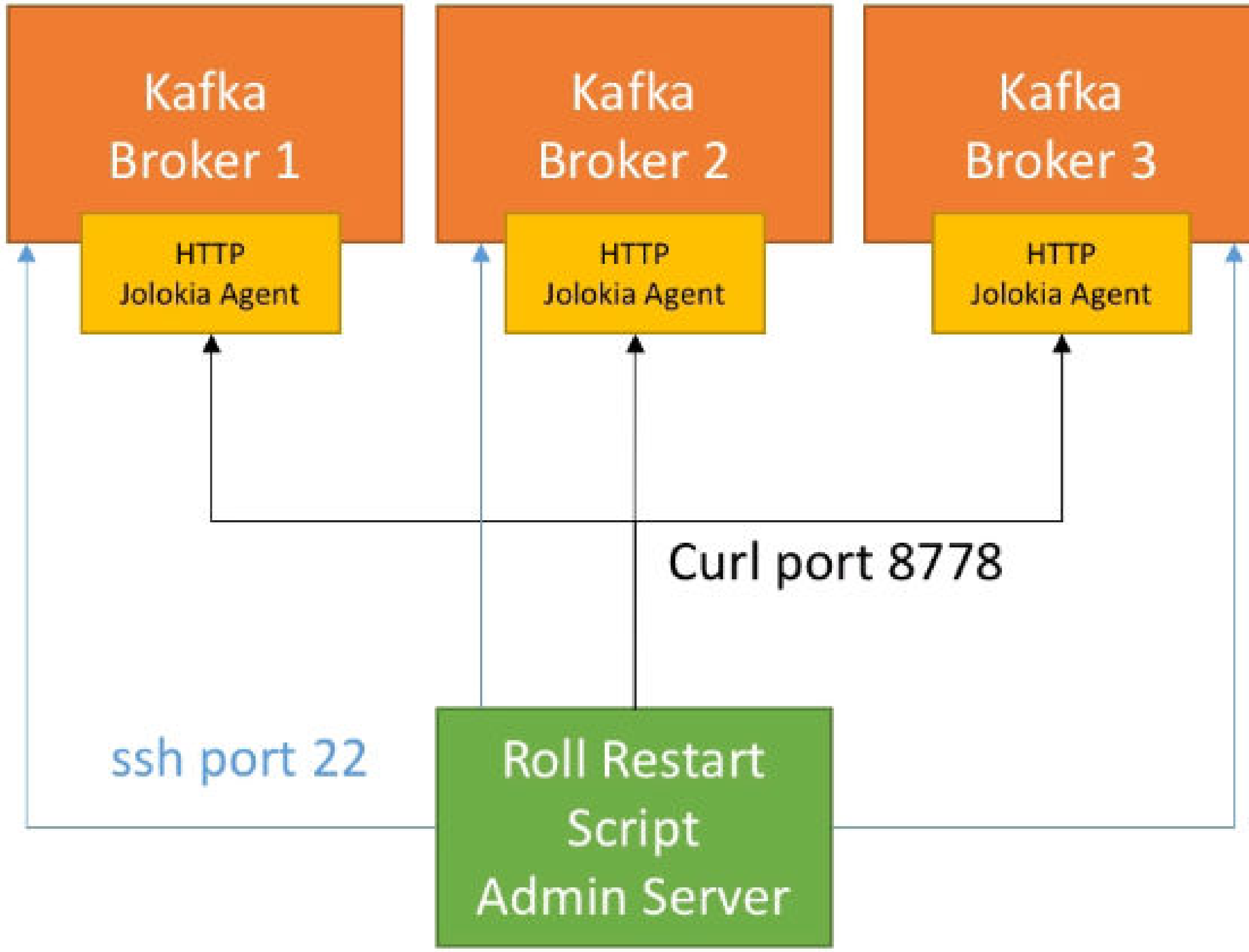
- We can use the kafka-utils by Yelp to perform automated roll restarts.



- Let’s install the Jolokia Agents on our Brokers

Rolling Restart of Brokers
Installing Yelp Tools

- The yelp tools will be installed on the administration machine
- We'll generate a SSH key
- We'll need to copy our SSH public keys to the brokers
- We'll test SSH is working from the administration machine
- We'll test kafka roll restart using Kafka-Tools by Yelp!



Kafka Upgrades Steps and Documentation.

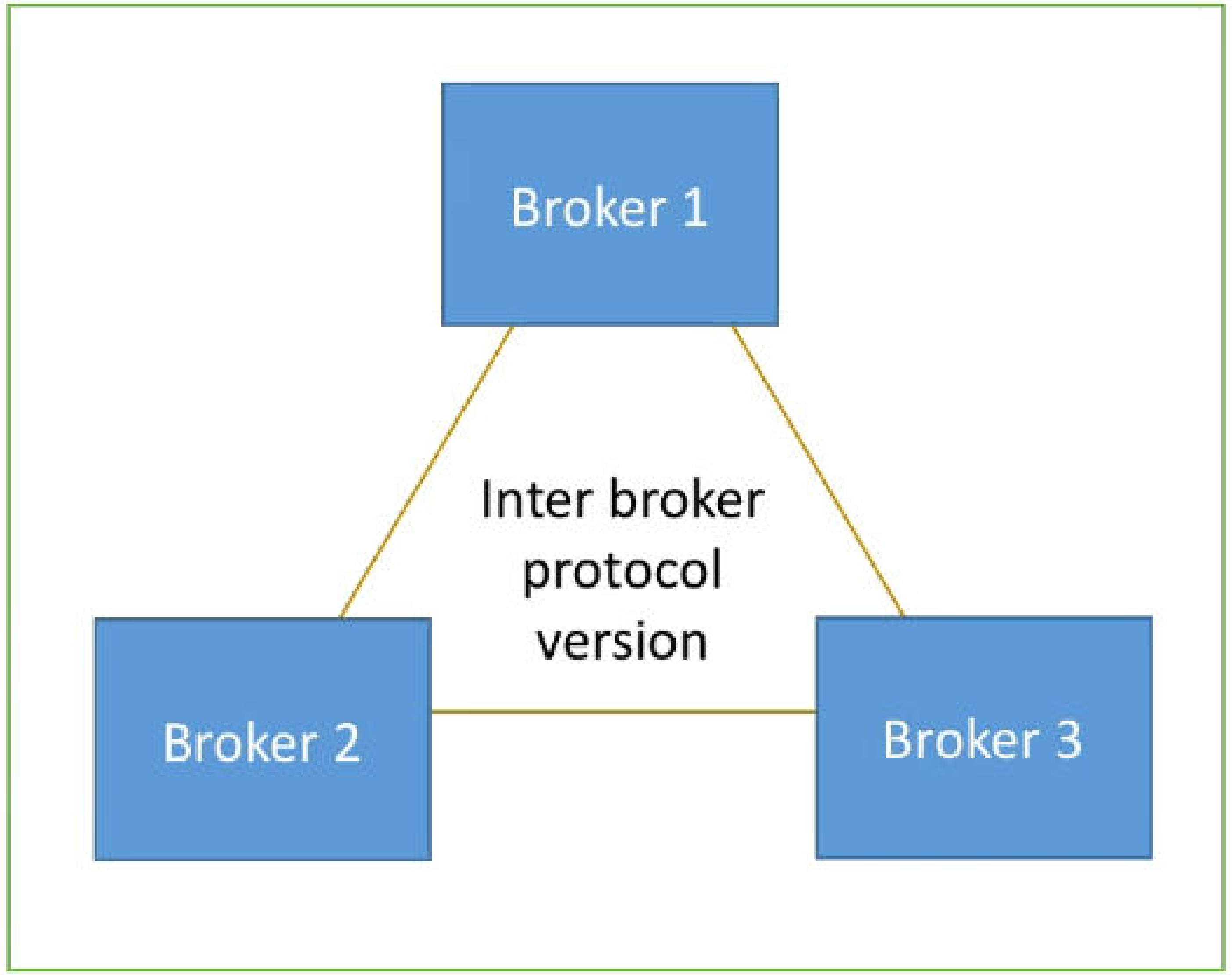
- Steps we have to perform (roll restart at the end of each step)

1. Setting inter broker and log version to current Kafka Version
2. Upgrade Kafka Binaries
3. Change inter broker protocol version
4. Upgrade Kafka Clients if documentation specifies it (all or most of them to avoid up & down conversion)
5. Upgrade message protocol version

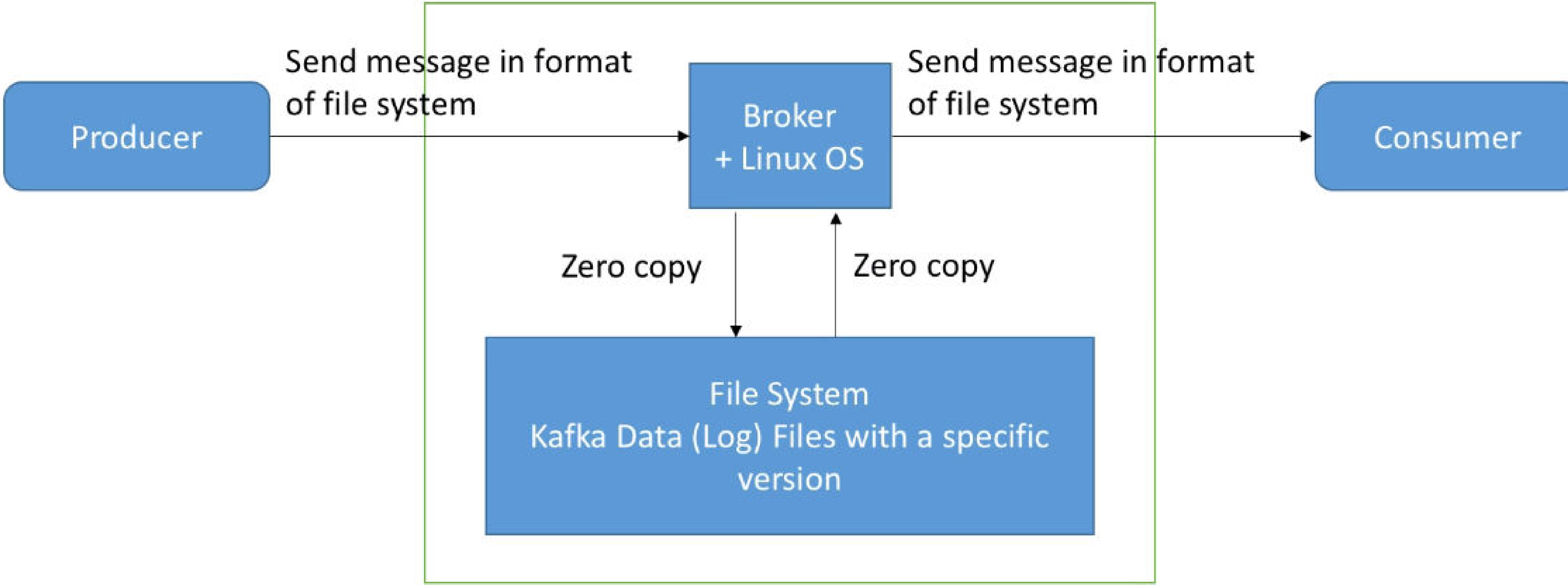
Always read documentation at:
https://kafka.apache.org/documentation/#upgrade

How brokers actually work Inter Broker Protocol Version

- Brokers communicate with each other using a specific protocol version.
- At each Kafka upgrade, brokers use a new protocol version to communicate to one another
- Newer brokers (ex v2.0) know all about all the protocol versions with older brokers (ex v1.1, 0.10, etc..)
- <u>Upgrading brokers require to first update the software, then telling all the brokers to use a new inter protocol version to talk to each other</u>



How brokers actually work
Message Format Version


Producer
Send message in format of file system
Broker + Linux OS
Send message in format of file system
Consumer
Zero copy
Zero copy
File System
Kafka Data (Log) Files with a specific version

Kafka Upgrades from 1.1 to 2.0 Hands-On

- Steps we have to perform (roll restart at the end of each step)

1. Setting inter broker and log version to current Kafka Version
2. Upgrade Kafka Binaries
3. Change inter broker protocol version
4. Upgrade Kafka Clients if documentation specifies it (all or most of them to avoid up & down conversion)
5. Upgrade message protocol version

Client Bi-Directional Compatibility

- As of Kafka 0.10.2 (introduced in June 2017), your clients & Kafka Brokers have a capability called bi-directional compatibility (because API calls are now versioned)

- This means:
  - An OLDER client (ex 1.1) can talk to a NEWER broker (2.0)
  - A NEWER client (ex 2.0) can talk to an OLDER broker (1.1)
- Bottom Line:
  - Always use the latest client library version if you can
  - You can upgrade Kafka without breaking your client applications

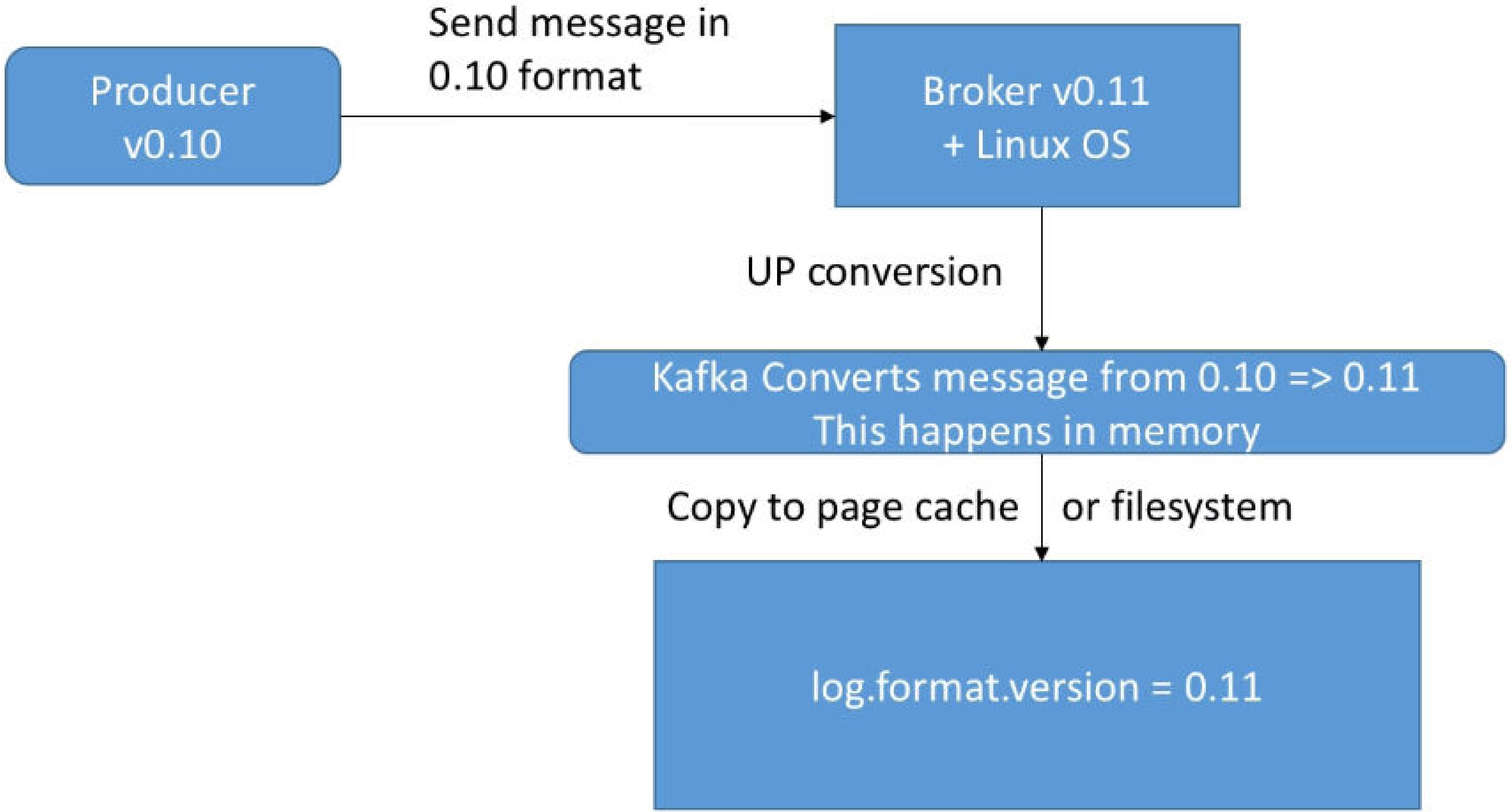
- More Reading: https://www.confluent.io/blog/upgrading-apache-kafka-clients-just-got-easier/

Up Conversions and Down Conversions

- When zero copy is lost, because an older client is run against a new log message format version, we run into up and down conversions, which has a performance impact

- In Kafka 2.0, Up and Down Conversion memory usage has been improved: https://cwiki.apache.org/confluence/display/KAFKA/KIP-283%3A+Efficient+Memory+Usage+for+Down-Conversion

Up Conversion


Producer
v0.10
Send message in
0.10 format
Broker v0.11
+ Linux OS
UP conversion
Kafka Converts message from 0.10 => 0.11
This happens in memory
Copy to page cache or filesystem
log.format.version = 0.11

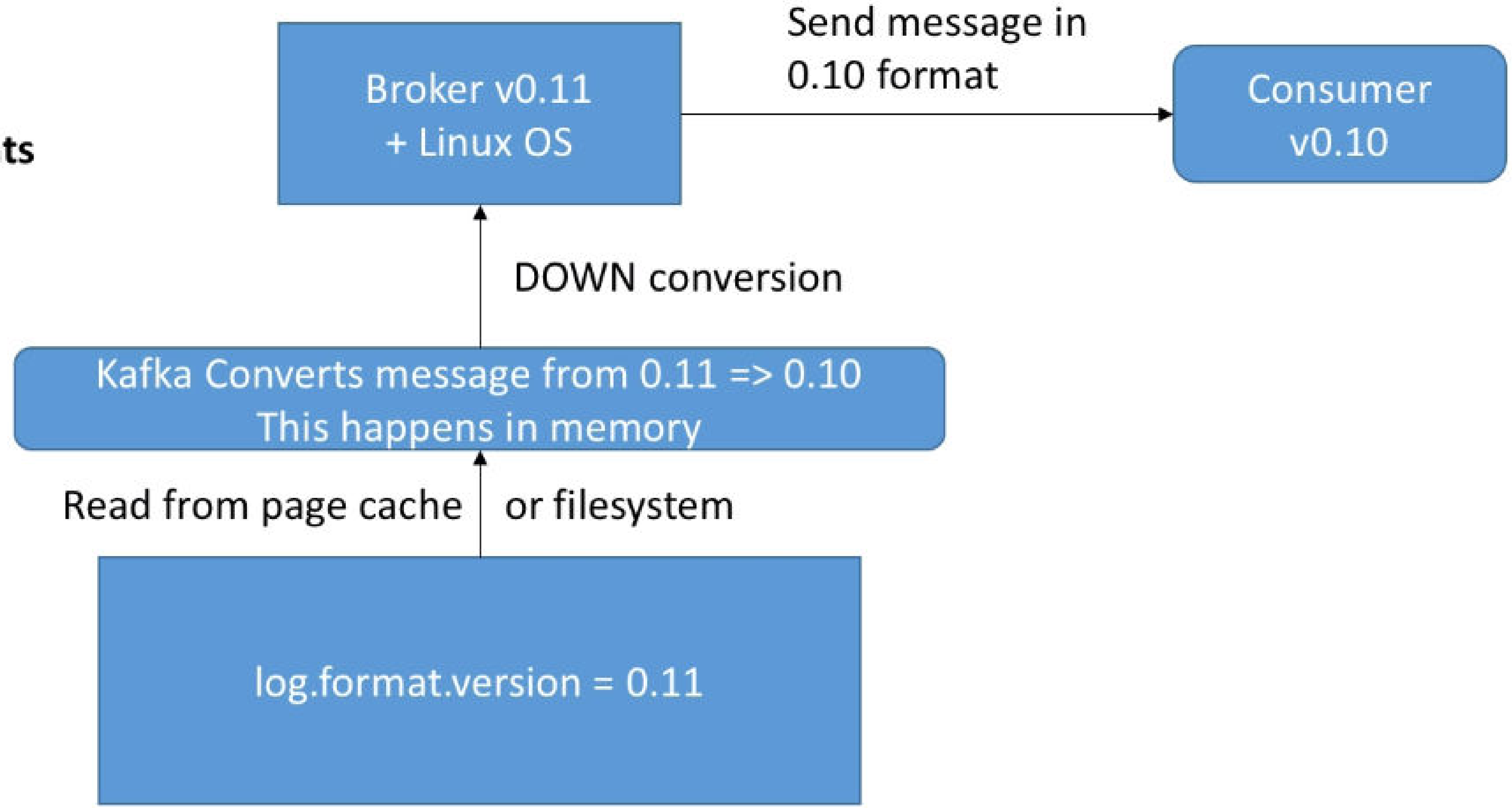
No zero copy in this case!

Note: this is only for older clients

Down Conversion

No zero copy in this case!

Note: this is only for older clients


Broker v0.11
+ Linux OS
Send message in
0.10 format
Consumer
v0.10
DOWN conversion
Kafka Converts message from 0.11 => 0.10
This happens in memory
Read from page cache
or filesystem
log.format.version = 0.11