0-Recommended Maximum File Sizes and Configuration

Please work with your systems office or hosting to provider to ensure these configurations are in place. After more performance, reliability, and stability work, these recommendations will be re-evaluated.

For additional information, please see the following:

MARC Bibs: Performance Testing Data Import (Nolana)

Folijet - Morning Glory Snapshot Performance testing

MARC Bibs: Performance Rancher testing (Lotus)

MARC Bibs: Performance Testing Data Import (Juniper/Kiwi)

MARC Bibs: Performance Testing Data Import (Iris)

MARC Bibs: Profiles used for PTF Testing

MARC Bibs: PTF Test results

Maximum File Size (MG) no background activity, no concurrent import jobs

- **CREATE** (SRS MARC, Instances, Holdings, Items): **100,000 MARC Bib records** (details)
- **UPDATE** (SRS MARC, Instances, Holdings, Items): **50,000 MARC Bib records**
- Still testing Data Import Lotus performance with background activity and concurrent import jobs

Import Statistics, with recommended values no background activity, no concurrent import jobs

	Morning Glory		Lotus				Juniper and Kiwi	
	2 instances		1 instance		2 instances		2 instances	
MARC Records	CREATE Duration	UPDATE Duration	CREATE Duration	UPDATE Duration	CREATE Duration	UPDATE Duration	CREATE Duration	UPDATE Duration
5000	7min	11min	13min	16min	8min	13min	15 minutes	20-30 minutes (depending on complexity of profile)
10,000	16min	22min	31min	40min	19min	25min		
30,000			1h 31min	1h 4 min	45min	1h 36min		
50,000	59min	1h42min	1h 34min	2h 17min	1h 21min	1h 51min	2h 30min+	11+ hours
100,000	2h20min	2h49min	3h 21min		3h 10min	4h 10min		

Maximum File Sizes (Juniper and Kiwi)

- **CREATE** Import (SRS MARC, Instances, Holdings, Items): **50,000 MARC records max**
- **UPDATE** Import: **5,000 MARC records max**

Import Statistics, with background activity

- With 5-users check-in/out background activities
- And concurrent imports by different tenants
    - Check-in/out time increases by 50%-100% depending on number of concurrent users.
    - Data import takes 2x longer to complete.

MARC Records	CREATE Duration	UPDATE Duration
1,000	10 minutes	10 minutes
5,000	30 minutes	40-60 minutes (depending on complexity of profile)
25,000	2-3 hours	8+ hours
50,000	5+ hours	22+ hours

Import Statistics, no background activity, no concurrent import jobs

MARC Records	CREATE Duration	UPDATE Duration

1,000	5 minutes	5 minutes
5,000	15 minutes	20-30 minutes (depending on complexity of profile)
25,000	60-80 minutes	4+ hours
50,000	150+ minutes	11+ hours

Note: For Lotus 500,000 Create Duration -15h 37min (details)

Key Settings and Configurations

mod-data-import

Properties related to file upload that should be set at mod-configuration are described in the doc https://github.com/folio-org/mod-data-import#module-properties-to-set-up-at-mod-configuration

System property that can be adjusted	Default value
file.processing.marc.raw.buffer.chunk.size	50
file.processing.marc.json.buffer.chunk.size	50
file.processing.marc.xml.buffer.chunk.size	10
file.processing.edifact.buffer.chunk.size	10

For releases prior to Kiwi it is recommended to set the ***file.processing.buffer.chunk.size*** property to **5** in order to prevent mod-source-record-storage from crashing with OOM during an Update import of 5,000 records. The property can be set to **1** to allow Update import of 10,000 records.

mod-source-record-manager

System property that can be adjusted	Default value	Comment	
srm.kafka.RawMarcChunkConsumer.instancesNumber	1		
srm.kafka.StoredMarcChunkConsumer.instancesNumber	1		
srm.kafka.DataImportConsumersVerticle.instancesNumber	1		
srm.kafka.DataImportJournalConsumersVerticle.instancesNumber	1		
srm.kafka.RawChunksKafkaHandler.maxDistributionNum	100		
srm.kafka.CreatedRecordsKafkaHandler.maxDistributionNum	100		

srm.kafka.DataImportConsumer.loadLimit	5		
security.protocol	PLAINTEXT		
ssl.protocol	TLSv1.2		
ssl.key.password	-		
ssl.keystore.location	-		
ssl.keystore.password	-		
ssl.keystore.type	JKS		
ssl.truststore.location	-		
ssl.truststore.password	-		
ssl.truststore.type	JKS		
di.flow.control.max.simultaneous.records	50	Defines how many records can be processed by the system simultaneously	Morning Glory
di.flow.control.records.threshold	25	Defines how many records from previous batch must be processed before throwing new records to pipeline	Morning Glory
di.flow.control.enable	true	Allows for single record imports to be processed while larger file imports are being processed	Morning Glory
di.flow.control.reset.state.interval	PT5M	Time between resetting state of flow control. By default it triggered each 5 mins.	Morning Glory
kafka.producer.batch.size	16*1024	Producer will attempt to batch records together into fewer requests whenever multiple records are being sent to the same partition. This helps performance on both the client and the server. This configuration controls the default batch size in bytes	Nolana
kafka.producer.linger.ms	0	Producer groups together any records that arrive in between request transmissions into a single batched request. Normally this occurs only under load when records arrive faster than they can be sent out. However in some circumstances the client may want to reduce the number of requests even under moderate load. This setting accomplishes this by adding a small amount of artificial delay that is, rather than immediately sending out a record the producer will wait for up to the given delay to allow other records to be sent so that the sends can be batched together	Nolana
kafka.producer.request.timeout.ms	30000	The configuration controls the maximum amount of time the client will wait for the response of a request. If the response is not received before the timeout elapses the client will resend the request if necessary or fail the request if retries are exhausted	Nolana
kafka.producer.delivery.timeout.ms	120000	An upper bound on the time to report success or failure after a call to send() returns. This limits the total time that a record will be delayed prior to sending, the time to await acknowledgement from the broker (if expected), and the time allowed for retriable send failures. **kafka.producer.delivery.timeout.ms must be equal to or larger than kafka.producer.linger.ms + kafka.producer.request.timeout.ms**	Nolana
kafka.producer.retry.backoff.ms	100	Time to wait before attempting to retry a failed request to a given topic partition	Nolana

mod-source-record-storage

System property that can be adjusted	Default value
srs.kafka.ParsedMarcChunkConsumer.instancesNumber	1
srs.kafka.DataImportConsumer.instancesNumber	1
srs.kafka.ParsedRecordChunksKafkaHandler.maxDistributionNum	100
srs.kafka.DataImportConsumer.loadLimit	5

srs.kafka.DataImportConsumerVerticle.maxDistributionNum	100
srs.kafka.ParsedMarcChunkConsumer.loadLimit	5
security.protocol	PLAINTEXT
ssl.protocol	TLSv1.2
ssl.key.password	-
ssl.keystore.location	-
ssl.keystore.password	-
ssl.keystore.type	-
ssl.truststore.location	JKS
ssl.truststore.password	-
ssl.truststore.type	JKS

mod-inventory

System property that can be adjusted	Default value
inventory.kafka.DataImportConsumerVerticle.instancesNumber	3
inventory.kafka.MarcBibInstanceHridSetConsumerVerticle.instancesNumber	3
inventory.kafka.DataImportConsumer.loadLimit	5
inventory.kafka.DataImportConsumerVerticle.maxDistributionNumber	100
inventory.kafka.MarcBibInstanceHridSetConsumer.loadLimit	5
security.protocol	PLAINTEXT
ssl.protocol	TLSv1.2
ssl.key.password	-
ssl.keystore.location	-
ssl.keystore.password	-
ssl.keystore.type	JKS
ssl.truststore.location	-
ssl.truststore.password	-
ssl.truststore.type	JKS

mod-invoice

System property that can be adjusted	Default value
mod.invoice.kafka.DataImportConsumerVerticle.instancesNumber	1
mod.invoice.kafka.DataImportConsumer.loadLimit	5
mod.invoice.kafka.DataImportConsumerVerticle. maxDistributionNumber	100
dataimport.consumer.verticle.mandatory	false *should be set to **true** in order to fail the module at start-up if data import Kafka consumer creation failed*
security.protocol	PLAINTEXT
ssl.protocol	TLSv1.2

ssl.key.password	-
ssl.keystore.location	-
ssl.keystore.password	-
ssl.keystore.type	JKS
ssl.truststore.location	-
ssl.truststore.password	-
ssl.truststore.type	JKS

More information on configuring modules involved in Data Import process can be found at this link.

- **Kafka (MSK)**:
  - auto.create.topics.enable = true
  - log.retention.minutes = 70-300
  - Broker's disk space: 300 GB
  - 4 brokers, replication factor = 3, DI topics partition = 2
  - Version 2.7 is 30% faster than version 1.6

JVM Settings

module	running instances	CPU	memory	memoryReservation	maxMetaspaceSize	Xmx
mod-data-import	1	256	2048	1844	512m	1292m
mod-source-record-manager	2	1024	2048	1844	800m	1024m
mod-source-record-storage	2	1024	1536	1440	512m	908m
mod-data-import-converter-storage	2	128	1024	896	128m	768m
mod-inventory	2	1024	2880	2592	512m	1814m
mod-inventory-storage	2	1024	2208	1952	512m	1440m

Key Improvements Delivered for Morning Glory:

- Implemented Flow control mechanism that allows OCLC single record imports to be prioritised over imports of large files
- Alleviated eventloop blocking during batch save of records
- Reduced conversion of parsed content into a marc4j record
- Improved performance of sql query for retrieving log entries
- Fixed race conditions during mapping parameter initialization
- Removed JobExecutionCache
- Optimised functionality to create Instance records

Key Improvements Delivered for Kiwi/Lotus:

- Performance
  - Improve speed and number of records for CREATEs and UPDATEs
  - Reduce Kafka message size: 6 Data Import topics messages >200KB/ea
  - Use less CPU for (de)serialization
  - Significantly improve speed of loading UI Landing page
  - Improve and optimize slow DB queries
- Remove events_cache topic and replace it by DB deduplication solutions (causes spikes in mod-inventory and the brokers, leads to instability and unpredictable outcomes)
- Improve resiliency and error handling to prevent import job from getting stuck

Key Improvements Delivered as of Iris Hotfix 3 and Juniper:

- No more accidental creation of duplicate records
- Ability to consistently make repeated updates on existing records
- CPU usage of multiple instances of the Data Import landing page now consumes 10% CPU, instead of maxing out at 100%
- When idle, mod-source-record-manager consumes 50% CPU instead of 90%
- Vertical scaling improves performance, especially with Iris HF2 Data Import modules