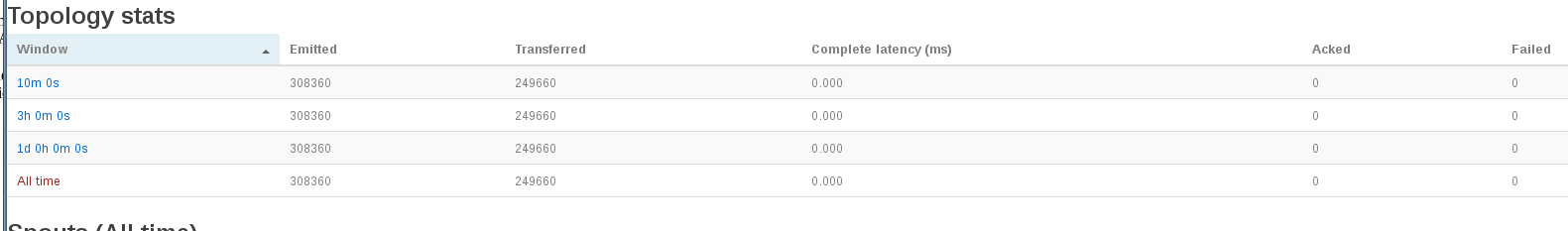
Metrics

Storm provides a set of metrics which it displays in the UI. These metrics are available through the Java API.

Connect using a NimbusClient object which communicates to the Nimbus Server to access the Storm Metrics Objects. Access to these objects is useful for understanding why both the ack count doesn't match what is actually sent and for creating a custom scheduler.



1) Create a connection to the nimbus server using the nimbus client:

NimbusClient nc **=** **new** NimbusClient**(**"localhost"**,**6627**);**

Nimbus**.**Client client **=** nc**.**getClient**();**

2) get the cluster summary which allows access to the storm metrics data objects. This is identical to the Topology and Cluster objects we use in the Scheduler document.

ClusterSummary cs **=** client**.**getClusterInfo**();**

System**.**out**.**println**(**"cs:"**+**cs**.**toString**());**

3) Access the topologies and descend to the Supervisor objects

List**<**TopologySummary**>** localTopos **=** cs**.**get\_topologies**();**

**for(**TopologySummary ts:localTopos**){**

System**.**out**.**println**(**"Topology Name:"**+**ts**.**get\_name**());**

System**.**out**.**println**(**"Topology id:"**+**ts**.**get\_id**());**

System**.**out**.**println**(**"Topology numexecutors:"**+**ts**.**get\_num\_executors**());**

System**.**out**.**println**(**"Topology numtasks:"**+**ts**.**get\_num\_tasks**());**

System**.**out**.**println**(**"Topology numworkers:"**+**ts**.**get\_num\_workers**());**

System**.**out**.**println**(**"Topology status:"**+**ts**.**get\_status**());**

System**.**out**.**println**(**"Topology uptime:"**+**ts**.**get\_uptime\_secs**());**

**}**

List**<**SupervisorSummary**>** localListSups **=** cs**.**get\_supervisors**();**

**for(**SupervisorSummary s: localListSups**){**

System**.**out**.**println**(**"supervisor numworkers:"**+**s**.**get\_num\_workers**());**

System**.**out**.**println**(**"supervisor host:"**+**s**.**get\_host**());**

System**.**out**.**println**(**"supervisor num used workers:"**+**s**.**get\_num\_used\_workers**());**

System**.**out**.**println**(**"supervisor uptime seconds:"**+**s**.**get\_uptime\_secs**());**

**}**

4) the TopologyInfo objects contain the components (spouts and bolts) to executor mappings

TopologyInfo ti **=** client**.**getTopologyInfo**(**topologyId**);**

System**.**out**.**println**(**"topology id:"**+**topologyId**);**

System**.**out**.**println**(**"Topology Name:"**+**ti**.**get\_name**());**

System**.**out**.**println**(**"Topology executors size:"**+**ti**.**get\_executors\_size**());**

System**.**out**.**println**(**"Topology status:"**+**ti**.**get\_status**());**

System**.**out**.**println**(**"Topology uptime:"**+**ti**.**get\_uptime\_secs**());**

System**.**out**.**println**(**"Topology executors size:"**+**ti**.**get\_executors**().**size**());**

java**.**util**.**List**<**ExecutorSummary**>** execSum **=** ti**.**get\_executors**();**

*// ExecutorSummary es = execSum.get(0);*

*// System.out.println("asdfasdf:"+es.get\_stats().get\_emitted().get("600").get("\_\_ack\_ack"));*

System**.**out**.**println**(**"+++++++++++++++++++++++++++++++"**);**

System**.**out**.**println**(**"+++++++++++++++++++++++++++++++"**);**

**for(**ExecutorSummary e **:** execSum**){**

System**.**out**.**println**(**"ExecutorSummary component\_id:"**+**e**.**get\_component\_id**());**

System**.**out**.**println**(**"ExecutorSummary host:"**+**e**.**get\_host**());**

System**.**out**.**println**(**"ExecutorSummary port:"**+**e**.**get\_port**());**

System**.**out**.**println**(**"ExecutorSummary uptime:"**+**e**.**get\_uptime\_secs**());**

ExecutorInfo ei **=** e**.**get\_executor\_info**();**

System**.**out**.**println**(**"ExecutorInfo taskStart:"**+**ei**.**get\_task\_start**());**

System**.**out**.**println**(**"ExecutorInfo taskEnd:"**+**ei**.**get\_task\_end**());**

*//careful there are 2 imports here... one for generated one for the daemon dont know the difference*

backtype**.**storm**.**generated**.**ExecutorStats es1 **=** e**.**get\_stats**();**

**if(**es1**!=null){**

System**.**out**.**println**(**"ExecutorStats emittedSize:"**+**es1**.**get\_emitted\_size**());**

System**.**out**.**println**(**"ExecutorStats transferredSize:"**+**es1**.**get\_transferred\_size**());**

Map**<**String**,** Map**<**String**,**Long**>>** emittedStats **=** es1**.**get\_emitted**();**

**for(**String procEmit: emittedStats**.**keySet**()){**

System**.**out**.**println**(**"Emitted Stats key:"**+**procEmit**);**

Map**<**String**,**Long**>** m **=** emittedStats**.**get**(**procEmit**);**

**for(**String emitS **:** m**.**keySet**()){**

System**.**out**.**println**(**"EmittedStats key:"**+**emitS**+**" value:"**+**m**.**get**(**emitS**));**

**}**

**}**

**for(**String s:es1**.**get\_transferred**().**keySet**()){**

**for(**String s1:es1**.**get\_transferred**().**keySet**()){**

System**.**out**.**println**(**" statK:"**+**s1**+**" value:"**+**es1**.**get\_transferred**());**

**}**

**}**

**}else{**

System**.**out**.**println**(**"STATS NULL!!!!!"**);**

**}**

**}**

The metrics sampling rate is controlled by topology.stats.sample.rate set to ½ second:

topology.stats.sample.rate: 0.05

Supervisors, Workers, Tasks

A supervisor resides on each node/server. Each supervisor runs in a JVM, we can get a list of supervisors/cluster. From there we can determine which supervisors live on which machine. We can specify this using both the hostname in Supervisor details and getSupervisorMeta().

Example; print out in the scheduler which tasks are running where, replicate this with the output of the logs.