# FatWire | Analytics 2.5

## Performance and Scalability

**Document Revision Date:** Jul. 30, 2009

FatWire® SOFTWARE

*FatWire Analytics: Performance and Scalability*
Document Revision Date: Jul. 30, 2009
Product Version: 2.5

**FatWire Technical Support**
www.fatwire.com/Support

**FatWire Headquarters**
FatWire Corporation
330 Old Country Road
Suite 207
Mineola, NY 11501
www.fatwire.com

Table of

# Contents

# Analytics 2.5 Performance Testing

The purpose of this guide is to provide you with performance and scalability testing results for the Analytics 2.5 installation.

This guide contains the following sections:

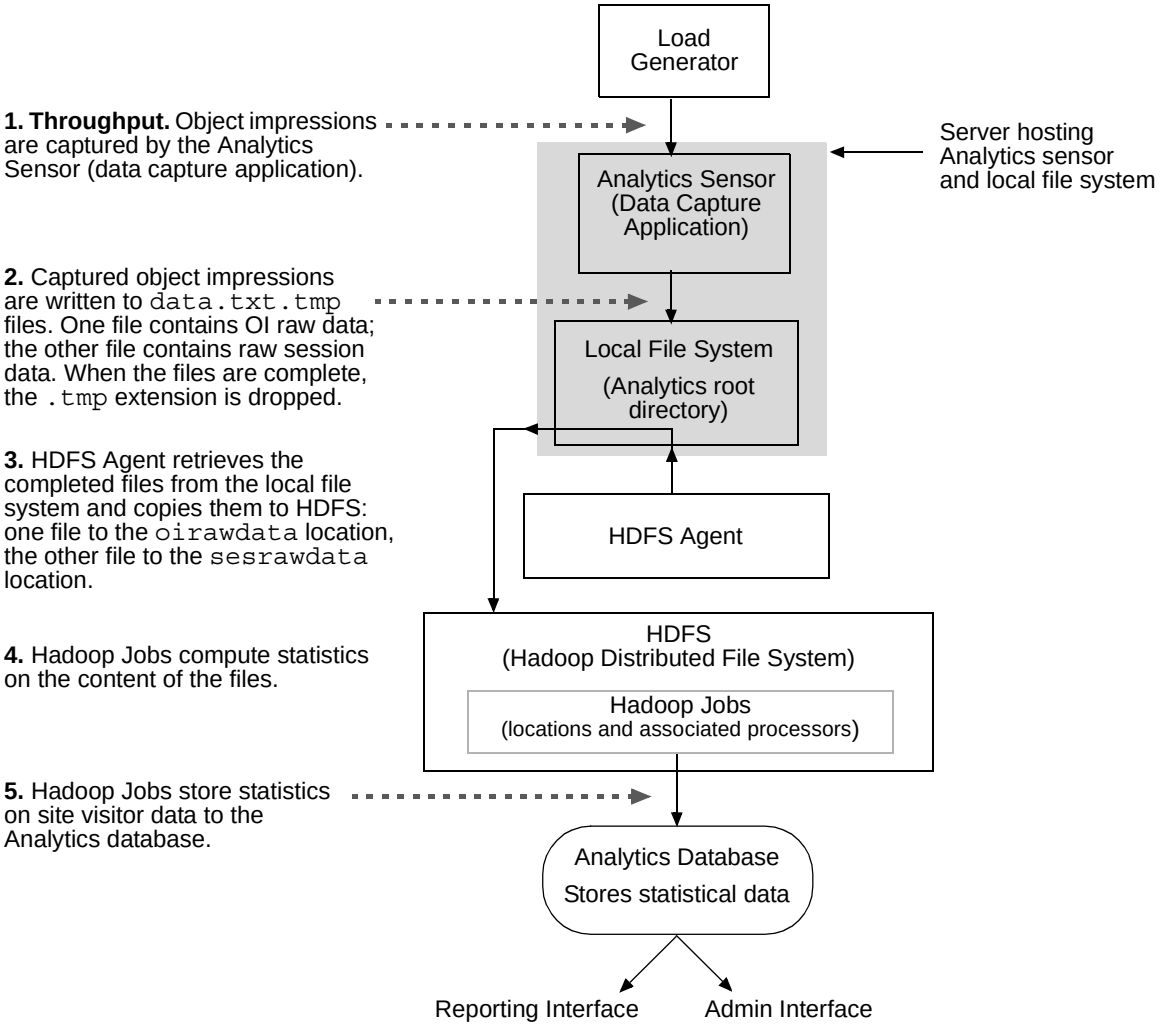
# Goals

Our first goal was to determine the maximum throughput that an Analytics system can support while processing 24 hours of session raw data. We tested the following systems: Analytics sensor host with one, two and three Hadoop nodes, and a single-server system (where both the Analytics sensor and Hadoop are installed). Our second goal was to demonstrate the scalability of our results.

Throughput is depicted as step 1 in Figure 1. **Throughput** is defined as the number of object impressions (**OI**) that are captured per second by the Analytics sensor. The same number of object impressions must be written to each of the `data.txt.tmp` files.

When the Analytics sensor is not overloaded (as in our tests), throughput is determined by how efficiently Hadoop Jobs processes its current volume of data. Typically the most resource-intensive computations involve session raw data. For this reason, we tested throughput while Hadoop Jobs processed session raw data.

**Figure 1:** Throughput and Data Processing



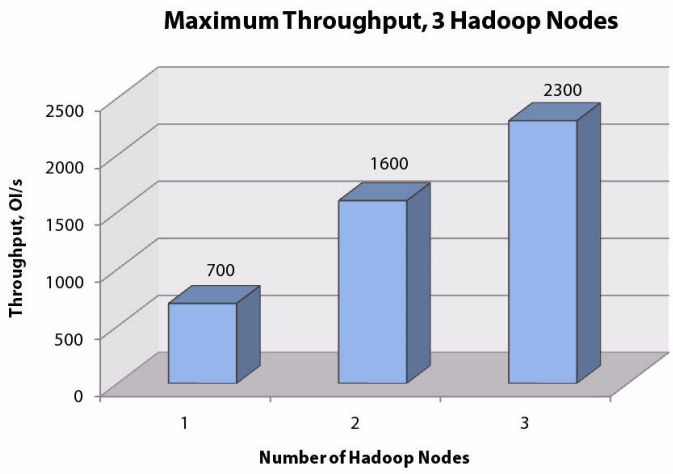
# Summary of Results

Four configuration scenarios (page 8) were tested for throughput. The results are summarized below. A full analysis can be found in "Results," on page 16.

## Analytics with Hadoop Nodes

The maximum supported throughput for the specified hardware is 700 OI/s per second for a single-node system, 1,600 OI/s for a two-node system, and 2,300 OI/s for a three-node system.

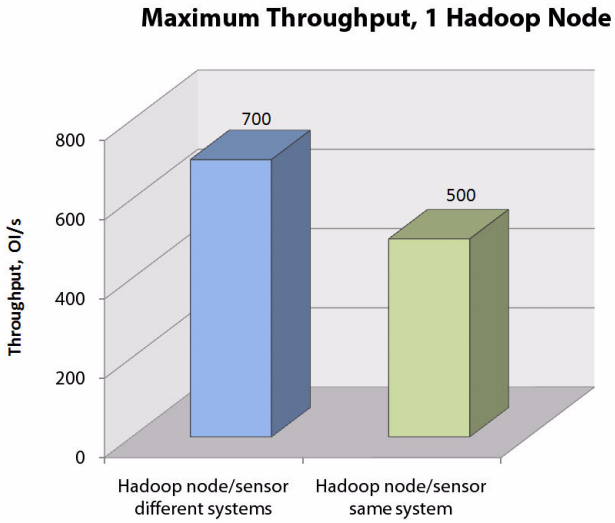
Analytics 2.5 shows the biggest performance improvement when a second Hadoop node is added. The scalability is near-linear for each additional node. These scenarios also show that a single sensor was sufficient in handling 2,300 object impressions per second (about 200 million hits per day).




Aside from adding another Hadoop node, using faster disks for the Hadoop file system is another way to improve performance of an Analytics 2.5 environment.

## Single-Server System

When Hadoop and the sensor are on the same server, the maximum throughput (500 OI/s) is significantly less than for scenario 1, where the two components are installed on separate servers (700 OI/s maximum throughput).



# Configurations Tested

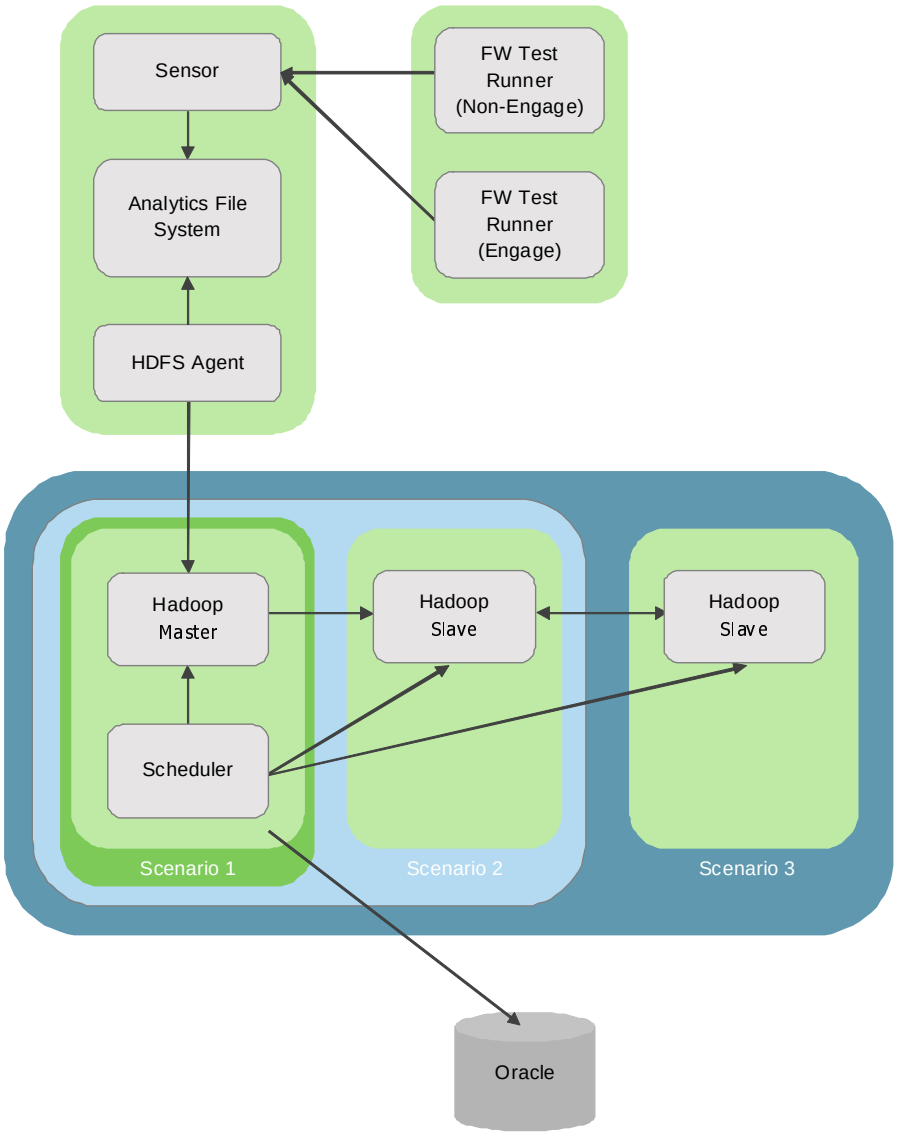
Four Analytics configurations were tested for scalability and throughput:

- Test Scenario 1: Analytics with a Single Hadoop Node
- Test Scenario 2: Analytics with Two Hadoop Nodes
- Test Scenario 3: Analytics with Three Hadoop Nodes
- Test Scenario 4: One Hadoop Node and Sensor on the Same Server

## Analytics with Hadoop Nodes

Figure 2 illustrates the configuration of Analytics systems with one, two, and three Hadoop nodes. (**HDFS** stands for "Hadoop Distributed File System.")

**Figure 2:** Test Scenarios 1, 2, and 3



## Test Scenario 1: Analytics with a Single Hadoop Node

The Analytics environment used for this test includes:

- A system with the sensor web application, HDFS Agent, and Analytics file system
- A single Hadoop node: Hadoop master node and scheduler application
- A system with an Oracle 11g database
- A system with the two FW TestRunner load generators

## Test Scenario 2: Analytics with Two Hadoop Nodes

The Analytics environment used for this test includes:

- A system with the sensor web application, HDFS Agent, and Analytics file system
- A system with two Hadoop nodes:
  - A system with the primary `NameNode`, Hadoop master node, and scheduler application
  - A system with the secondary `NameNode` and Hadoop slave node
- A system with an Oracle 11g database
- A system with the two FW TestRunner load generators

## Test Scenario 3: Analytics with Three Hadoop Nodes

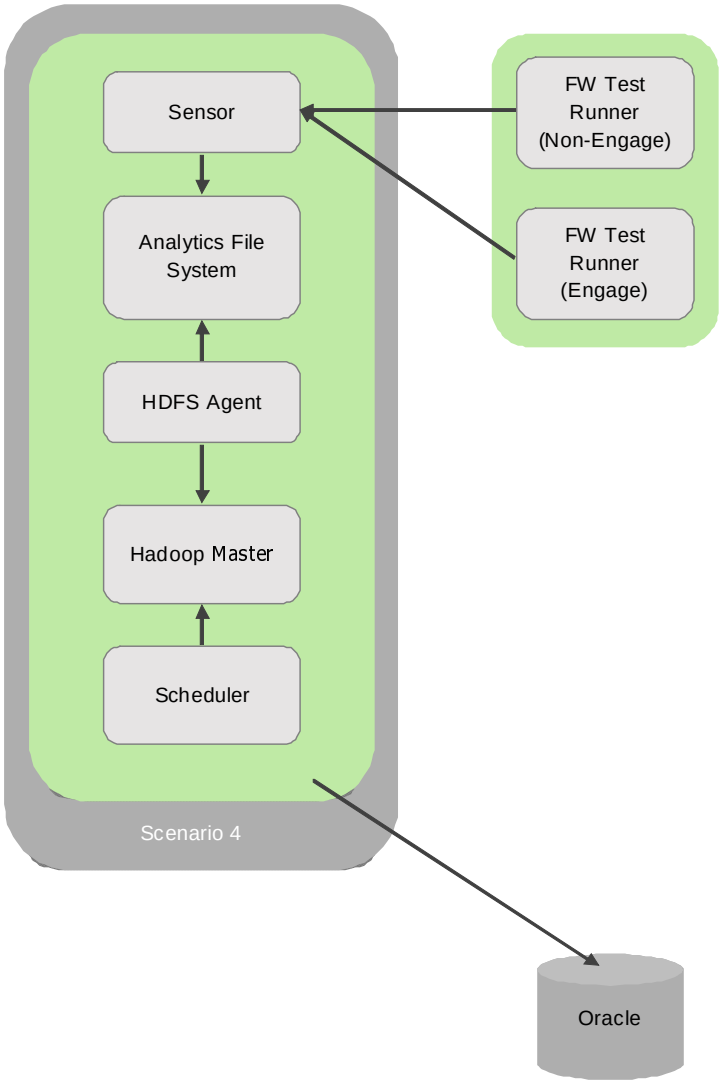
The Analytics environment used for this test includes:

- A system with the sensor web application, HDFS Agent, and Analytics file system
- A system with three Hadoop nodes:
  - A system with the primary `NameNode`, Hadoop master node, and scheduler application
  - A system with the secondary `NameNode` and Hadoop slave node
  - A system with the second Hadoop slave node
- A system with an Oracle 11g database
- A system with the two FW TestRunner load generators

## Single-Server System

Figure 3 illustrates the configuration of an Analytics system with the sensor and Hadoop on the same server. (**HDFS** stands for "Hadoop Distributed File System.")

**Figure 3:** Test Scenario 4




## Test Scenario 4: One Hadoop Node and Sensor on the Same Server

The Analytics environment used for this test includes:

- A single Analytics server with the following components: sensor web application, HDFS Agent, Analytics file system, Hadoop master node, and scheduler application
- A system with an Oracle 11g database
- A system with the two FW TestRunner load generators

## Hardware Specifications

Hardware used in all tests is the following.

- **Sensor**
  - Amazon High CPU Extra Large Machine Image
  - 20 EC2 Compute Units (2 x Intel Xeon E5345 Quad Core CPU @ 2.33GHz)
  - 7GB of Memory
  - 920GB Drive Space
- **Hadoop Nodes**
  - Amazon High CPU Extra Large Machine Image
  - 20 EC2 Compute Units (2 x Intel Xeon E5345 Quad Core CPU @ 2.33GHz)
  - 7GB of Memory
  - 10GB Drive Space
  - 1TB Amazon EBS Volume
- **Load Generators**
  - Amazon High CPU Extra Large Machine Image
  - 20 EC2 Compute Units (2 x Intel Xeon E5345 Quad Core CPU @ 2.33GHz)
  - 7GB of Memory
  - 10GB Drive Space
- **Database**
  - Amazon Standard Large Machine Image
  - 4 EC2 Compute Units (AMD Opteron 2218 HE Dual Core CPU @ 2.6GHz)
  - 7.5GB of Memory
  - 850GB Drive Space

## Software Specifications

Software used in all tests is the following:

- **Sensor**
  - Linux 2.6.24-23
  - Tomcat 6.0.16
  - Sun JDK 1.6.0.07 x64
- **Hadoop Nodes**
  - Linux 2.6.24-23
  - Sun JDK 1.6.0.07 x64
  - Hadoop 0.18.3
- **Load Generators**
  - Linux 2.6.24-23
  - Sun JDK 1.6.0.07 x64
- **Database**
  - Oracle Enterprise Linux 5.1 x64
  - Oracle 11g 64bit

# Test Specifications

When testing scenarios 1–4 (page 8), we measured throughput while Hadoop Jobs was processing session raw data. To determine if the system was overloaded, we monitored the duration of the Hadoop jobs, CPU utilization, and disk utilization.

## Session Raw Data

The greatest load occurs when an Analytics system collects new object impressions while processing the latest 24 hours of session raw data (using the session and visitor jobs).

For each test scenario, we pre-generated two `data.txt` files, each containing the equivalent of 24 hours of session raw data, collected under constant throughput. For every 100 object impressions per second (8.64 million hits per day), `data.txt` contains about 3.15GB of data. The following files were generated:

**Table 1:** Session Raw Data files

| Test Scenario | Session Raw Data Files Generated |
|---|---|
| 1 | • 22.05GB file representing 700 object impressions per second (about 60 million hits per day)<br>• 25.20GB file representing 800 object impressions per second (about 69 million hits per day) |
| 2 | • 50.40GB file representing 1,600 object impressions per second (about 138 million hits per day)<br>• 53.55GB file representing 1,700 object impressions per second (about 147 million hits per day) |
| 3 | • 69.30GB file representing 2,200 object impressions per second (about 190 million hits per day)<br>• 72.45GB file representing 2,300 object impressions per second (about 200 million hits per day) |
| 4 | • 15.75GB file representing 500 object impressions per second (about 43 million hits per day)<br>• 18.90GB file representing 600 object impressions per second (about 52 million hits per day) |

Data contained in the files was generated using two FatWire TestRunner load generators. One load generator created a sample file with non-Engage URLs; the other created a sample file with Engage URLs. The FatWire TestRunner load generators generate raw data at a 5:1 ratio of non-Engage to Engage data.

## Test Methods

Our goal was to find the maximum throughput that an Analytics environment can support. In each test, we monitored a given Analytics environment at a given throughput. In the next test, we increased the throughput, and so on, until the maximum supported throughput was found. The tested environments are shown on page 8.

We tested each environment as described next.

1. The Analytics environment was initialized:
   a. The Hadoop file system was formatted and the Analytics file system was cleared.
   b. Hadoop, the sensor, and the scheduler were started.
   c. The pre-generated session raw data file (`data.txt`) was copied to the local file system.
   d. The FatWire TestRunners for generating Engage and Non-Engage data were started.
   e. The HDFS Agent and monitoring scripts were started. The scripts monitored the sensor and Hadoop systems for 5 hours.
2. The test began with simultaneous data collection and data processing:
   a. At *time*=0 the sensor was hit with the given throughput. It responded by generating two new `data.txt.tmp` files: one file for OI raw data, the other file for session raw data. Simultaneously, the HDFS Agent copied the relevant pre-generated 24-hour session raw data file (see Table 1) to the `sesrawdata` location in Hadoop Jobs for processing (by session and visitor jobs).
3. The test continued for the next five hours:
   a. The sensor, continuously hit with the given throughput, recorded incoming OI raw data and session raw data, while Hadoop Jobs processed the pre-generated session raw data file.

      Throughout, we monitored the duration of Hadoop Jobs, CPU utilization, and disk utilization to determine whether overload occurred in the Analytics environment.
   b. Four hours after the test began, the Analytics sensor rotated the OI raw data file (from `data.txt.tmp` to `data.txt`), and HDFS Agent copied the rotated `data.txt` file to the `oirawdata` location in the Hadoop file system. In the meantime, the sensor generated a new OI raw data file in the local file system.

      **Note**

      For information about the properties that set rotation time and check for unprocessed files, see “Analytics Settings in global.xml,” on page 14.

   c. For the remainder of the test, the rotated `data.txt` file was processed by the scheduler (while new object impressions were recorded in the new OI raw data file, `data.txt.tmp`).

      **Note**

      The `data.txt.tmp` file for session raw data was not processed. Each performance test ran only 5 hours, whereas session raw data is processed at the end of 24 hours.
4. At 5 hours the test was terminated. The Analytics environment was re-initialized beginning with step 1, and the test was re-run at the next higher throughput. The cycle of tests was repeated until the maximum throughput for each Analytics environment was found.

# Tuning

To get the best performance from Analytics, we tuned a number of parameters in the Analytics and Hadoop configuration files:

- Analytics Settings in global.xml
- Hadoop Settings in hadoop-site.xml
- Hadoop Hosts in the ‘masters’ File
- Tomcat (Sensor)

## Analytics Settings in global.xml

In the code below, the bold lines were tuned to maximize performance of the Analytics system.

```
<param type="string" name="sensor.thresholdtime" value="240" />
<param type="string" name="session.rotate.delay" value="360" />
<param type="string" name="midnight.offset" value="0" />
<param type="string" name="NumberOfProcessorThreads" value="3" />
<param type="string" name="archive.enabled" value="false" />
<param type="string" name="scheduler.checkinterval" value="10" />
```

- The value for `sensor.thresholdtime` was increased from 10 minutes to 240 minutes (4 hours). Because the sensor waits a longer time between rotations of the object impression raw data file, it collects a larger data file. The scheduler handles larger files (on the order of gigabytes) more efficiently than a large amount of smaller files. Increasing the threshold time decreases the total amount of time the scheduler spends processing object impression data.
- The value for `scheduler.checkinterval` was increased to 10 minutes. This value increases the amount of time the scheduler waits between scans of the Hadoop file system before checking for unprocessed files. Since the frequency of files being uploaded to the Hadoop file system was reduced in the `sensor.thresholdtime` property, the performance hit created by the frequency of the scheduler scans was also reduced.

## Hadoop Settings in hadoop-site.xml

Table 2 lists the Hadoop properties in the `hadoop-site.xml` file that have been tuned for better performance.

**Table 2:** Hadoop Settings in hadoop-site.xml

| Property | Value | Description |
|---|---|---|
| dfs.replication | 1 | Represents one Hadoop node. |
| | 2 | Represents two Hadoop nodes. |
| | 3 | Represents three Hadoop nodes. |
| dfs.block.size | 268435456 | |

**Table 2:** Hadoop Settings in hadoop-site.xml

| Property | Value | Description |
|---|---|---|
| `mapred.child.java.opts` | `-Xmx768m` | The value for this property includes `-Xmx768m` to ensure that enough memory is allocated for each map and reduce task. Each task normally uses a couple hundred megabytes of RAM. |
| `mapred.map.tasks` | `20` | These properties control the total number of map and reduce tasks each job is split into. The slower the computer, the larger the amount of smaller tasks each job will be split into. In this case, the number of total map and reduce tasks is double the maximum amount of running map and reduce tasks. |
| `mapred.reduce.tasks` | `10` | |
| `mapred.tasktracker.map.tasks.maximum` | `10` | These properties control the maximum number of map and reduce tasks. Since map tasks are more CPU intensive, the maximum number of map tasks is based on the power of the CPUs used for the Hadoop nodes. Because reduce tasks are more memory intensive, the maximum number of reduce tasks is based on the amount of memory of the Hadoop nodes. |
| `mapred.tasktracker.reduce.tasks.maximum` | `5` | |

## Hadoop Hosts in the 'masters' File

The following changes were made to the Hadoop `masters` files:

- For scenarios 1 and 4, the host name of the Hadoop master node was specified in the `masters` file. Having a single Hadoop node requires running both the `NameNode` and `Secondary NameNode` on the same system.
- For scenarios 2 and 3, the host name of the first Hadoop slave node was specified in the `masters` file. Having multiple Hadoop nodes allows the `NameNode`, `Secondary NameNode`, and the load generated by the two to be split between multiple systems.

## Tomcat (Sensor)

- Java Options:

  ```
  -Xmx4096m -Dfile.encoding=UTF-8
  ```

  The amount of memory needed for the sensor is based on the number of object impressions that are expected to hit the sensor during the busiest period of time. With 4GB of RAM, the sensor was able to handle more than 2,300 object impressions per second, the maximum required for the test scenarios.

- HTTP Connector:

  ```
  connectionTimeout=20000
  maxThreads=5000
  ```

  Due to the large number of threads the sensor utilizes under load, a large number is recommended for the maximum allowed thread.

# Results



## Scenario 1: One Hadoop Node

Scenario 1 was designed to show the maximum throughput an Analytics environment with one Hadoop node running on the specified hardware (page 11) can support.

### Data and Analysis

- The first throughput tested was 700 object impressions per second. The sensor was hit with this amount of throughput for 5 hours while the 22.05GB session raw data file was being processed. Although the disk of the Hadoop node was under considerable load, the total load on the system was within acceptable limits. All session and visitor jobs were completed in 3 hours and 25 minutes, the longest of which was the `SessionMerger` job, completing in 2 hours.
- The second throughput tested was 800 object impressions per second. The sensor was hit with this amount of throughput for 5 hours while the 25.20GB session raw data file was processed. In comparison with the previous run, the number of requests on the disk in service or waiting for service was higher. There was also an increase in the average wait time per request. The load on the disk surpassed the acceptable limit. We concluded that 800 object impressions per second was more throughput than the single Hadoop node Analytics environment could support.

### Summary

The maximum throughput supported by this Analytics environment with one Hadoop node is 700 object impressions per second. During this test it was clear that the disk where the Hadoop file system resides created a bottleneck.

## Scenario 2: Two Hadoop Nodes

Scenario 2 was designed to show the maximum throughput an Analytics environment with two Hadoop nodes running on the specified hardware (page 11) can support.

### Data and Analysis

The first throughput tested was 1,600 object impressions per second. The sensor was hit with this amount of throughput for 5 hours while the 50.40GB session raw data file was processed. Although the disks of the Hadoop nodes were under considerable load, the total load on the system was within acceptable limits. All session and visitor jobs were completed in 3 hours and 20 minutes, the longest of which was the `SessionMerger` job, completing in 2 hours and 20 minutes.

The second throughput tested was 1,700 object impressions per second. The sensor was hit with this amount of throughput for 5 hours while the 53.55GB session raw data file was processed. In comparison with the previous run, the number of requests on the disks in service or waiting for service was higher. There was also an increase in the average wait

time per request. The load on the disk of the Hadoop master node surpassed the acceptable limit. We concluded that 1,700 object impressions per second was more throughput than the two-Hadoop node Analytics environment could support.

### Summary

The maximum throughput supported by this Analytics environment with two Hadoop nodes is 1,600 object impressions per second, which is more than double the throughput supported by the single Hadoop node environment. This greater than expected increase can be explained by the move of the secondary `NameNode` from the Hadoop master node to the Hadoop slave node. This configuration change was made for two reasons: to split the load from the file system between the two Hadoop nodes, and to split the load from the two `NameNodes`. Without this change, the Hadoop master node would be shouldering a much larger load than the slave nodes. This test again showed the disk on the Hadoop nodes to be the bottleneck.

## Scenario 3: Three Hadoop Nodes

Scenario 3 was designed to show the maximum throughput an Analytics environment with three Hadoop nodes running on the specified hardware (page 11) can support.

### Data and Analysis

The first throughput tested was 2,200 object impressions per second. The sensor was hit with this amount of throughput for 5 hours while the 69.30GB session raw data file was processed. Although the disk was under considerable load, the total load on the system was within acceptable limits.

The second throughput tested was 2,300 object impressions per second. The sensor was hit with this amount of throughput for 5 hours while the 72.45GB session raw data file was processed. In comparison with the previous run, the number of requests on the disk in service or waiting for service was higher. There was also an increase in the average wait time per request. Even with the increased load on the disk, the total load on the system was within acceptable limits. All session and visitor jobs were completed in 3 hours and 50 minutes, the longest of which was the `SessionMerger` job, completing in 2 hours and 31 minutes.

### Summary

The maximum throughput supported by this Analytics environment with three Hadoop nodes is 2,300 object impressions per second. The average maximum supported throughput *per node* from the previous scenario is 800 object impressions per second. The addition of a third Hadoop node increased the maximum supported throughput by 700 object impressions per second. This shows a near-linear relationship between adding a Hadoop node and increasing the maximum supported throughput. This test again showed the disk on the Hadoop nodes to be the bottleneck.

## Scenario 4: One Hadoop Node and Sensor on the Same System

Scenario 4 was designed to show the maximum throughput an Analytics environment with the Hadoop node and the sensor on the same system can support when running on the specified hardware (page 11).

### Data and Analysis

The first throughput tested was 600 object impressions per second. The sensor was hit with this amount of throughput for 5 hours while the 18.90 GB session raw data file was processed. The load on the disk surpassed the acceptable limit. We concluded that 600 object impressions per second was more throughput than the Analytics environment with the Hadoop node and sensor on the same system could support.

The second throughput tested was 500 object impressions per second. The sensor was hit with this amount of throughput for 5 hours while the 15.75 GB session raw data file was processed. Although the disk was under considerable load, the total load on the system was within acceptable limits.

### Summary

The maximum throughput supported by this Analytics environment with the Hadoop node and sensor on the same system is 500 object impressions per second. Addition of the sensor to the Hadoop node system reduced the maximum supported throughput by 29%, compared to an environment where the Hadoop node and sensor are on separate systems. The sensor consumes a sizeable amount of memory and adds more load to the disk, which is already a bottleneck to the environment.

## Conclusions

The following table summarizes the results of our Analytics 2.5 performance tests:

| Scenario | Maximum Throughput, OI/s | Scaling Factor Relative to Scenario 1 |
|---|---|---|
| 1. Analytics with one Hadoop node | 700 | 1.0 |
| 2. Analytics with two Hadoop nodes | 1,600 | 2.3 |
| 3. Analytics with three Hadoop nodes | 2,300 | 3.3 |
| 4. Single-server (hosting one sensor and one Hadoop node) | 500 | 0.71 |