
Addendum: iorate v3.x New Features and Usage

This section details new functionality added in iorate versions 3.07 through 3.13. These features extend the core functionality described in the legacy User Guide.

1. I/O Dashboard and Reporting

Prior to version 3.10, iorate only provided feedback via the log file and the final .perf file. Version 3.x introduces real-time feedback and easier-to-parse summaries.

Real-Time Console Dashboard

By default, iorate now displays a real-time dashboard in the terminal while tests are running. This dashboard updates once per second and displays:

- Current CPU Utilization.
- Read/Write IOPS.
- Read/Write Bandwidth (automatically scaling between MB/s and GB/s).
- Current Average Latency.

To suppress this output (useful for cron jobs or background execution), use the flag:
--disable-io-dashboard

Summary CSV Output

In addition to the .perf file (which details every device individually), iorate now produces a summary CSV file (default: iorate.csv). This file aggregates performance across all active devices for every second of the test run.

- **Columns:** Time, Read/s, KB_Read/s, Avg_Read_Resp_ms, Writes/s, KB_Write/s, Avg_Write_Resp_ms, Total_IOPS, Total_KB_Sec, Avg_Total_Resp_ms.

Direct I/O Changes

As of version 3.10, **Direct I/O is enabled by default** (equivalent to the old -u flag). This bypasses the operating system's file system cache to measure raw device performance.
To use Buffered I/O (OS Cache), you must explicitly use:
--no-direct-io

2. Thread Scaling Automation

Version 3.07 introduced features to automate the testing of scalability without manually editing the devices.ior file to change thread counts.

Overriding Thread Counts

You can ignore the count parameter specified in devices.ior and force a global thread count for all devices using:
--threads=<n>
Example: ./iorate --threads=8 will run 8 threads per device, regardless of what is in the config file.

Automated Scaling Runs

You can configure iorate to run a test suite multiple times, automatically increasing the load (thread count) each time.

- --scale_threads_by=<n>: The number of threads to add to the base count for each subsequent run.
- --scale_threads_count=<n>: The number of additional runs to perform.

Example:
./iorate --threads=1 --scale_threads_by=4 --scale_threads_count=2

1. **Run 1:** Uses 1 thread per device (Base).
2. **Run 2:** Uses 5 threads per device (1 Base + 4 Scaled).
3. **Run 3:** Uses 9 threads per device (1 Base + 8 Scaled).

3. Distributed / Networked Testing

As of version 3.13, iorate supports a Server/Client architecture. This allows a central "Report Host" to manage configuration files and aggregate results from multiple "Worker Clients."

Setting up the Report Host (Server)

The Report Host acts as the central aggregator. It serves configuration files to clients and collects their CSV reports.
Command:
./iorate --listen-as-report-host

File Preparation on Server:
Place the following files in the directory where you run the Report Host:

1. tests.ior: The global test sequence sent to all clients.
2. patterns.ior: The global I/O patterns sent to all clients.
3. devices.ior: (Optional) A default device list.
4. **Per-Host Device Files:** (Recommended) You can create specific device files for specific clients named <hostname>.devices.ior (e.g., db-server-01.devices.ior).

Setting up Worker Clients

Clients connect to the Report Host to download their configuration and upload their results.

Command:

./iorate --report-host-name=<SERVER_IP_OR_NAME> --retrieve-test-files

Advanced Client Options:

- --retrieve-test-files: The client will download patterns.ior and tests.ior from the server. It will also attempt to download a host-specific device file (e.g., myhostname.devices.ior). If the server does not have a host-specific file, the client will use its local devices.ior.
- --stay-up-after-all-runs: After the test cycle completes, the client will not exit. It will remain running, sleeping and polling the server for new instructions/configurations. This allows for continuous testing without restarting binaries on client machines.

Network Workflow

1. **Config Fetch:** When the client starts, it requests config from the Report Host.
2. **Device Config Logic:**
    - If the Report Host has <client_hostname>.devices.ior, it sends it. The client uses this file.
    - If the Report Host does *not* have that file, the client uses its **local** devices.ior file.
3. **Live Reporting:** During the run, clients send UDP packets to the Report Host, allowing the Server to display a "Combined Hosts Stats" dashboard in real-time.
4. **Result Aggregation:** When a test run completes, the client uploads its CSV report to the Report Host. The server saves this as <client_hostname>.iorate.csv.

4. Configuration File Updates

- **Reuse Percentage:** The reuse parameter in patterns.ior now supports the full range of **1% to 100%**. (Previously restricted to 10%-99%).
- **Max Volume Size:** Supported addressing has been increased from 4TB to **96TB**.

5. Updated Command Line Reference (v3.13)

Standard Options:

- -f <file>: Specify device file (Default: devices.ior)
- -p <file>: Specify pattern file (Default: patterns.ior)
- -t <file>: Specify test file (Default: tests.ior)
- -o <base>: Base name for output files (Default: iorate)
- -r <rate>: Scale target IOPS to <rate>%
- -u: Use Direct I/O (Default behavior in 3.x)

New/Advanced Options:

- --no-direct-io: Disable Direct I/O (use Buffered I/O).
- --disable-io-dashboard: Turn off the real-time terminal display.
- --threads=<n>: Force specific thread count per device.
- --scale_threads_by=<n>: Threads to add per scaling run.
- --scale_threads_count=<n>: Number of scaling runs.
- --listen-as-report-host: Run as Aggregator/Config Server.
- --report-host-name=<host>: Connect to specific Aggregator.
- --retrieve-test-files: Download config (tests.ior, patterns.ior, *.devices.ior) from Server.
- --stay-up-after-all-runs: Client loops indefinitely waiting for new runs.