Reclaimed Space Monitor — User Guide

Version: 2.9.1
Script filename example: *monitor_reclaimed.sh*

Overview

This script provides a real-time, terminal view of reclaimed storage bytes from the QoreStor ctrlrpc cleaner status. It polls on an interval, shows per-group totals (LOCAL, CLOUD, GRAND), deltas, average and instant rates, and sparkline trends. A live countdown, compact header, alerts, and optional CSV logging are included.

What You'll See

Header

- Title with start and last updated (pulses each refresh).

- Status strip showing RUNNING/PAUSED and baseline time.

- Help strip (toggled by 'h') and the current refresh interval.

Per-Group Sections

- Totals, cumulative delta since baseline, and step delta since last refresh.

- LOCAL/CLOUD show cleaner period start, average rate, and instant (last-interval) rate.

- Colourised sparklines of recent rates; height is EMA; colour is trend.

Footer

- Optional divider and bottom countdown to the next refresh.

Requirements

- Linux shell with Bash 4+

- Access to ctrlrpc output:

```
ctrlrpc -p 9911 show.cleaner_status
```

- awk, date, tput, Unicode/ANSI-capable terminal

Installation

1) Save the script to a file, e.g.: monitor_reclaimed.sh

2) Make it executable:

```
chmod +x monitor_reclaimed.sh
```

3) Run it from a terminal:

```
./monitor_reclaimed.sh
```

Running & Options

```
--csv[=FILE]          Enable CSV logging (optional file path)
--interval, -i SECS   Refresh interval seconds (default 3600)
--col N               Column for reclaimed bytes (default 13)
--skip N              Lines to skip after 'Cleaner summary' (default 12)
--ts-col N            Timestamp column (default 4)
--spark-width=N       Force spark width (0=auto)
--spark-points=N      Spark buffer length (default 30)
--no-spark            Disable sparklines
--debug               Verbose parser debug
```

Live Controls (Keys)

+ Faster (−5m)

− Slower (+5m)

r Manual refresh

p Pause/resume (timer)

b Baseline now (resets cumulative deltas)

h Toggle help strip

c Toggle countdown line

f Toggle divider bar

q Quit (restores terminal state)

Sparklines & Colour Logic

Each block is the EMA of the instant reclaim rate (B/h) over the last interval. Height is scaled to the window's min–max. The block's colour shows trend vs the previous block with a deadband:

- Green = up beyond deadband
- Red = down beyond deadband
- Grey = within deadband

The first visible block is neutral (grey).

Deadband tuning (environment variables):

```
COLOUR_DEADBAND_PCT=2        # relative % of previous sample
COLOUR_DEADBAND_ABS_BPH=0    # absolute floor in B/h (use max of the two)
```

Period Line Colouring

The period line (avg + inst) inherits the colour of the newest spark block so the text matches the current trend. Disable with FOLLOW_SPARK_TEXT=0.

Baseline Behavior

- Auto-baseline on first run and if totals shrink (window reset).
- Press 'b' to set a manual baseline; timestamp appears in the status strip.
- Cumulative deltas are computed since the active baseline.

Countdown & Footer

- Bottom countdown shows time to next refresh (Green/Yellow/Red thresholds).
- Toggle countdown with 'c' and divider with 'f'.
- Interval changes via +/- update the header immediately without forcing a refresh.

CSV Logging

Enable with --csv or --csv=FILE; format:

```
timestamp,LOCAL_bytes,CLOUD_bytes,GRAND_bytes
```

Alerts

If the raw instant rate exceeds ALERT_RATE_BPH (default ~32 GiB/h) an alert line prints and the terminal bell is triggered. Toggle with ALERTS_ON=0.

Environment Configuration (selected)

```
INTERVAL=3600           # clamp 300..7200
INTERVAL_STEP=300       # +/- step
```

```
SPARK_ON=1                 # 0 to disable
SPARK_POINTS=30            # history length
SPARK_WIDTH=0              # 0=auto
FOLLOW_SPARK_TEXT=1        # 0=plain text
EMA_NUM=2 EMA_DEN=5        # smoothing
MIN_SAMPLE_SECS=60         # ignore short zero-byte samples
MIN_VISIBLE_BPH=1          # floor for tiny but nonzero
SHOW_COUNTDOWN=1           # bottom timer
COUNTDOWN_OFFSET=2         # rows from bottom
SHOW_FOOTER_BAR=1          # divider above timer
UNICODE_BORDERS=0          # ASCII or Unicode
```

Troubleshooting

No colours/odd glyphs → ensure UTF-8 font/locale; try UNICODE_BORDERS=0.

‘invalid octal number’ → script normalizes timestamps; use GNU date/awk if possible.

First block red → seeding logic sets the first block grey; ensure you’re on the latest script.

Flicker → countdown is throttled; press 'c' to hide if needed.

Examples

```
./monitor_reclaimed.sh -i 300 --csv
```

```
SPARK_WIDTH=60 SPARK_POINTS=20 ./monitor_reclaimed.sh
```

```
ALERT_RATE_BPH=$((64*1024*1024*1024)) ./monitor_reclaimed.sh
```