Fridays are for artifacts

Decode. Analyze. Understand.

Cheatsheet CLI – commands Apple Unified Log

This is a concise manual for the macOS log command-line interface (CLI), which allows users to collect, search, filter, and analyze system logs. It provides essential commands, options, and examples for effective forensic and diagnostic use.

Usage:

log <command>

Global options:
-?, --help Show help information
-q, --quiet Reduce output verbosity
-v, --verbose Enable verbose output (useful for debugging or forensic analysis)

Commands:

1. collect
Collect system logs into a `.logarchive` file
Example:
log collect --output /path_to/filename.logarchive

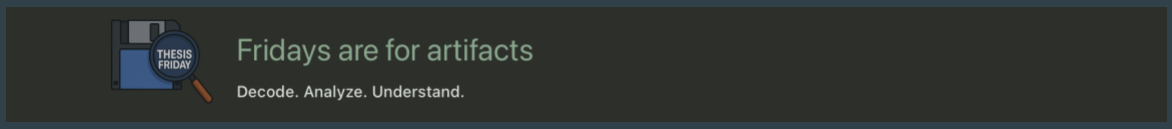
2. config
View or change logging system settings
Example:
log config --status

3. erase
Delete system logging data (only works with SIP disabled or in Recovery Mode)
Example:
log erase --all

4. repack
Repack a `.logarchive` file using a predicate
Example:
log repack system_logs.logarchive --predicate 'process == "loginwindow"' --output filtered_logs.logarchive

5. show (see also log show and log show –predicate)
View or search logs
Example:
log show --predicate 'eventMessage contains "login"' --info --style syslog

6. stream
Stream real-time logs
Example:
log stream --predicate 'subsystem == "com.apple.loginwindow"' --style syslog



Options for `log collect`:

--device : Collect logs from the first found device
--device-name <name> : Collect logs from device with given name
--device-udid <UDID> : Collect logs from device with specific UDID
--last <num>[m|h|d] : Collect logs from the past X minutes, hours, or days
--output <path> : Save the logarchive to the specified path
--size <num>[k|m] : Limit the size of collected logs
--start <time> : Collect logs from the specified time
--predicate <predicate> : Filter collected logs using the given predicate

Notes:

- If no output path is specified, `system_logs.logarchive` will be created in the current directory.
- If the output path is a directory, the file `system_logs.logarchive` will be created inside.
- A path ending in `.logarchive` will result in a file of that name.
- Using a predicate may impact performance and memory usage.

Valid time formats:

- 'Y-M-D H:m:s+zzzz' (e.g., 2025-06-15 14:00:00+0200)
- 'Y-M-D H:m:s' (e.g., 2025-06-15 14:00:00)
- 'Y-M-D' (e.g., 2025-06-15)
- '@unixtime' (e.g., @1718452800)

Examples:

log collect --device --output /path_to/filename.logarchive
log collect --device-udid <UDID> --output /path_to/filename.logarchive
log collect --device-name "Ed’s iPhone" --last 1h
log collect --output ~/systemlogs.logarchive
log collect --start "2025-12-31" --output /path_to/filename.logarchive
log collect --start "2025-12-31 06:30:00" --output /path_to/filename.logarchive

Options for `log show`:

Show contents of a logarchive.
Only “default” level logs are shown unless --info and/or --debug is specified.

Usage:

log show [options] <archive>
or: log show [options]

Options:

--[no-]backtrace : Show or hide backtraces
--[no-]debug : Show or hide debug-level events
--[no-]info : Show or hide info-level events
--[no-]loss : Show or hide message loss events
--[no-]signpost : Show or hide signposts

--color <mode> : Color output mode (auto, always, none)
--end <date> : Show logs up to a specified end date
--last <num>[m|h|d] : Show recent logs for given timeframe
--[no-]pager : Use `less` to paginate output
--predicate <predicate> : Filter logs using NSPredicate format
--process <pid> | <name> : Filter by process ID or name
--source : Show source file and line number
--start <date> : Show logs from specified start date
--style <style> : Output style (default, syslog, json, ndjson, compact)
--timezone local | <tz> : Display timestamps in specific timezone (e.g., UTC, Europe/Amsterdam)
--mach-continuous-time : Show mach continuous time instead of walltime
--unreliable : Annotate output with reliability flags

Valid time formats:

'Y-M-D H:m:s+zzzz' (e.g., 2025-06-15 12:00:00+0200)
'Y-M-D H:m:s' (e.g., 2025-06-15 12:00:00)
'Y-M-D' (e.g., 2025-06-15)
'@unixtime' (e.g., @1718452800)

Example:

Export logarchive to JSON:
log show --archive /path_to/example.logarchive --start "2025-03-09 14:50:00" --info --style json > output.json

Valid predicate fields:
activityIdentifier, bootUUID, category, composedMessage, continuousNanosecondsSinceBoot, creatorActivityIdentifier, creatorProcessUniqueIdentifier, date, formatString, logType, machContinuousTimestamp, parentActivityIdentifier, process, processIdentifier, processImagePath, processImageUUID, sender, senderImageOffset, senderImagePath, senderImageUUID, signpostIdentifier, signpostScope, signpostType, size, subsystem, threadIdentifier, timeToLive, traceIdentifier, transitionActivityIdentifier, type

Valid event types:
activityCreateEvent, activityTransitionEvent, userActionEvent, traceEvent, logEvent, timesyncEvent, signpostEvent, lossEvent, stateEvent

Valid log types:
default, release, info, debug, error, fault

Valid signpost scopes:
thread, process, system

Valid signpost types:
event, begin, end

Valid comparison operators:
AND, &&, & : logical AND

OR, || : logical OR

NOT, ! : logical NOT

!=, <>, ==, = : equality test

<, >, <=, =<, >=, => : greater/less than

BEGINSWITH : begins with

CONTAINS : contains

ENDSWITH : ends with

LIKE : wildcard pattern (? and *)

MATCHES : regex-style match

Examples:
log show --predicate 'process == "log"'

Face ID unlock example:
log show --predicate 'subsystem == "com.apple.chrono:keybag" AND eventMessage CONTAINS "locking -> unlocked"' --info

Touch ID example:
log show --predicate 'eventMessage CONTAINS "kAppleBiometricFingerOnEvent" OR eventMessage CONTAINS "kAppleBiometricFingerOffEvent"' --info

Volume buttons example:
log show --predicate '(eventMessage CONTAINS "rawVolumeIncreasePress" OR eventMessage CONTAINS "rawVolumeDecreasePress") AND process == "SpringBoard"' --info

Tap-to-Wake example:
log show --predicate 'subsystem == "com.apple.BackBoard:TouchEvents" AND eventMessage CONTAINS "tapToWake"' --info

Orientation change example:
log show --predicate 'subsystem == "Orientation" AND eventMessage CONTAINS "Received orientation"' --info

Log TTL analysis using `log stats`

Usage:
log stats [options] --archive <path_to_logarchive>

or: log stats [options]

Description:

Calculate and display statistics for a log archive or the local log store.

Options:

--archive <archive>	: Path to the .logarchive file
--count <count> | all	: Limit output per section (default: 5)
--sort <sort-mode>	: Sort by event count or byte size
--last <num>[m|h|d] | boot	: Limit analysis to recent time or since boot
--start <date>	: Start analysis from given time
--end <date>	: End analysis at given time
--style human | json	: Output format
--[no-]pager	: Use pagination with `less`

Modes:

--overview	: Default full summary
--per-book	: Stats per logbook
--per-file	: Stats per file
--sender <sender>	: Filter stats by sender
--process <process>	: Filter stats by process name
--predicate <predicate>	: Filter stats using a predicate

Sort modes:

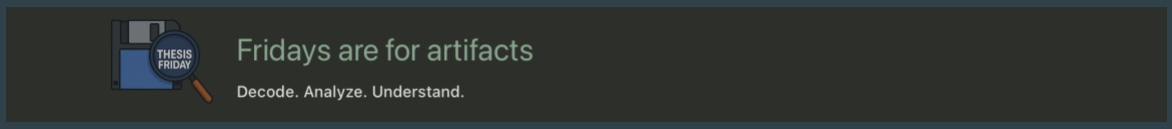
events	: Sort by number of log events
bytes	: Sort by byte volume

Valid time formats:

'Y-M-D H:m:s+zzzz'	e.g., 2025-03-09 14:50:00+0100
'Y-M-D H:m:s'	e.g., 2025-03-09 14:50:00
'Y-M-D'	e.g., 2025-03-09
'@unixtime'	e.g., @1719982200

Examples:
Calculate TTL from unlock to last known event:
log stats --archive ~/example.logarchive --start "2025-03-09 14:50:16" --end "2025-03-09 14:56:15" --style human --overview

Analyze log frequency after Face ID:
log stats --archive ~/aul_iphone12mini.logarchive --start "2025-03-09 14:50:43" --last 5m --style json



Show top log producers within TTL window:
log stats --archive ~/aul_iphone12mini.logarchive --start "2025-03-09 14:50:16" --end "2025-03-09 14:56:15" --sort bytes --count 10 --style human

For more insights, forensic tools, and advanced research on Apple system logging, visit

****[thesisfriday.com](https://www.thesisfriday.com)** — the platform for digital forensics, research and expert collaboration.**

Fridays are for artifacts – Decode Analyze Understand

Happy hunting!