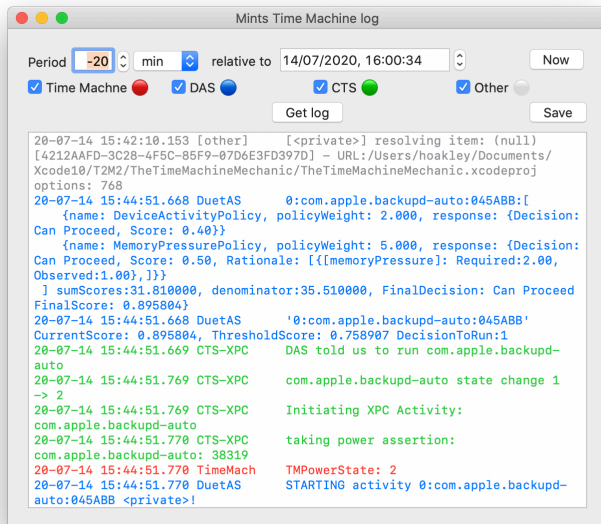


# Start

Mints is a growing collection of tools to provide deeper insight into macOS, to help you understand how its features work, and to help diagnose its problems. In this second beta-release, four tools are provided: these provide log extracts covering iCloud, TCC and privacy protection, and Time Machine backups, and look for Universal binaries.

Complex systems in macOS write dialogues to the log while they operate. Following these in a normal log extract can be difficult, because lines in each conversation are interspersed with other irrelevant entries. Mints' log browsers are different because they bring together entries from many different sub-systems and services, helping you to follow those dialogues. The use of colour to distinguish different sub-systems makes these conversations even more understandable.



→ [Start](#)

→ [Contents](#)

# Contents

→ [Main window](#)

→ [Log window controls](#)

→ [iCloud log](#)

→ [TCC log](#)

→ [Time Machine log](#)

→ [Universal Binary Checker](#)

→ [Updates](#)

→ [Change list](#)

→ [Further information](#)

# Main window



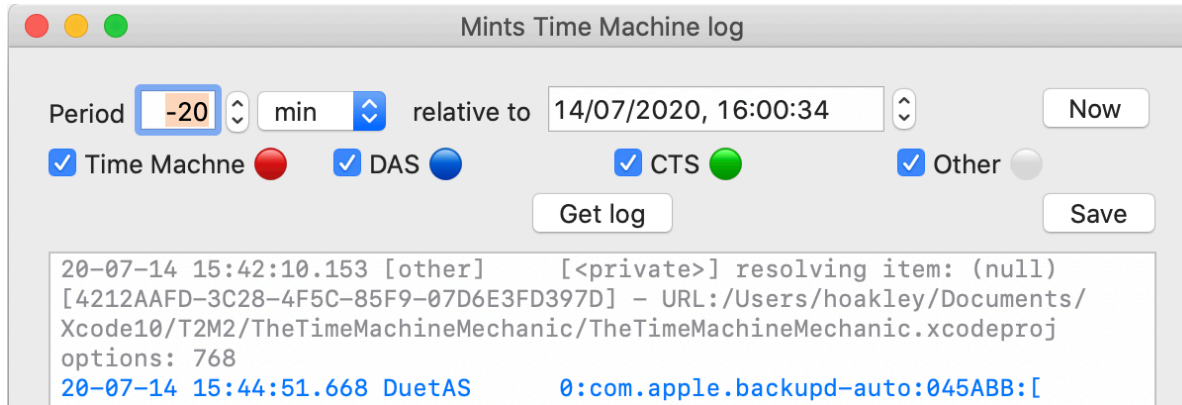
The main window currently offers just four buttons, which open:

- the → [iCloud Log Window](#), to explore iCloud messages in the log,
- the → [TCC Log Window](#), to explore TCC and privacy system messages in the log,
- the → [Time Machine Log Window](#), to explore Time Machine backup messages in the log,
- the → [Universal Binary Checker](#), to discover which apps and other code runs native on Apple Silicon Macs (*available in Mojave and later only*).

When you close this window, Mints will quit.

→ [Log window controls](#)

# Log window controls



Each log window in Mints offers a simple method to specify the precise period over which log entries are to be extracted. This is based on a reference time, entered in the **relative to** box using its standard controls, and a **Period** offset from that reference, set using its stepper arrows or by editing the text directly. You can also copy and paste times in the **relative to** box: simply select an item in the box and use the normal commands.

The offset entered in the **Period** box has a popup menu to determine its units. There is also a button **Now** which will set the reference time to the immediate present. Use that with a period of  $-20$  sec to get log entries over the last 20 seconds.

To get a log extract for the 1 minute *before* a reference time, simply set **Period** to  $-1$  and the popup menu to **min**. To obtain an extract for 10 seconds each side of a time in the past, the total period wanted is 20 seconds. You can either specify that using the reference time at the start of that period and 20 sec, or the time at the end of the period and  $-20$  sec.

→ [Log window controls \(concluded\)](#)

## Log window controls (*concluded*)

The shortest period which can be used here depends on the version of macOS. In Sierra, periods less than about 30 seconds are unreliable, and entries can be missed. In Mojave and Catalina, shorter periods can be used, even below 10 seconds if you wish.

Times in the **relative to** box are in the current time zone for the set date. On days when clocks change, they use the second of the two zones. For example, when the change to summer time occurs on a day, times entered here for that day use summer time throughout.

The row of controls below the time settings set which sub-systems are shown in the log view in the main part of the window. Each uses an emoji to indicate the colour in which those log entries are displayed.

The last controls are two buttons: **Get log**, which runs the log show command to fetch a fresh log extract, parse and display it, and **Save**, which saves what's currently displayed of the log in Rich Text format.

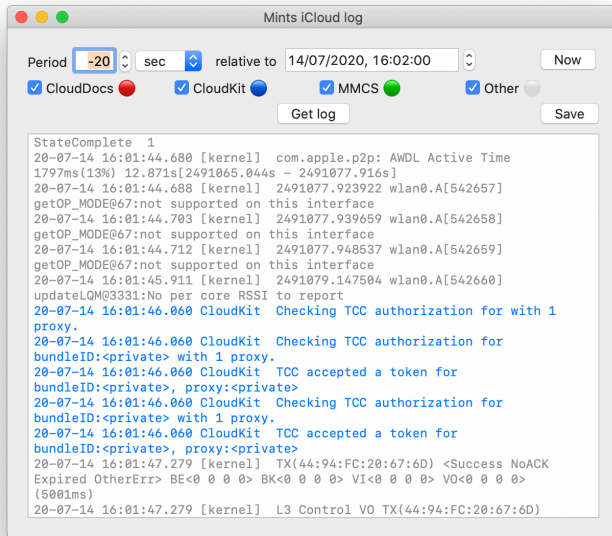
To enlarge the size of the text in any log extract, press ⌘+; to reduce the size, press ⌘- .

→ [iCloud log](#)

→ [Start](#)

→ [Contents](#)

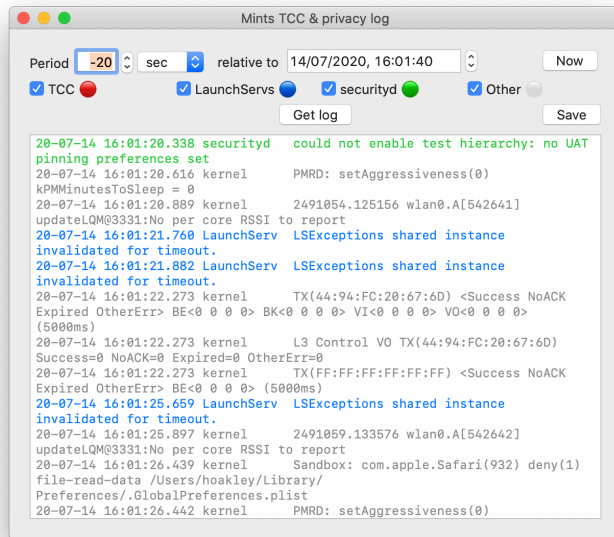
# iCloud log



The iCloud log extract shows entries from the three major sub-systems involved in iCloud transactions, and various related services and the kernel in the Other category. These are the same as shown in my free iCloud utility Cirrus.

→ [TCC log](#)

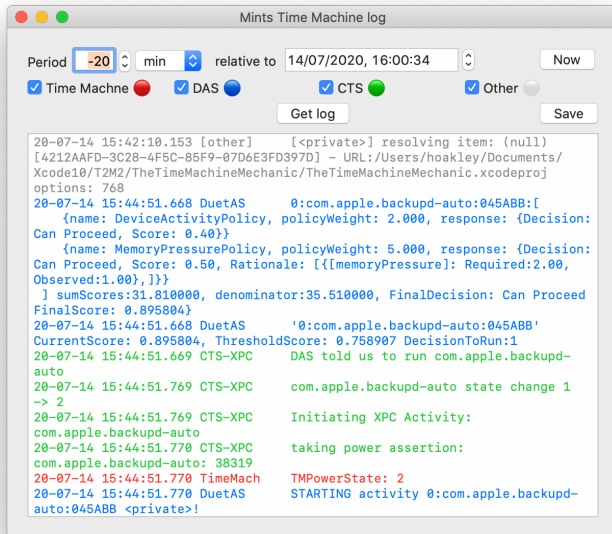
# TCC log



The TCC log extract shows entries from the three major sub-systems involved in controlling privacy and controlled access in Mojave and later. The Other category here includes additional services and the kernel. These are the same as shown in my free utility Taccy.

→ [Time Machine log](#)

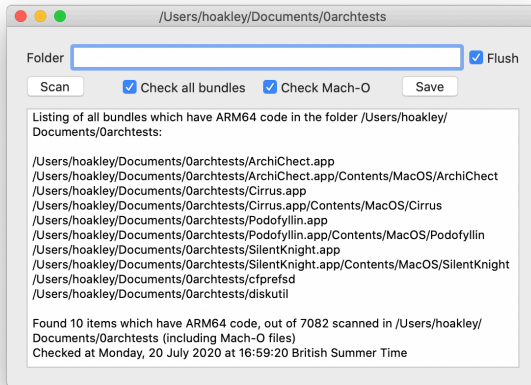
# Time Machine log



The Time Machine log extract shows entries from the three major sub-systems involved in scheduling, despatching and making Time Machine backups. The Other category here shows related services, but doesn't include the kernel. Although based on the extracts analysed by my free utility The Time Machine Mechanic, these extracts are new for Mints.



# Universal binary checker (Mojave and later)



Before scanning for Universal software, decide whether you just want apps scanned, or whether you want all code-containing bundles scanned. To scan apps only, leave the checkbox labelled **Check all bundles** unchecked; to scan all bundles, ensure it is ticked (the default).

Next, decide whether you want all executable (Mach-O) code files scanned too. These include dynamic libraries (dylibs), command tools, and other code which is not delivered in a bundle. To scan all executable code, ensure that the checkbox labelled **Check Mach-O** is ticked (the default); for a basic scan, leave that checkbox unchecked. Scanning all bundles and all Mach-O files will take significantly longer.

Then decide which folder you wish to scan. You can select that in either of two ways:

- leave the **Folder** text box empty, and click on the **Scan** button. You will then be prompted to select the folder using the normal Open File dialog;
- enter a standard path into the **Folder** text box. You can use the standard shortcut of ~ to indicate your Home folder. Once you have entered the path, click on the **Scan** button to start the scan. If the **Flush** checkbox is ticked when you do this, the path in the **Folder** text box will be cleared when the scan starts; if you uncheck the **Flush** checkbox, your path will be left for you to edit it for the next scan.

→ [Universal binary checker](#) (concluded)

## Universal binary checker *(concluded)*

Scanning takes time. If you put the root path / in the **Folder** text box, this can take many minutes, even an hour or so. You may wish to scan top-level folders like /Applications one at a time rather than attempting to scan your entire Mac. During this period, the window shows a busy ‘spinner’ next to the **Scan** button, and you can work normally with other apps. Give the app time to complete its scan, and once complete it will display the results.

Scanning takes plenty of memory. Because of the deep traversal which it uses, Mints consumes memory by the GB in order to check every nook and cranny in the folder it is scanning. This should work fine with virtual memory, though. Again, you can reduce this by performing scans on smaller folders.

When a scan is complete, the title of the window will change to show the folder which has just been scanned. Change the size of the text in the output using ⌘+ to make it larger, to a maximum of 60 points, or ⌘– to make it smaller, to a minimum of 4 points.

To save the results of a scan as a text file, click on the **Save** button. You will then be prompted to select the name of the text file.

The scanner performs a deep traversal of the chosen folder, using `Bundle.executableArchitectures` to list executable architectures for each bundle. To check for Mach-O code files, it first inspects the ‘magic’ initial four bytes of every file. If it finds that they are `0xfeedface`, it then calls `lipo` to check whether the supported architectures include ARM64.

# Updates

Whenever you open Mints, it may check to see if an update is available. This *doesn't* use the popular Sparkle mechanism for updating in place, but works as detailed here.

Once Mints has successfully completed its integrity check, it checks whether update checking has been turned off in its preferences file. If that has, it abandons any attempt to check for updates. If checking is allowed, it then checks when it last checked for updates. If that was more than 12 hours ago, it continues to perform the check. It then connects to my GitHub server, from where it downloads a list of current versions of my apps. It doesn't upload any data to the GitHub server at all, and no statistics beyond GitHub normal connection figures are collected either: no personal identifiers are recorded. If there is an update available, Mints then checks that its location is on this WordPress blog, and posts a dialog which invites you to download the update.

If you click on the **Download** button, it then points your default browser at that update, which should trigger the update to be downloaded to your normal downloads folder. The update is received as a regular Zip archive, and is exactly the same as you would download from the Downloads page here. It also carries a quarantine flag, so that when you unZip it and install the app inside, it undergoes normal first run 'Gatekeeper' security checks. If you click on the **Ignore** button, Mints won't remind you about it again for another 12 hours.

An additional item at the end of the **Help** menu explains the update status. If no update check is performed, or the check fails, the last item reads **Update not checked**. If the check is performed and update information is obtained, even when no update is available or you decline to download it, that menu item reads **Checked for update** and is ticked (but still disabled).

You can customise this behaviour by changing Mint's preferences. The keys to use are:

- `noUpdateCheck`, a Boolean. When set to `true`, this disables all update checking. Default is `false`.
- `updateCheckInt`, a real number (Double). When set to a value greater than 1.0, the minimum time interval between checks, in seconds. Default is 43200, which is 12 hours. If you set it to any value less than 1, Ulbow will reset it automatically to that default.

To change either of these, use a Terminal command of the form

```
defaults write co.electiclight.Mints updateCheckInt '10'
```

which works properly through the preferences server `cfprefsd`.

# Further Information

The monospace font using in versions of macOS up to and including Mojave is `monospacedDigitSystemFont`; that used in Catalina is `monospacedSystemFont`, which is unfortunately not available in earlier versions of macOS. Both are used in their regular weight.

Predicates used to obtain log extracts include:

- `subsystem == "com.apple.cloudDocs" OR subsystem == "com.apple.CloudKit" OR subsystem == "com.apple.mmcS" OR processImagePath CONTAINS[c] "cloudD" OR processImagePath CONTAINS[c] "bird" OR processID = 0 (for iCloud)`
- `subsystem == "com.apple.TCC" OR subsystem == "com.apple.LaunchServices" OR subsystem == "com.apple.SecurityD" OR subsystem == "com.apple.Sandbox" OR processImagePath CONTAINS[c] "tccd" OR processImagePath CONTAINS[c] "sandboxD" OR processID = 0 (for TCC)`
- `subsystem == "com.apple.TimeMachine" OR (subsystem == "com.apple.DuetActivityScheduler" AND eventMessage CONTAINS[c] "Rescoring all") OR (subsystem == "com.apple.xpc.activity" AND eventMessage CONTAINS[c] "com.apple.backup-auto") OR eventMessage CONTAINS[c] "backup" OR eventMessage CONTAINS[c] "Time Machine" OR eventMessage CONTAINS[c] "TimeMachine" (for Time Machine)`

Extensive information about Mints and the unified log is available from the [Eclectic Light Company blog](#), and the [product page](#) in particular. That can be accessed through the **Mints Support** command in the **Help** menu too.

## Change list

*1.0b4:*

- changed standard log text colours to black/white, red, blue, green
- added Universal Binary Checker for 10.14 and above.

### *1.0b3:*

- tried to fix further bug opening log windows in Sierra.

### *1.0b2:*

- fixed weird bugs in the log window nib.

### *1.0b1:*

- first beta release, with iCloud, TCC and Time Machine log browsers.

20 July 2020.