Collecting logs for REM

Contents

Introduction

This article explains how to capture Server side and Client side logs on Remote Expert Mobile, RE Mobile.

It applies to RE Mobile deployments with CUCM, UCCX and UCCE.

Versions: 10.6(x), 11.5(1), 11.6(1)

Server side logs

SSH on to the box

Login using the **rem-ssh** user and then switch users to the **root** user via:

```
su - root (enter password)
```

Log Capture Scripts

To capture a failure scenario you can use the logcapture script. If you are running in a HA environment it maybe easier to stop REAS2 and REMB2 and switch to a single REM and single MB to reduce the amount log logs to collect, then do:

On REAS (as root): `/opt/cisco/<version>/REAS/bin/logcapture.sh -c -p -v -z -f /root/reas-logs.tar`

On MB (as root): `/opt/cisco/<version>/CSDK/media_broker/logcapture.sh -c -p -v -z -f /root/mb-logs.tar`

After starting the above log capture, repeat the failure scenario, stop the log captures with Ctrl-C command and then send the resulting tar files for analysis.

Note: To copy the files you need to move them to /home/rem-ssh and set permission so the rem-ssh user can copy them e.g

```
mv /root/reas.tar /home/rem-ssh/
chown rem-ssh:rem-ssh /home/rem-ssh/reas.tar
```
(now the file can be copied off server)

To copy logs off the server use an SFTP client using the 'rem-ssh' user account and password.

Manual Server Log Retrieval

If for what ever reason the scripts are not appropriate, for example you want to look at some logs yourself or the issue is related to server start-up issues then below lists useful log locations:

```
/opt/cisco/<version>/REAS/domain/log/

/opt/cisco/<version>/REAS/domain/servers/appserver<host>/log/

/opt/cisco/<version>/CSDK/media_broker/

/opt/cisco/<version>/CSDK/media_broker/rtp-proxy-instances/mb-*/
```

If the service has been restarted you may be requested to tar up the log directories as archives so we can check through historical data.

Here's an example of the commands you would need:

```
/opt/cisco/<version>/REAS/domain/#tar cvfz reas-domain-logs.tar.gz log/
/opt/cisco/<version>/REAS/domain/servers/appserver-<host>/#tar cvfz
reas-server-logs.tar.gz log/
```

Which will produce:

```
/opt/cisco/<version>/REAS/domain/reas-domain-logs.tar.gz
/opt/cisco/<version>/REAS/domain/servers/appserver-<host>/reas-server-
logs.tar.gz
```

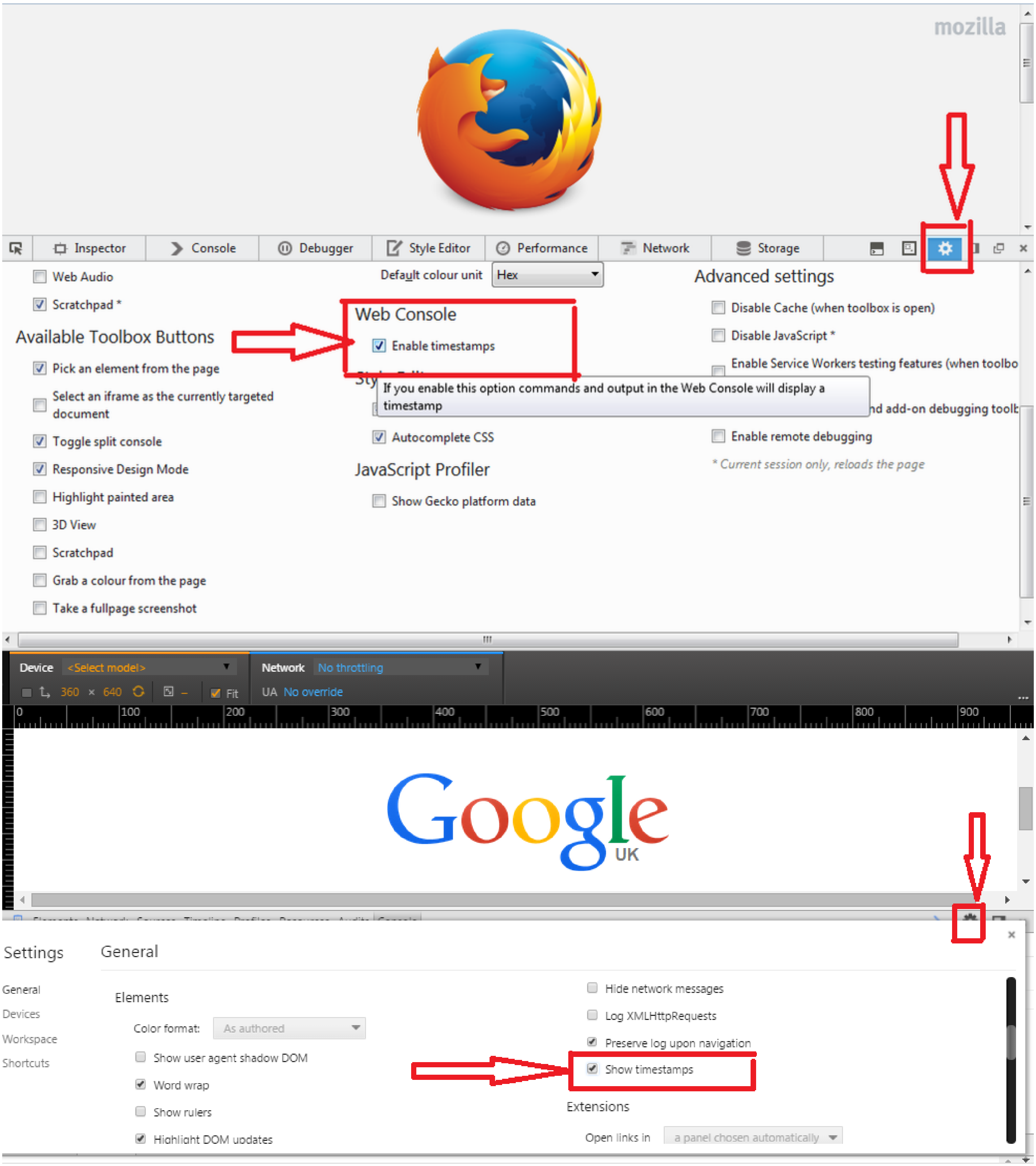
Client side Logs

Browser Console logs

Before obtaining browsers logs (sometimes also referred to as console/client logs) from Chrome or Firefox, could you please follow the instructions below to ensure that timestamps are enabled on your browser, these can be very useful to us when attempting to resolve an issue.

Enabling timestamps:

Applies to both Chrome and Firefox: Press F12 on your browser and select the "cog" settings button as highlighted in the the screenshot below, you will then see the menu illustrated below, ensure that the highlighted boxes are ticked.



☐ Web Audio

☑ Scratchpad *

Available Toolbox Buttons

☑ Pick an element from the page

☐ Select an iframe as the currently targeted document

☑ Toggle split console

☑ Responsive Design Mode

☐ Highlight painted area

☐ 3D View

☐ Scratchpad

☐ Grab a colour from the page

☐ Take a fullpage screenshot

Default colour unit: Hex

Web Console

☑ Enable timestamps

If you enable this option commands and output in the Web Console will display a timestamp

☑ Autocomplete CSS

JavaScript Profiler

☐ Show Gecko platform data

Advanced settings

☐ Disable Cache (when toolbox is open)

☐ Disable JavaScript *

☐ Enable Service Workers testing features (when toolbo

nd add-on debugging toolb

☐ Enable remote debugging

* Current session only, reloads the page

Settings

General

Devices

Workspace

Shortcuts

General

Elements

Color format: As authored

☐ Show user agent shadow DOM

☑ Word wrap

☐ Show rulers

☑ Highlight DOM updates

☐ Hide network messages

☐ Log XMLHttpRequests

☑ Preserve log upon navigation

☑ Show timestamps

Extensions

Open links in: a panel chosen automatically

- **Chrome**: If using Chrome for a client then you can see the logs in the console, shown by bringing up the page context menu (normally by right clicking in the browser window) and selecting "Inspect Element", and then selecting the Console tab. These logs are normally detailed enough for us.
  Note: Once the test is complete right click anywhere in the console window and select "save as" to save the logs to file.

  On some occasions we need more detailed logs that are obtained by opening a new instance of chrome in debug mode as follows:

Windows:
```
<Chrome Directory>\chrome.exe --enable-logging --v=9 --vmodule=*libjingle/source/*=9 --user-data-dir=c:\chromedebug
```
Mac:
```
/Applications/Google\ Chrome.app/Contents/MacOS/Google\ Chrome --enable-logging --v=9 --vmodule=*libjingle/source/*=9 --user-data-dir=/User/<your user dir>/chromedebug
```

- **Firefox**: Firefox console logs are accessed in a similar way to Chrome above. Right click within the firefox window and either select "Inspect Element" *or* after the right click just press Q. Once the window has appeared there will be a "Console" tab, the logs found here are again normally detailed enough for us.

- **Internet Explorer**: IE works the same as Firefox above without the shortcut of Q. To access the logs from IE simply right click, then click "Inspect Element". This will then bring up a set of windows, just click the "Console" tab. These logs should contain enough information for us to diagnose.

How to obtain the Browser Console Logs

1. Right click in your browser window, left click 'inspect' or 'inspect element'. This will bring up the developer tools
2. Click on the console tab
3. Recreate the issue you are seeing.
4. To save: Each console varies from browser to browser but generally you can: Right click (inside the console tab window) > Save as (notepad text file). Or Right Click > Select All, Right click > Copy. Then you'll want to paste that into a text file.
5. These text files tend to be small so you will be able to attach it directly to the ticket.

Collecting HAR network logs

The HTTP Archive format or **HAR**, is a JSON-formatted archive **file** format for logging of a web browser's interaction with a site.

How to generate a HAR on Chrome

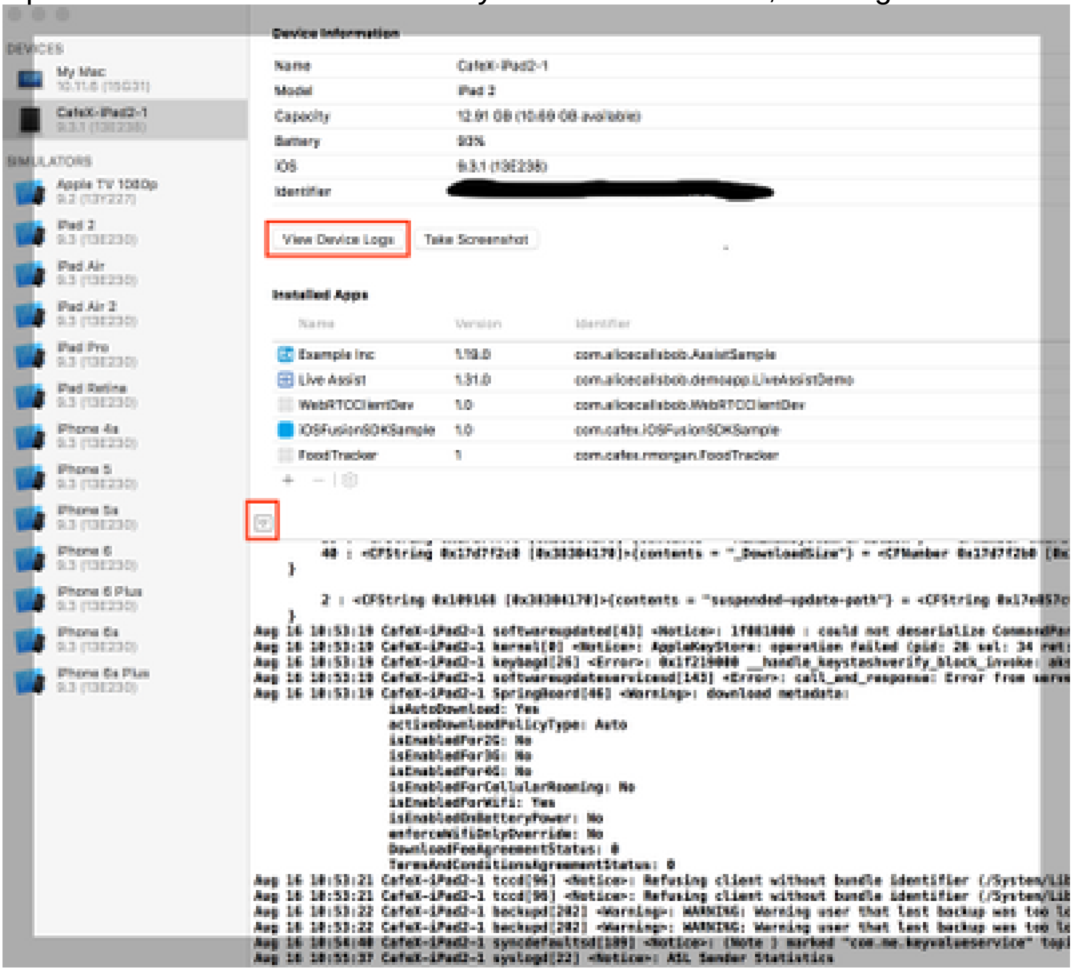
1. Open Google Chrome and go to the page where the issue is occurring.
2. From the Chrome menu bar select **View > Developer > Developer Tools** .
3. From the panel opens at the bottom of your screen, select the **Network** tab.
4. Look for a round Record button ( ) in the upper left corner of the Network tab, and make sure it is red. If it is grey, click it once to start recording.
5. Check the box next to **Preserve log** .
6. Click the Clear button ( ) to clear out any existing logs from the Network tab.
7. Recreate the issue you are seeing.
8. Once you have reproduced the issue, right click anywhere on the grid of network requests, select **Save as HAR with Content** , and save the file to your computer.

How to generate a HAR on Microsoft Edge

1. Browse to the URL where you are seeing the issue.
2. Navigate to Developer tools (use F12 as a shortcut) and select the "Network" tab.
3. Recreate the issue you are seeing.
4. Click on "Export as HAR" followed by Save As... to save the HAR file.

iPad/Android device logs

- **iPad:** We support using Xcode, here are the instructions on how to find the logs:
  You can obtain iPad logs by plugging the iPad into a MAC, opening XCode and navigating to **Window -> Devices** and selecting your iPad in the left hand menu. You can view the device console by clicking the arrow in the bottom left corner of the right hand panel. To get a copy of the logs 'select all' in the console window and paste into a text file.
  To get the Crash logs click the **View Device Logs** button and select the Crash item in the left hand pane from the time of the crash you are interested in, then right click and select **export log**.



Device Information

Name	CafeX-iPad2-1
Model	iPad 2
Capacity	12.91 GB (10.69 GB available)
Battery	93%
iOS	9.3.1 (13E238)
Identifier	[redacted]

Installed Apps

Name	Version	Identifier
Example Inc	1.19.0	com.alicecallsbob.AssistSample
Live Assist	1.31.0	com.alicecallsbob.demoapp.LiveAssistDemo
WebRTCClientDev	1.0	com.alicecallsbob.WebRTCClientDev
iOSFusionSDKSample	1.0	com.cafex.iOSFusionSDKSample
FoodTracker	1	com.cafex.rmorgan.FoodTracker

```
        40 : <CFString 0x17d7f2c0 [0x38384170]>{contents = "_DownloadSize"} = <CFNumber 0x17d7f2b0 [0x
    }

        2 : <CFString 0x1b9168 [0x38384170]>{contents = "suspended-update-path"} = <CFString 0x17e857c
    }
Aug 16 10:53:19 CafeX-iPad2-1 softwareupdated[43] <Notice>: 1f881000 : could not deserialize CommandPa
Aug 16 10:53:19 CafeX-iPad2-1 kernel[0] <Notice>: AppleKeyStore: operation failed (pid: 28 sel: 34 ret:
Aug 16 10:53:19 CafeX-iPad2-1 keybagd[26] <Error>: 0x1f219000 __handle_keystashverify_block_invoke: ak
Aug 16 10:53:19 CafeX-iPad2-1 softwareupdateservicesd[143] <Error>: call_and_response: Error from serve
Aug 16 10:53:19 CafeX-iPad2-1 SpringBoard[46] <Warning>: download metadata:
                    isAutoDownload: Yes
                    activeDownloadPolicyType: Auto
                    isEnabledFor2G: No
                    isEnabledFor3G: No
                    isEnabledFor4G: No
                    isEnabledForCellularRoaming: No
                    isEnabledForWifi: Yes
                    isEnabledOnBatteryPower: No
                    enforceWifiOnlyOverride: No
                    DownloadFeeAgreementStatus: 0
                    TermsAndConditionsAgreementStatus: 0
Aug 16 10:53:21 CafeX-iPad2-1 tccd[96] <Notice>: Refusing client without bundle identifier (/System/Lib
Aug 16 10:53:21 CafeX-iPad2-1 tccd[96] <Notice>: Refusing client without bundle identifier (/System/Lib
Aug 16 10:53:22 CafeX-iPad2-1 backupd[282] <Warning>: WARNING: Warning user that last backup was too lo
Aug 16 10:53:22 CafeX-iPad2-1 backupd[282] <Warning>: WARNING: Warning user that last backup was too lo
Aug 16 10:54:40 CafeX-iPad2-1 syncdefaultsd[189] <Notice>: (Note ) marked "com.me.keyvalueservice" topi
Aug 16 10:55:37 CafeX-iPad2-1 syslogd[22] <Notice>: ASL Sender Statistics
```

- For for older **Xcode 5 & 6** versions. You can obtain iPad logs by plugging the iPad into a MAC, opening XCode 5 and navigating to **Window -> Organiser -> Devices** and selecting your iPad in the left hand menu. In the drop down menu select console. For XCode 6 navigate to **Window -> Devices** and select your iPad in the left hand menu, when click the view device logs button.
- If you don't use XCode and your using iOS7 you can download the *iPhone Configuration Utility* for windows or Macs and connect your iPad to it to collect the console logs. For iOS8 you can download and collect logs via http://lemonjar.com/iosconsole/ if using a MAC.
- **dSYM file**. If your **custom iPad app** has crashed please include the dSYM file as well as the crash log. The dSYM file can be found in xCode by Right Clicking on your archive -> Show in Finder -> Right click on file and click on Show package content

- **Android:** There are a number of ways to obtain logs from an Android device.
    1. In the lab if using Android Studio use this guide https://developer.android.com/studio/debug/am-logcat.html
    2. If you use Eclipse then this tutorial will help https://www.utest.com/courses/android-debug-bridge-part-1-how-to-capture-logcat-files-using-adb (sign in required)
    3. Note: From Android 4.1 onwards installing a *Log capture* app onto the device will only give you access to the apps own logs unless the device is rooted.

Getting REAS WG Configuration

Providing the configuration for the cluster is also normally needed along with logs. The easiest way to provide this is to navigate to

https://REAS_MASTER:8443/admin/gateway/1.0/configuration

You need to provide the CSDK web_plugin_framework credentials. Then cut and paste the contents of the page into a text file and attach it to the ticket.

Note: Please don't use MS notepad as it corrupts the formatting of the page