

# View and Export Application Crash Reports

When an application crashes, a crash report is stored on the device. A crash report describes the conditions under which the application terminated, and in most cases includes a complete stack trace for each executing thread. Crash reports are a useful tool for debugging issues in an application.

If an app has the [Collect Crash Reports](#) policy applied, Apperian collects the crash report from the device, encrypts it, and stores it in the Apperian data base. On the Crash Reports tab of the app's details page, Apperian lists the crash reports it has collected for each version of the app. From that list, you can view reports and export `.crash` files to send to developers for further analysis.

iOS Crash reports with stack traces need to be **symbolicated** before they can be analyzed. Symbolication replaces memory addresses with human-readable function names and line numbers. For more information on iOS crash reports, including instructions on symbolicating a `.crash` file, see [Understanding and Analyzing iOS Application Crash Reports](#) in the iOS Developer Library.

## Before You Begin

Apply the Collect Crash Reports application policy to any iOS or Android app for which you want to collect crash reports in Apperian. If you are prompted to sign the app after you apply the policy, you will need to deploy a new version to your users before the policy will be effective on devices where the app is already installed. For instructions, see [Apply Policies to an App](#).

To view and export an app's crash reports

1. On the Apperian Portal navigation bar, click **Applications** to display the list of applications. Use the **Search** box to search for a particular application.
2. Click the application icon or name to display the Details page.
3. Click on the **Crash Reports** tab to display a table listing each crash report collected from devices running the app. The Crash Reports tab is available for any iOS or Android app with the Collect Crash Reports policy applied, even if there have not yet been any reports collected.

The screenshot shows the 'Crash Reports' tab for the application 'TransitDataImport'. The page includes a navigation bar with tabs for Details, Rating, Inspection, Policies, Signing, Approval Log, and Crash Reports. The 'Crash Reports' tab is active, displaying a table of collected crash reports. The table has the following columns: Date of Crash, App Version, OS Version, Device Model, and Export. The table lists six crash reports in reverse-chronological order. Red callouts highlight specific information: 'Number of crashes: 6', 'Number of reports collected', 'Date of last crash: 06/26/2014 - 8:33 AM', 'Date of most recent crash', 'Click to export the report to a .crash file.', 'Click a row to view the report.', and 'Version of the app that crashed.'

Date of Crash	App Version	OS Version	Device Model	Export
06/26/2014 - 8:33 AM	2.0	7.0.6	iPad2,2	⬇
06/26/2014 - 8:33 AM	2.0	7.0.6	iPad2,2	⬇
6/26/2014 - 8:28 AM	2.0	7.0.6	iPad2,2	⬇
6/26/2014 - 8:28 AM	1.0	7.0.6	iPad2,2	⬇
6/26/2014 - 8:28 AM	1.0	7.0.6	iPad2,2	⬇
06/26/2014 - 8:28 AM	1.0	7.0.6	iPad2,2	⬇

The table lists reports in reverse-chronological order. To view a particular report, click on the row. See the [example](#) report below. To export a report, click the arrow in the **Export** column and then click **OK** to save a `.crash` file to your Downloads folder. You can view the contents of the `.crash` file with various applications, including text editors, Apple Console, or Xcode.

**i** To symbolicate a `.crash` file, you will need both the application binary (`.ipa`) that was uploaded to Apperian and the `.dSYM` file that was generated when that binary was built. These must be an exact match with the version of the application that crashed, otherwise the report cannot be fully symbolicated. For instructions on symbolicating a `.crash` file, see [Understanding and Analyzing iOS Application Crash Reports](#) in the iOS Developer Library.

## Example Crash Report

The following example shows a crash report collected from an iOS application.

## Crash Report

Incident Identifier: 6A953B73-D933-4212-ACA9-3C7F2A3649EB  
CrashReporter Key: TODO  
Hardware Model: iPad2,2  
Process: TransitDataImpor [1839]  
Path: /var/mobile/Applications/F3942484-E653-4034-8515-66077B60A56/TransitDataImport.app/TransitDataImport  
Identifier: lo.objc.TransitDataImport  
Version: 1.0  
Code Type: ARM  
Parent Process: launchd [1]

Date/Time: 2014-03-23 15:58:51 +0000  
OS Version: iPhone OS 7.0.6 (11B651)  
Report Version: 104

Exception Type: SIGSEGV  
Exception Codes: SEGV\_ACCERR at 0xc  
Crashed Thread: 3

Thread 0:

0	libsystem_kernel.dylib	0x38e01a8c	0x38e01000 + 2700
1	CoreFoundation	0x2e0437c3	0x2dfa5000 + 649155
2	CoreFoundation	0x2e041f2f	0x2dfa5000 + 642863
3	CoreFoundation	0x2dfacc27	0x2dfa5000 + 31783
4	CoreFoundation	0x2dfaca0b	0x2dfa5000 + 31243
5	GraphicsServices	0x32c8d283	0x32c85000 + 33411
6	UIKit	0x30850049	0x307e0000 + 458825
7	TransitDataImport	0x000fc453	0xfb000 + 5203
8	libdyld.dylib	0x38d5dab7	0x38d5c000 + 6839

Thread 1:

0	libsystem_kernel.dylib	0x38e0183c	0x38e01000 + 2108
1	libdispatch.dylib	0x38d41f9b	0x38d37000 + 44955

Thread 2:

0	libsystem_kernel.dylib	0x38e14c7c	0x38e01000 + 81020
1	libsystem_pthread.dylib	0x38e7ac84	0x38e7a000 + 3204

Thread 3 Crashed:

0	TransitDataImport	0x000fda68	0xfb000 + 10856
1	Foundation	0x2e9aa767	0x2e98e000 + 116583
2	Foundation	0x2ea59cbf	0x2e98e000 + 834751
3	TransitDataImport	0x000fd885	0xfb000 + 10373
4	CoreData	0x2de547bd	0x2ddb3000 + 661437
5	libdispatch.dylib	0x38d38d07	0x38d37000 + 7431
6	libdispatch.dylib	0x38d3eb9f	0x38d37000 + 31647
7	CoreData	0x2de548fb	0x2ddb3000 + 661755
8	TransitDataImport	0x000fd6d5	0xfb000 + 9941
9	Foundation	0x2e9a77db	0x2e98e000 + 104411
10	Foundation	0x2ea4b995	0x2e98e000 + 776597
11	libdispatch.dylib	0x38d3e68f	0x38d37000 + 30351
12	libdispatch.dylib	0x38d3fd71	0x38d37000 + 36209
13	libdispatch.dylib	0x38d3ff59	0x38d37000 + 36697
14	libsystem_pthread.dylib	0x38e7adbf	0x38e7a000 + 3519
15	libsystem_pthread.dylib	0x38e7ac84	0x38e7a000 + 3204

This report includes a stack trace for each executing thread.