

Recovering Apple Safari History Binary PList (Property List) Files

- [Introduction](#)
- [MAC OS X](#)
- [Apple Safari Browser](#)
- [Creating a Blade Recovery Profile](#)
- [File Header Section](#)
- [File Landmark Section](#)
- [File Footer Section](#)

Introduction

In the Mac OS X Cocoa, NeXTSTEP, and GNUstep programming frameworks, property list files are files that store serialized objects. Property list files use the filename extension .plist, and are therefore often referred to as plist files. Property list files are often used to store a user's settings. They are also used to store information about bundles and applications, a task served by the resource fork in the old Mac OS.

MAC OS X

In [Mac OS X 10.0](#), an XML format plist was introduced, with a public DTD defined by Apple. The XML format supports non-ASCII characters and storing NSValue objects (which, unlike GNUstep's ASCII property list format, Apple's ASCII property list format does not support). Since XML files, however, are not the most space-efficient means of storage, [Mac OS X 10.2](#) introduced a new format where property list files are stored as binary files. Starting with [Mac OS X 10.4](#), this is the default format for preference files.

Apple Safari Browser

The Apple Safari browser stores a history of visits to web pages in a plist. These plists come in two flavours depending on which Safari version was present on the system. In older versions of the browser, the history is stored in an XML formatted property list which is reasonably easy to parse and recover from unallocated clusters. The individual history records are stored in dictionary objects which have a nice structure and are easy to recover using RBE (Record Based Extraction) techniques.

In newer releases of the browser (for example Safari v4 for Windows) the plist is stored in a proprietary binary plist structure which is more difficult to parse and recover from unallocated clusters. With the binary structure, recovery is only possible using FBE (File Based Extraction).

As NetAnalysis v1.50 (and above) can parse and read Safari binary plist data, we will create a Blade Recovery Profile to attempt to recover these files.

Creating a Blade Recovery Profile

In this example, we are going to create a Blade recovery profile to extract binary plist files. To create a profile, select Personal Profile Database from the Tools menu. Select Add New to create a blank profile (as shown in Figure 1).

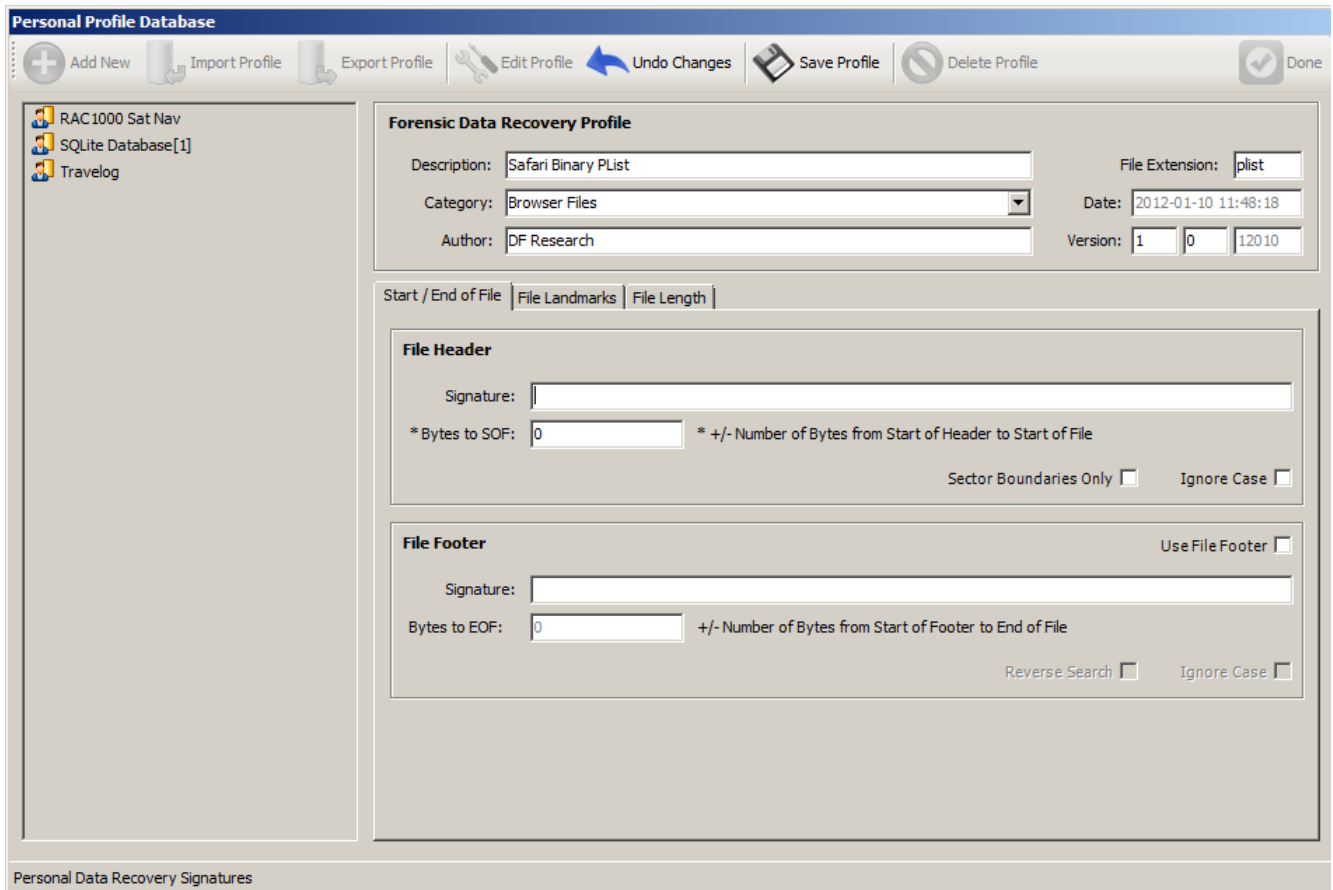


Figure 1

In the Description Field, we have entered “Safari Binary PList”, in the Category field we have entered “Browser Files” and in the Extension field we entered “plist”. The Author and version numbers are automatically generated for you.

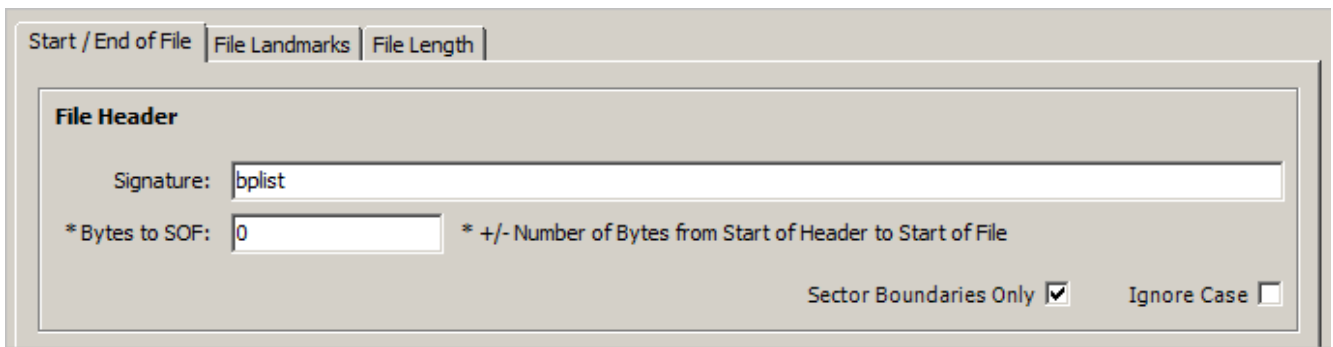
File Header Section

The next section to complete is File Header Section. The signature field holds the string regular expression which identifies that pattern of bytes at the start of a file (or segment of data). This is sometimes referred to as the “file signature” or “magic number”. This section contains information about the start of the file. To identify an appropriate file header signature, we will need to examine the structure of the binary plist. To do this, I have loaded a binary plist file into a hex viewer. Examination of the file (and with reference to the binary file specification) shows that the header contains the lower case string “bplist”. See Figure 2.

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
00000000	62	70	6C	69	73	74	30	30	D2	00	01	00	02	00	03	00	bplist000.....
00000016	0B	5F	10	0F	57	65	62	48	69	73	74	6F	72	79	44	61	...WebHistoryDa
00000032	74	65	73	5F	10	15	57	65	62	48	69	73	74	6F	72	79	tes...WebHistory
00000048	46	69	6C	65	56	65	73	73	69	6F	6E	AF	10	DE	00	04	FileVersion...P..
00000064	00	0D	00	11	00	15	00	19	00	1C	00	20	00	24	00	28\$. (
00000080	00	2C	00	30	00	34	00	38	00	3C	00	40	00	45	00	4A	...0.4.8.<.@.E.J
00000096	00	4F	00	53	00	57	00	5B	00	5E	00	62	00	65	00	69	.O.S.W.[.^b.e.i
00000112	00	60	00	71	00	75	00	79	00	7D	00	81	00	84	00	88	.m.q.u.y.}.I.I.I
00000128	00	8C	00	90	00	93	00	96	00	99	00	9D	00	A0	00	A3	.I.I.I.I.I.I.I.£
00000144	00	93	00	97	00	9A	00	9D	00	A0	00	A3	00	A6	00	A9	.I.I.I.I.I.I.I.£
00000160	00	93	00	97	00	9A	00	9D	00	A0	00	A3	00	A6	00	A9	...>>.¿.Ä
00000176	00	93	00	97	00	9A	00	9D	00	A0	00	A3	00	A6	00	A9	.I.Ö.x.Û.P.ä
00000192	01	08	01	0C	01	10	01	15	01	18	01	1B	01	1F	01	23	...i.ó.÷.ù.y..
00000208	01	26	01	2A	01	2D	01	31	01	34	01	38	01	3C	01	40#
00000224	01	44	01	48	01	4C	01	50	01	54	01	58	01	5C	01	60	&*.~.1.4.8.<.@
00000240	01	64	01	68	01	6C	01	70	01	74	01	78	01	7C	01	80	.D.H.L.P.T.X.\.´
00000256	01	84	01	88	01	8C	01	8F	01	93	01	97	01	9B	01	9F	.d.h.l.p.t.x. .I
00000272	01	A2	01	A7	01	AB	01	AF	01	B3	01	B7	01	BB	01	BF	.I.I.I.I.I.I.I.I

Figure 2

With this information, we can now enter the signature “bplist” for the start of the file as shown in Figure 3.



The screenshot shows a software interface with three tabs: "Start / End of File", "File Landmarks", and "File Length". The "File Landmarks" tab is active. Inside this tab, there is a section titled "File Header". It contains a text input field for "Signature:" with the value "bplist". Below it is another input field for "* Bytes to SOF:" with the value "0". To the right of this field is a note: "* +/- Number of Bytes from Start of Header to Start of File". At the bottom right of the "File Header" section, there are two checkboxes: "Sector Boundaries Only" which is checked, and "Ignore Case" which is unchecked.

Figure 3

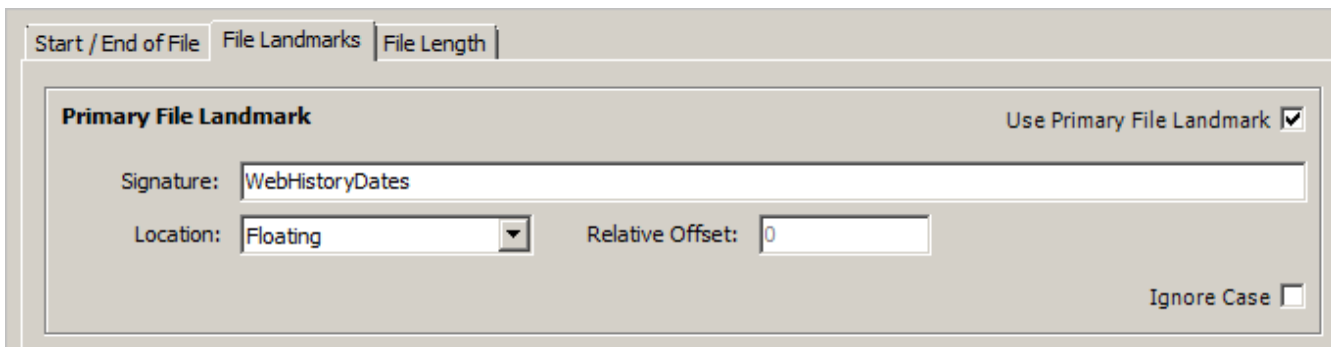
As we only want to recover plists which were originally history files, and not binary plist data embedded in other files, we will check “Sector Boundaries Only”. The signature is also case sensitive as we only want to recover “bplist” when all the characters are lower case, so the “Ignore Case” option is left unchecked.

File Landmark Section

The file landmark section allows you to improve the recovery capability even further. If you think of the file header and footers as bookends, the file landmark section refers to any data which can be found within the two boundaries. The landmark can be found at a specific offset, or at any position within the file. The landmark also uses regular expression patterns, and you can also select Unicode data.

Examination of the Safari History plist shows that the history records are stored in a dictionary where the key is “WebHistoryDates” and the data stored inside an array of dictionaries inside this object. The string “WebHistoryDates” can therefore be used as a Landmark within the file.

Check the “Use File Landmark” option in the File Landmark section, and enter the text “WebHistoryDates” in the Signature field. This string is also case sensitive so leave “Ignore Case” unchecked. In this case, as the exact location of the landmark can change, we will leave the Location field set to “Floating”. See Figure 4.



The screenshot shows the same software interface as Figure 3, but now the "File Landmarks" tab is active. It features a section titled "Primary File Landmark". At the top right of this section is a checkbox labeled "Use Primary File Landmark" which is checked. Below this, there is a text input field for "Signature:" containing the text "WebHistoryDates". Underneath, there are two more input fields: "Location:" with a dropdown menu set to "Floating", and "Relative Offset:" with a text input field containing "0". At the bottom right of the "Primary File Landmark" section, there is an unchecked checkbox for "Ignore Case".

Figure 4

File Footer Section

The file footer section contains information to allow the end of the file to be found. By selecting the Use File Footer check box, the file footer fields will be activated. As with the other signature sections, you have the ability to use a regular expression pattern for this field.

In the case of binary plist files, there is no recognised footer, so we will need to devise a way to identify the end of the file. Examination of the binary plist structure shows that it does have a standard structure for the end of the file. The trailer structure is shown below in Figure 5.

