Searching for Yahoo Chat fragments in Unallocated Space

Detective Eric Oldenburg, Phoenix Police Department

Purpose and Goal

To demonstrate a methodology used for locating Yahoo Instant Messenger chat fragments that reside in unallocated space. This can be used where Yahoo IM installations do not contain valid archive (.dat) files to be parsed. This methodology will *significantly* reduce false hits.

What this methodology WILL do:

Using a GREP expression, locate chat fragments and decode them into readable text, with the following information:

Date/Time
Whether message was sent or received
Message content

What this methodology WILL NOT do:

Recover the user name of the remote user. This information is normally retrieved from the directory name that the chat archive files are written to when logging is enabled. This information is not contained within the archive log file itself, and therefore can not be determined from the data recovered in unallocated space.

Just because the remote user name is not retrieved, does not mean that the information is useless. Information about the remote user can always be gleaned from the content of the chat itself. This can also be a great way to corroborate an undercover chat (or chat where the victim computer is accessible) with a suspect where the suspect machine does not contain valid chat log files. Chat from the U/C officer (or victim) can be compared with chat fragments found on the suspect machine.

NOTE: This methodology uses EnCase version 6.8 and YAHOOM.exe (a chat archive decoder). Although these are the programs used, this methodology can be applied to any other programs that perform similar functions.

Introduction

The author was given computer evidence (running Windows XP) in an investigation of a child sex crimes case. The suspect had reportedly been using Yahoo Instant Messenger to chat with his 16 year old future step daughter. According to the victim, the suspect chatted with her in a sexual manner. The suspect admitted to chatting with the victim, using Yahoo Instant Messenger, but not in a sexual manner. The suspect gave consent to search his computer.

The victim alleged sexual intercourse with the suspect who denied it. The only corroborating evidence in this case are possible chat logs that would verify the victim's statements.

Chat logging was not enabled, so it was necessary to search in unallocated space for the chat logs.

Using GREP searches in EnCase, I was able to locate and subsequently reconstruct several valid chat sessions that were found in unallocated space.

Investigative Issues

Issue 1: The default installation configuration for Yahoo! Instant Messenger (current version 8, please verify earlier versions) is to archive chats but delete them upon log off, leaving chat logs in unallocated space.

Issue 2: Yahoo! IM dat files have no consistent header, making them difficult to search for.

Issue 3: The chat logs are XOR encrypted, so clear text searches for user names will not work.

Investigative Process

1. Verify the presence of Yahoo Instant Messenger on the computer.
2. Locate screen names used. Necessary for decryption of chat fragments.
3. Determine whether or not logging in enabled.
4. Search for any chat fragments in unallocated space.
5. Create readable .dat files
6. Decode encrypted chat fragments.

Methodology

Verify the presence of Yahoo Instant Messenger on the computer.

This process is typically straight forward. The default location of Yahoo IM is:

C:\Program Files\Yahoo!\Messenger.

Locate screen names used.

This step is important because we will need the local user name to decrypt the chat fragments properly. We can find the user names that were used to chat on Yahoo in:

C:\Program Files\Yahoo!\Messenger\Profiles

This typically contains a directory for each user name used.

The registry also contains a list of profiles associated with Yahoo! IM. It also contains some valuable information about amount of login attempts per user name, etc. This information found in the NTUSER.DAT file and will contain different data depending on which NTUSER file is used. They are located in the key:

HKEY_CURRENT_USER\Software\Yahoo\Pager\profiles

Determine whether or not logging in enabled.

Presence of logging can be located in the key:

HKEY_CURRENT_USER\Software\Yahoo\Pager\profiles\USERNAME\Archive

This contains keys called “Enabled” and “AutoDelete”. If they are flagged with values of 0x00000001, then archives are kept but deleted when the user exits Yahoo IM. If the values are 0x00000000, then logging is not enabled. (this may or may not indicate the presence of chat fragment in unallocated space.

Search for any chat fragments in unallocated space.

This is where we get our hands dirty. Roll up your sleeves and lets tackle the problem.

It is necessary to understand the structure of the messages as they are written to the disk. If you have a valid chat archive .dat file, using an hex editor, you can see the basic structure of the chats.

Each message contains the following fields in the following order:

UNIX DATE	TYPE VALUE	USER VALUE	MESSAGE LENGTH	MESSAGE	TERMINATOR
4 BYTES	4 BYTES	4 BYTES	4 BYTES	VARIABLE	4 BYTES

UNIX DATE: This is a 4 byte value that is the amount of seconds that have elapsed since 1/1/1970.

NOTE: It is important to known what year or range of years you are searching for messages from. Use a hex converter to convert dates to UNIX hex values. The last byte is the most significant, so we will only include this byte in the search. For example: between 10/01/2006 and the present will be either 0x45 0x46 or 0x47

TYPE VALUE: Searchable messages have the 4 byte value 06 00 00 00.

USER VALUE: This 4 byte value indicates whether or not the message was sent by the local user or received by them. the value is either 01 00 00 00 (received) or 00 00 00 00 (sent).

MESSAGE LENGTH: This is the hex value of the amount of bytes the message takes up. This will be variable, however, because instant messages are usually shorter than 255 characters, we can assume that the 4 byte value will be XX 00 00 00, with XX being any hex character.

TERMINATOR: This is a machine sent from the future to kill John Connor. Actually, its a 4 byte value that indicates the end of the message. It is always 00 00 00 00.

Yahoo IM chat archives do not have a standard header so it is necessary to search for common byte values in all messages. The common byte values we will use to build our GREP expression will be:

Date	(Think of the most significant byte, the last one)
Type Value	(Is it either a 00 00 00 00 (sent) or 01 00 00 00 (received))
Message Length	(first byte can be anything followed by 00 00 00, assuming the message is at least one character long.)

So, our search will be built in this manner:

DATE (1 BYTE) - \x45|\x46|\x47 - 0x45 0x46 or 0x47

(this example covers date range of ~ September 2006 to the present)

TYPE VALUE (4 BYTES) - \x06\x00{3,3} – 0x06 followed by 0x00 3 times

USER VALUE (4 BYTES) - \x01|\x00\x00{3,3} – 0x01 or 0x00 followed by 0x00 3 times

LENGTH (4 BYTE) – [^x\00]\x00{3,3} – any hex character followed by 0x00 3 times

Because of the variation of data beyond these byte values, we will not include them in the GREP search, rather we will visually inspect them when viewing our search hits.

Our complete GREP expression will look like this:

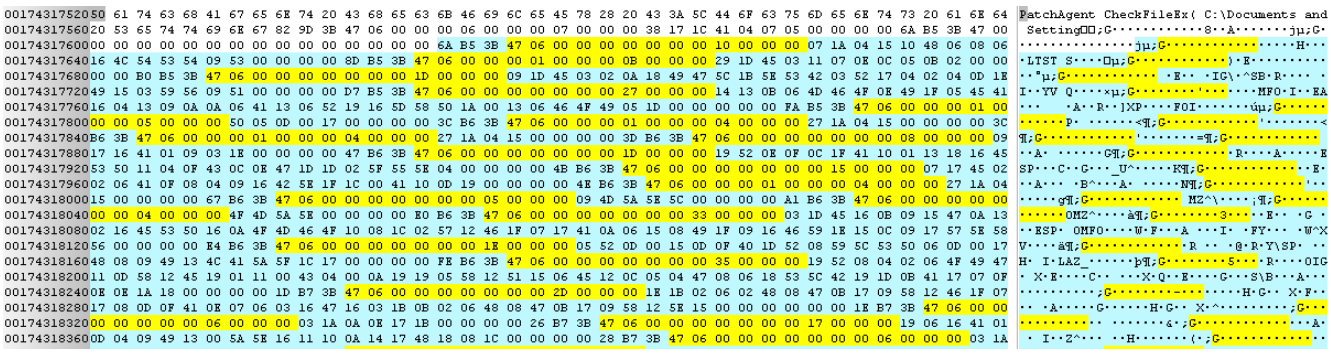
GREP EXPRESSION:

\x45|\x46|\x47\x06\x00{3,3}\x01|\x00\x00{3,3}[^x\00]\x00{3,3}

Note: Be sure to adjust the UNIX DATE BYTE values to correspond to the date range you are looking for.

The messages data found will usually be sequentially written to the disk in small data groups. This will result in search hits in close proximity to each other. This is important because visually inspecting your search hits will allow you to quickly rule out false positives and locate chat conversations.

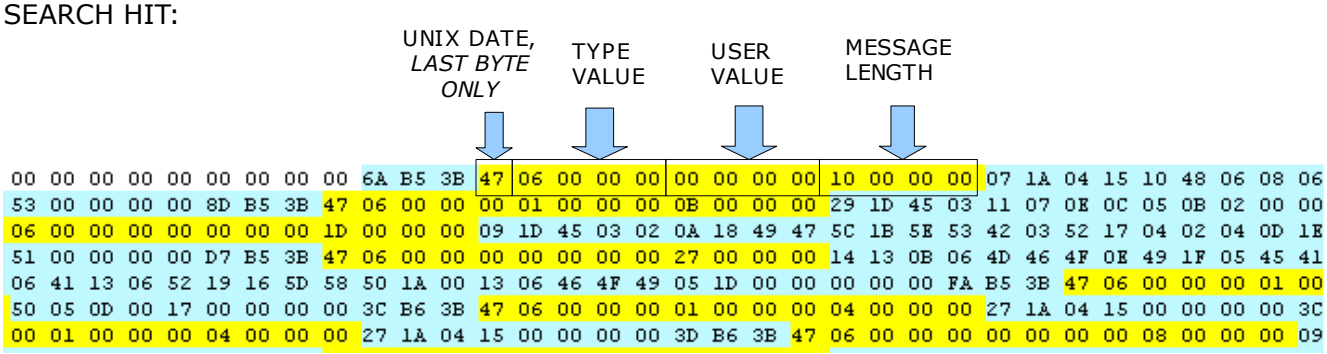
Example of search hits clustered together:



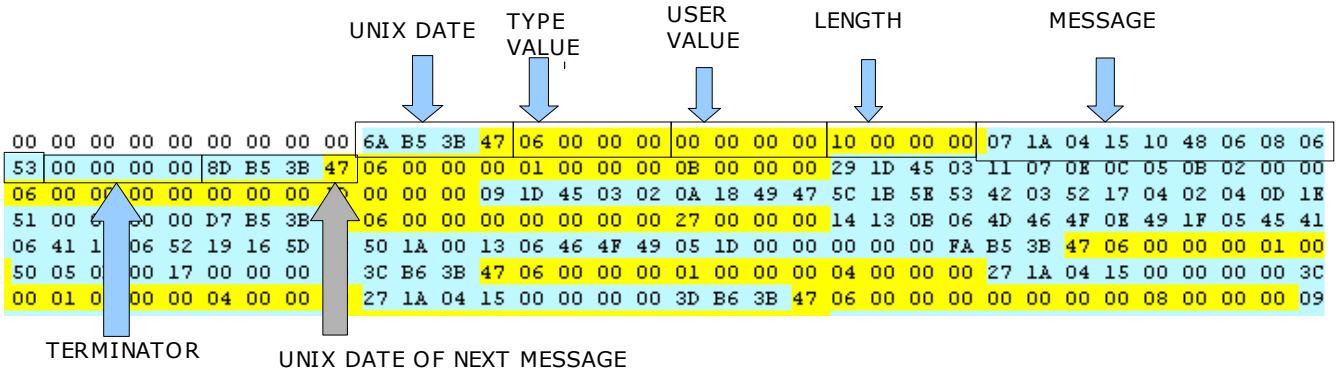
This clustering of search hits is indicative of a chat conversation.

Lets look at the hit itself:

SEARCH HIT:



ENTIRE MESSAGE:

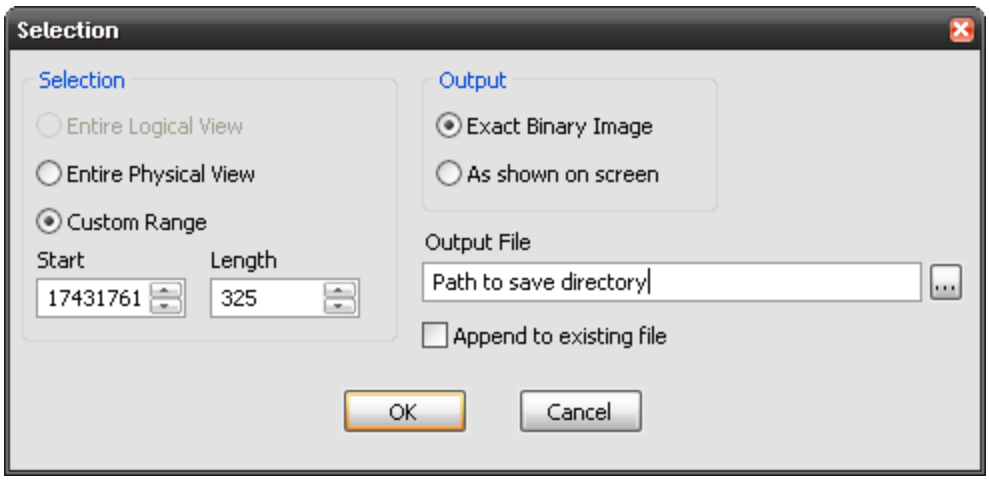


Notice that the actual message data block starts 3 bytes before the search hit. These 3 bytes complete the UNIX date stamp of the message. The message length in this example is 10. Counting the bytes from the message start to the terminator verifies this value. The next byte will be the beginning of UNIX date stamp of the next message.

Create readable .dat files

The search hits need to be parsed and converted into a .dat file that will be decryptable. This will require visual inspection and exportation from the image file.

Export the entire cluster of messages in a data file. In EnCase, highlight the data, right click and export using the "Exact Binary Image" option. **Use .dat as the file extension.** This allows the decryption program to see the file.



IMPORTANT: The exportation requires beginning the data carving on a date stamp and ending on a message terminator. If this is not done correctly, the decryption program may cause errors when reading the file.

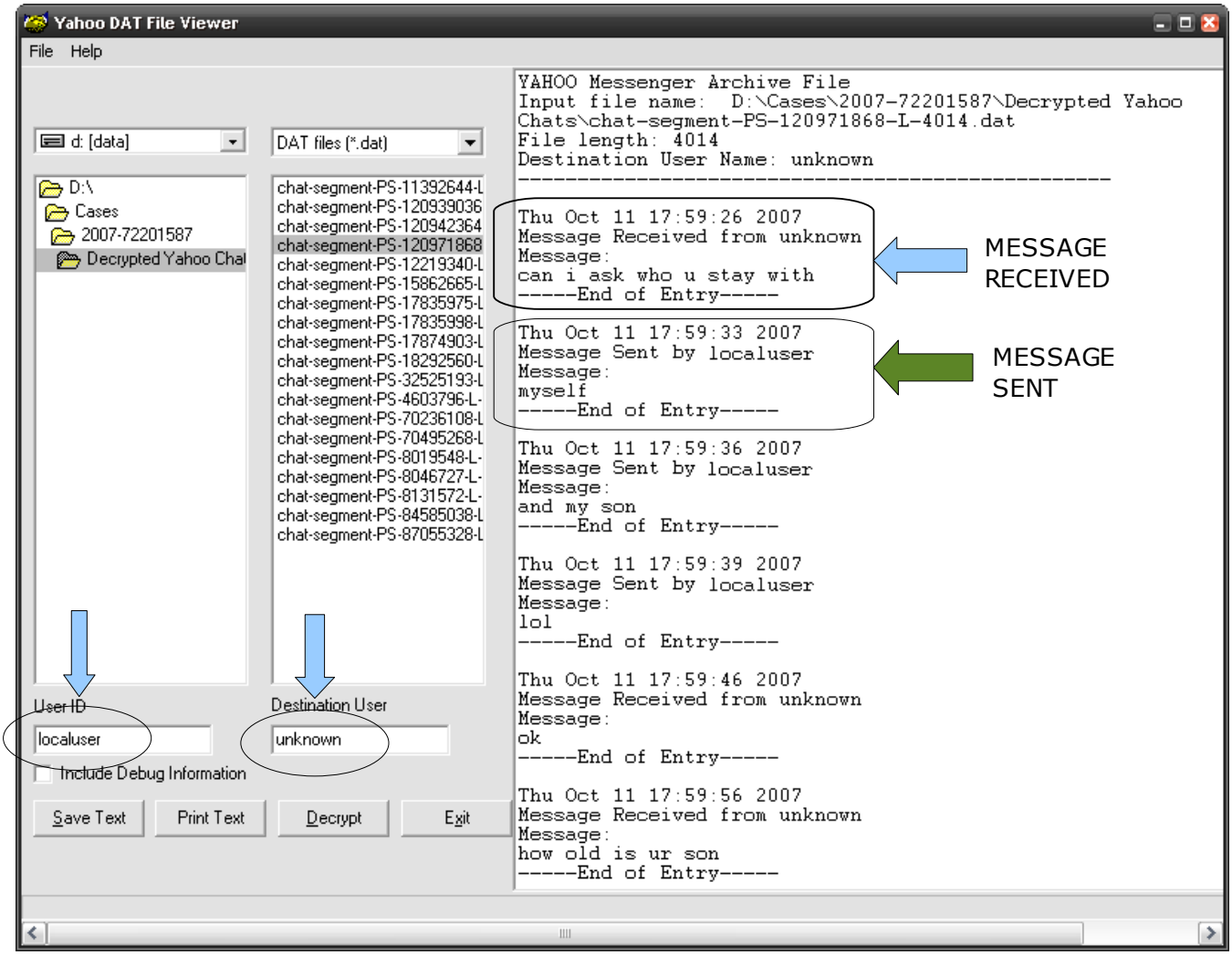
Decode encrypted chat fragments.

Once we have created the .dat files, we need to decrypt them. Run YAHOOM.exe and point it to the .dat files you created.

IMPORTANT: you MUST supply the correct local user name to decrypt the message content, otherwise it will not decode properly. YAHOOM.exe also requires a remote user name to be entered before it will decrypt the file. This remote user name does not need to be correct. Because you wont know what it its enter "unknown".

Enter the local user name and "unknown" for the remote user. Then point the program to the .dat files you created and hit Decrypt. The program displays the messages in order. If the local user name is inputted incorrectly, the only information that will not be readable is the message content itself. If this happens, recheck the user name or try another one.

YAHOOM.exe allows the exportation of the resulting chat archive to a text file.



Conclusion and Tips:

- This methodology uses YAHOOM.exe to decrypt the messages. Although this is not the only free message decryption tool, it seems to work the best on the .dat file created using this method. Feel free to try another tool if it works better for you.

- There are several different ways to write this GREP expression, however, the author has found that this way appears to find the most hits while reducing the amount of false positives.

- It is important to realize that the chat fragments are in unallocated space and because of this, may only contain partial information for each piece of message data.

- The missing data may cause YAHOOM.exe to lock up when trying to export the chat to a text file. If this happens, re export the chat fragment to a new .dat file removing the last chat segment from the dat file, being careful to end on a message termination byte.

- As with most computer forensic methods, this might not be the best. Feel free to tweak it to your liking. Hopefully is a good starting point to locate this type of data. If you happen to find a way to improve upon this methodology, please let the author know.

Detective Eric Oldenburg - Phoenix Police Department – eric.oldenburg@phoenix.gov