To: Distribution

From: Lindsey Spratt

Date: 06/21/78

Subject: Auditing I/O

Problem

Many users of Multics want a way to efficiently log I/O activity (make a record of input and output on an I/O switch). Two sites in particular, USL and AFDSC, have private tools for this purpose. It is useful for CRT users because they frequently have no hard copy to reference. It's also useful for wading through a long terminal session to find specific kinds of interactions. Finally, the ability to edit the log is useful for changing input lines and for constructing input lines from output lines.

Proposal

The uninstalled audit_ I/O module, which has been used unofficially at MIT and Phoenix for years, provides the logging and editing features. Auditing adds little storage overhead (as little as one record) and very little processing overhead (about two subroutine calls per line audited and a character comparison per input line). It comprises making entries in a file, hereafter referred to as the audit file, according to the whim of the user and the input and/or output activity on the user specified switch. It also allows printing the contents of the audit file. This provides the log referred to above in the form of the audit file. These capabilities make it easy to produce sample terminal sessions and to re-issue input, edited or un-edited.

Please send comments to:

Spratt.Multics,

or

Lindsey Spratt
Honeywell Information Systems
575 Tech Square
Cambridge, Mass. 02139

Or call:

(617) 492-9321
HVN 261-9321

Implementation

The auditing mechanism consists of the following four modules:

1. audit_
   is an I/O module, spliced between the switch to be audited and that switch's I/O module. It maintains the audit file and does some editing of the audit file. with the audit editor.

2. print_audit_file
   is a command that prints the audit file in a user specified format. Among other things, it provides a stable user interface in the face of changes to the audit file.

3. attach_audit
   is the command to set up auditing. If no control arguments are given, it sets up auditing on user_i/o for input and output, with editing on.

4. detach_audit
   is the command to discontinue auditing. If no control arguments are given, it discontinues auditing on user_i/o.

Documentation

In addition to the documentation of the individual modules, a narrative description of auditting I/O , as a whole, is needed. Where in the manuals to put a description of this sort is a problem. Either the Reference Guide or the Commands Reference seem likely locations.

Following is the draft documentation for the above modules. The editor is documented in the description of audit_.

Name: attach_audit

This command sets up a specified I/O switch, with a stream_input_output opening to be audited by the audit_ I/O module.

Usage

attach_audit {old_switch {new_switch}} {-control_args}

where:

1. old_switch
   is the name of the I/O switch to be audited. If no switch is given, user_i/o is the default. If only one switch is specified, it the old_switch.

2. new_switch
   is the name of an I/O switch to be used by the audit_ I/O module. If only one switch argument is given, it is the old_switch. The default value for new_switch is audit_i/o.<time>, where <time> is the value returned by date_time_ with "-" replacing blanks, and the time zone and day of the week removed.

3. atd_args
   are the attach description arguments selected from the control arguments for the audit_ I/O module. See the discussion of audit_ in the MPM Subroutines for more information. The default is -truncate.

Notes

If used with no arguments, attach_audit sets up auditing for the user_i/o I/O switch with input and output audited and editing on.

Auditing of old_switch is done by moving the attachment of old_switch to new_switch and then attaching old_switch to new_switch via audit_. See the MPM Subroutines discussion of the audit_ I/O module and the MPM Commands discussion of detach_audit for more information.

---
detach_audit

Name: detach_audit

This command discontinues auditing of a specified I/O switch by the audit_ I/O module.

Usage

detach_audit {switch}

where switch is the name of an I/O switch for which auditing is to be discontinued. If no switch is given, the default value assumed is user_i/o.

Note

See the MPM Subroutines document on the audit_ I/O module and the MPM Commands document on attach_audit.

print_audit_file

Name: print_audit_file, paf

This command prints the file produced by the audit_ I/O module. The format of the output, which lines are selected to be output, and the file to which the output is directed are all user specifiable.

Usage

print_audit_file {path} {-control_args}

1. path
is the name of the audit file to be printed. If no path is given, the current audit file is used if audit_ is attached. Otherwise, the most recent file in the home directory with the audit suffix is used.

2. control_args
The control arguments are:

-output_file path, -of path
specifies the pathname of the file to which print_audit_file directs its output. If no pathname is given, the output is printed on user_output.

-line_type line_identifiers, -lt line_identifiers
prints only those lines in the audit file having one of the specified line identifiers. If this control argument is not specified, the default is to print any type of line in the file. Any number of line identifiers may be given, separated by spaces. See below for a detailed discussion of line identifiers.

-line_address addr_blocks, -la addr_blocks
print only the lines in the audit file in any specified address block. The default is to print any line. Any number of address blocks may be specified, separated by spaces. An address block specification consists of either a single line address, or a pair of line addresses separated by a comma. See below for a detailed discussion of line addressing.

-line_numbers, -ln
print the line numbers.

print_audit_file

-metering, -mt
print the audit file with metering at the beginning of each line, preceding the tag if -tags is also specified. The default is not to print the metering.

-tags
print the audit file with the identifying tags at the beginning of each line, following metering information if -metering is also specified. The default is not to print the tags.

Notes

The -line_type and -line_address control arguments each specify a set of printable lines. The lines which are printed are those lines in both sets. Hence, if a line in the audit file is identified by at least one of the line identifiers in the -line_type argument and is in at least one of the address blocks specified by the -line_address argument, it will be printed. Otherwise, it will not be printed.

See the MPM Commands description of attach_audit and detach_audit, and the MPM Subroutines description of the audit_ I/O module for further information on auditing I/O.

Line_Identifiers

A line identifier comprises at most one tag and any number of string specifications. The line identifier must be quoted if it contains blanks or commas. A tag is an entry identifier generated by the audit_ I/O module when it makes an entry in the audit file. An entry may not end in a new line character (e.g., metering entries), or it may be longer than one line (e.g., a single output string containing new_lines). Hence, metered activity produces doubly tagged lines and output containing new_lines can produce untagged lines(only the first line of the entry having a tag). A string specification consists of a string of characters delimited by slashes. A line is identified by a line identifier if it has the same tag and contains all of the strings in the line identifier. The tags are:

EL	edit line, in audit editor
IC	result of a get_chars
IL	result of a get_line
M	metering data
OC	result of a put_chars
TC	control request trace
TM	mode request trace

print_audit_file

Metering data is not printed as a separate line, but as a header on the metered line.

example

```
IL          lines with tag IL
/foo/       lines with string "foo"
EL/yes/     lines with tag EL and string "yes"
/wh/TM      lines with tag TM and string "wh"
/file//out/ lines with string "file" and string "out"
```

Line_Addressing

There are four basic ways to identify a line in the audit file. A line address consists of any one of these four ways, or a combination of them. Two additional types of addressing are available in composite, or combination, addresses. The four basic ways are:

1. time of entry
2. absolute line number
3. entry tag
4. character string in the line

The two additional types of addressing are:

5. relative line reference
6. relative time reference

The time is given in 24 hour notation, down to tenths of a minute (e.g., 5:24 pm is 1724.0). Any number with a decimal point is interpreted as a time. A line can be addressed with the time only if audit metering was on.

A line number is either an integer or the special character $. The dollar sign indicates the last line of the file. Tags are one of the upper case letter pairs listed in the discussion of line identifiers.

A character string to be matched must be enclosed in, or delimited by, slashes("/"); upper-case letter pairs between slashes are not recognized as tags, just part of the string to be matched.

Relative line numbers are preceded by a plus sign (+) or a minus sign(-), to indicate the nth line preceding or following the starting address established by the preceding elements, respectively. A relative time is indicated in the same way as a relative line number, except it has a decimal point.

A composite address is a concatenation of two or more elements, an element being any one of the six addressing methods mentioned above. If an absolute line number is used, it can appear only once in a composite address, and then it must be the

print_audit_file

first (or left-most) element in the composite. Similarly, a time of entry can appear only once in a composite and must be placed in the left-most position of the composite. Neither absolute line number or time of entry need be used in a composite address, and both cannot be used in the same composite.

Composite addresses are evaluated from left to right. There can be no separation (i.e., blanks or commas) between elements. Each successive element is evaluated and the resulting address becomes the starting address for the evaluation of the next element. If the element to be evaluated is an entry tag, the search for a line with a matching tag begins at the starting address established by the preceding elements. As above, the resulting address is the matched line's address. If the current element to be evaluated is a relative line number, The starting line number is either added to or subtracted from by the amount indicated. If the element to be evaluated is a relative time, the evaluation process is somewhat more complicated. First, the time associated with the starting line number is determined. Second, the relative time is either added to or subtracted from it. Finally, the resulting line address is the first line having a time greater than or equal to the new time.

Example:

```
5.0          line occurring soonest after 0005.0 (12:05 am)
7            line number 7
IL           first line having tag IL
/what?/      first line containing the character string "what?"
7-2          line number 5 (second line preceding line 7)
$            last line in file
0C/why?/     first line containing character string "why?" after
              the first occurance of a line with tag 0C
0834.4TM-5   fifth line preceding first occurance of tag TM
              following 8:34.4 pm
```

Example Usage

The sample audit file was produced by the sample terminal session given in the documentation for audit_. It is named sample and is in the user's working directory. An exclamation mark is used to indicate user input lines. See the audit_ I/O module documentation in the MPM Subroutines for an explanation of the format of the audit file.

```
----------------                                    ----------------

print_audit_file                                    print_audit_file
----------------                                    ----------------

sample audit file

0000022 0000057 IL: io control user_i/o (audit_trace audit_meter audit_edit)
0000057 0000011 TC: io_call
0000011 0000012 TC: audit_meter
0000012 0000020 M : 1141.3    7.198  543 0000020 0000011 TC: io_call
0000011 0000020 M : 1141.3    0.011    0 0000020 0000011 TC: audit_edit
0000011 0000020 M : 1141.5    0.036   22 0000020 0000020 IL: io control user_i/o
0000020 0000020 M : 1141.5    0.063   16 0000020 0000042 OC: io_call: Expected argument missing, order
0000042 0000020 M : 1141.7    0.035   21 0000020 0000003 IL: !e
0000003 0000020 M : 1141.7    0.062   28 0000020 0000002 IL: p
0000002 0000020 M : 1141.7    0.015    1 0000020 0000020 OC: io control user_i/o
0000020 0000020 M : 1142.1    0.017   17 0000020 0000035 IL: e io control user_i/o ^audit_meter
0000035 0000020 M : 1142.1    0.016   11 0000020 0000011 TC: io_call
0000011 0000020 M : 1142.1    0.006    2 0000020 0000013 TC: ^audit_meter
0000013 0000019 IL: s/$/ audit_modes/r
0000019 0000032 EL: io control user_i/o audit_modes
0000032 0000011 TC: io_call
0000011 0000090 OC: audit modes: audit_input,audit_output,audit_edit,audit_trace,^audit_meter,audit_trigger=!
0000090 0000011 IL: cwd sample
0000011 0000006 IL: solve
0000006 0000001 TM:
0000001 0000002 IL: 4
0000002 0000013 IL: 1,-50,10,-21
0000013 0000003 IL: -1
0000003 0000001 TM:
0000001 0000039 OC: iterations,f(x),f'(x),initial x,final x0000039 0000084 OC:
         4           4.31958125e-002    -4.96391034e+001    -1.00000000e+000
 1.92068887e-002 0000084 0000001 OC:
0000001 0000006 IL: solve
0000006 0000002 IL: 4
0000002 0000010 IL: b/-50/r!E
0000010 0000013 EL: 1,-50,10,-21
0000013 0000003 IL: 20
0000003 0000039 OC: iterations,f(x),f'(x),initial x,final x0000039 0000084 OC:
        10           4.05459558e-002    -4.96381655e+001     2.00000000e+001
 1.92602717e-002 0000084 0000001 OC:
```

---

print_audit_file

---

print_audit_file

---

examples

```
!  paf sample -la 5
   aucit_edit
!  paf sample -la 1,5
   io control user_i/o (audit_trace audit_meter audit_edit)
   io_call
   audit_meter
   io_call
   audit_edit
!  paf sample -la 1,5 -lt IL
   io control user_i/o (audit_trace audit_meter audit_edit)
!  paf sample -lt IL/audit/ -la 1141.5EL+2,/cwd/+4 -tags
   OC: audit_modes: audit_input,audit_output,audit_edit,^audit_trace,
   \c^audit_meter,audit_trigger=!
   IL: cwd sample
   IL: solve
   IL: 4
   IL: 1,-50,10,-21
!  paf sample -lt EL -metering
   io control user_i/o audit_modes
   1,-50,10,21
```

Name: audit_

The audit_ I/O module is used to monitor input and/or output directed over a given stream I/O switch. Entries of various kinds are appended to the audit file in response to input and output on the specified switch. These are described in detail below.

Entry points in this module are not called directly by users; rather, they are accessed through the I/O system.

Attach_Description

The attach description has the following form:

audit_ switch_name {-control_args}

where:

1. switch_name
   is the name of an I/O switch to be inserted between the existing switch and its I/O module.

2. control_args
   The control arguments are:

   -truncate, -tc
   truncates the old audit file, if it has the same name as the new one.

   -extend
   extends the old audit file, if it has the same name as the new one.

   -pathname, -on
   specifies the pathname of the new audit file. The default is time.audit in the home directory, where time is the time returned by date_time_ at the time of attachment.

The attachment of audit_ does an implicit open of the switch. Attachment is simplified by use of the attach_audit command. (See the MPM Commands section for a description of attach_audit and detach_audit.)

Control_Operation

The control requests for the audit_ module are listed below. Control requests not preceded by "audit_" are passed on to the I/O module being audited.

audit_trace
: cause tracing of all control and mode calls to the module. An entry with a TC or TM identifier describing the contents of the given call is placed in the audit file.

audit_truncate
: truncate the audit file.

audit_input
: turn on auditing for input lines.

audit_output
: turn on auditing for output lines.

audit_edit
: enable editing.

audit_transparent
: turn off auditing of audit and audit edit requests, as well as their results. EL entries are still audited.

audit_suspend
: disable all audit capabilites.

audit_resume
: restore audit capabilities that were in effect before the last audit_suspend.

audit_meter
: write a metering record before each entry in the file containing the actual time of the metering, incremental cpu time since the last metering, and the incremental page faults.

audit_trigger=x
: set the audit request trigger to the character specified by x.

audit_modes
: return the current audit modes in a char (256) varying string.

audit_file_size=n
set the maximum number of records for the audit file to n. When the maximum is reached, the file is scrolled. If n is the character string "unlimited", the file will grow without limit.

Five of the control requests, audit_trace, audit_input, audit_output, audit_edit, and audit_meter, can be turned off by preceding them with a "^". The default modes supplied at attachment are:

```
audit_input
audit_output
^audit_trace
audit_edit
^audit_meter
audit_trigger=!
audit_file_size=unlimited
```

Other Operations

All other operations are passed on to the audited I/O module.

Audit Requests

The audit_edit control must be issued to use these requests. A three character sequence is used to make an audit request; the audit trigger character ("!" by default), followed by the specific request character, followed by a new_line character.

!.
print "audit I/O module".

!?
print a brief description of audit requests available.

!e
enter the audit editor.

!E
enter the audit editor, with the input line processed as edit requests.

!a
abbrev expand the input line. See the MPM documentation on abbrev for more information.

!r

replay the input line. That is, display the input line without a new line. Further input up to the next new line is appended to the redisplayed input. This is the input line which is passed through the audit_ I/O module.

!t

instructs the audit_ module not to log the input line, i. e. to make it transparent.

!n

no operation. The input line to which this is appended is not passed through the audit_ module.

The Audit File

The audit file, by default, has the pathname:

>udd>Project_id>Person_id>time.audit

where time is the time returned by the date_time_ subroutine with "--" insert. The time zone and the day of the week aren't used. This produces a new file for every attachment of the audit_ module. An example entry name is 05/25/78--0723.5.audit . Since the audit file can be a multi-segment file, there is no limit on its size, other than that set with audit_file_size.

Each entry is preceded by a 20 character header, with fields as follows:

1 - 7	last entry length
9 - 15	this entry length
17 - 18	entry type identifier
19 - 20	:

The entry type identifiers are:

EL

edit line, in audit editor

IC

result of a get_chars

IL

result of a get_line

M
metering data

OC
result of a put_chars

TC
control request trace

TM
mode request trace

Editing

The audit editor edits and executes lines that have been logged by the audit_ I/O module. The audit editor is enabled by the audit_edit control request. When enabled, it is invoked by a two-character sequence consisting of the trigger character followed by a request character of "e" or "E". If the request character is "e", text preceding the trigger character becomes the current edit line. If no text precedes the trigger character, then the previous input line becomes the current edit line. If the request character is "E", the previous input line becomes the current edit line and text preceding the trigger character is immediatiely executed as editor requests. After the initial actions described above, the editor enters a read - execute loop. The audit editor can be called recursively.

Request Syntax

Requests can follow each other directly or may be separated by blanks, tabs, and newlines. A colon (":") can precede any command to indicate that all audit entries are to be processed. The default is to process only audit entries tagged as input. Requests requiring string arguments allow any reasonable character (not newline) to be used as a delimiter. For the purpose of this document, "/" is used as the string delimiter.

Miscellaneous Requests:

```
.          print "audit editor"
?          print a brief description of the available requests
p[N]       print N lines (if not specified, N = 1)
s/S/S'/    substitute S' for each occurence of S
d tag,n    set the default search tag to tag.
           use n characters of the tags to match (n equals 1 or 2)
```

```
          if no n is given, 1 is the default value
a         expand abbreviations in the edit line
l         set the edit line to the last line returned
t         print the type of the current audit line
```

Mode Requests

```
V on       enter verbose mode
V off      exit verbose mode
A on       audit the input and output from the editor
A off      don't audit the editor
```

Cursor Movement Requests

```
+[N]       forward N lines (if not specified, N = 1)
-[N]       backward N lines (if not specified, N = 1)
f/S/       forward to the next succeeding line containing S
b/S/       backward to the next preceding line containing S
```

Execute Requests

```
x          pass the line being edited to the command processor
e          pass the rest of the request line to the command processor
```

Exit Requests

```
n          return a newline
r          if non-null, return the rest  of the request line;
           else return the current edit line (without the trigger sequence)
q          return what was passed in (including trigger sequence)
```

Notes

1. The last delimiter may be omitted from an s request if it is the last request on a line.
2. If S is null in an s, f or b request, the last S specified in a previous s, f or b request is used.
3. If a cursor movement request is the last request on a request line, the current line is displayed.
4. No lines in the audit file are changed by the editor; only copies are modified.
5. The S field of s, f and b requests is interpreted as a

qedx-style regular expression.

6. The S' field of an s request is interpreted as in qedx, i.e., the & convention is supported.

Examples

The following examples are first, a sample terminal session. In this sample, user input lines are indicated by an exclamation mark(!). Second, the audit file produced by the sample terminal session. Third, usage of various audit and audit editor requests not in the sample terminal session. Each of the examples in this third set is given as though it is the next input line in the sample terminal session.

Sample terminal session

```
!  attach__audit audit_i/o user_i/o
!  ic control user_i/o (audit_trace audit_meter audit_edit)
!  ic control user_i/o
   io_call: Expected argument missing, order
!  !e
!  p
   ic control user_i/o
!  e ic control user_i/o ^audit_meter
!  s/$/ audit_modes/r
   audit modes: audit_input,audit_output,audit_edit,audit_trace,^audit_meter,audit_trigger=!
!  cwd sample
!  solve
!  4
!  1,-50,10,-21
!  -1
   iterations,f(x),f'(x),initial x,final x
           4          4.31958125e-002     -4.96[illegible]34e+001     -1.00000000e+000
   1.92068887e-002
!  solve
!  4
!  b/-50/r!E
!  20
   iterations,f(x),f'(x),initial x,final x
          10          4.05459558e-002     -4.9638[illegible]55e+001      2.00000000e+001
   1.92602717e-002
```

------ ------

audit_ audit_

------ ------

Sample_audit_file

```
0000022 0000057 IL: io control user_i/o (audit_trace audit_meter audit_edit)
0000057 0000011 TC: io_call
0000011 0000012 TC: audit_meter
0000012 0000020 M : 1141.3    7.198   543 0000020 0000011 TC: io_call
0000011 0000020 M : 1141.3    0.011     0 0000020 0000011 TC: audit_edit
0000011 0000020 M : 1141.5    0.036    22 0000020 0000020 IL: io control user_i/o
0000020 0000020 M : 1141.5    0.063    16 0000020 0000042 OC: io_call: Expected argument missing, order
0000042 0000020 M : 1141.7    0.035    21 0000020 0000003 IL: !e
0000003 0000020 M : 1141.7    0.062    28 0000020 0000002 IL: p
0000002 0000020 M : 1141.7    0.015     1 0000020 0000020 OC: io control user_i/o
0000020 0000020 M : 1142.1    0.017    17 0000020 0000035 IL: e io control user_i/o ^audit_meter
0000035 0000020 M : 1142.1    0.016    11 0000020 0000011 TC: io_call
0000011 0000020 M : 1142.1    0.006     2 0000020 0000013 TC: ^audit_meter
0000013 0000019 IL: s/$/ audit_modes/r
0000019 0000032 EL: io control user_i/o audit_modes
0000032 0000011 TC: io_call
0000011 0000090 OC: audit modes: audit_input,audit_output,audit_edit,audit_trace,^audit_meter,audit_trigger=!
0000090 0000011 IL: cwd samole
0000011 0000006 IL: solve
0000006 0000001 TM:
0000001 0000002 IL: 4
0000002 0000013 IL: 1,-50,10,-21
0000013 0000003 IL: -1
0000003 0000001 TM:
0000001 0000039 OC: iterations,f(x),f'(x),initial x,final x0000039 0000084 OC:
         4           4.31958125e-002    -4.96391034e+001    -1.00000000e+000
 1.92068887e-002 0000084 0000001 OC:
0000001 0000006 IL: solve
0000006 0000002 IL: 4
0000002 000001C IL: s/-50/r!E
000001C 0000013 EL: 1,-50,10,-21
0000013 0000003 IL: 20
0000003 0000039 OC: iterations,f(x),f'(x),initial x,final x0000039 0000084 OC:
        10           4.05459558e-002    -4.96381655e+001     2.00000000e+001
 1.92602717e-002 0000084 0000001 OC:
```