DATE: January 3, 1974

TO: Distribution

FROM: R. F. Mabee

SUBJECT: Metering tools and security

This is a revision of MTB-011. The first two sections below are taken verbatim from that bulletin, while the part proposing a solution has been modified to incorporate suggestions from HISI and PDO. The essential points of difference are:

1) Exact compatability is unnecessary. This eliminates the contested phcs_ transfer vector.

2) The compiled limits table will be in the libraries. The installation will maintain it.

Problem statement

Hardcore data segments contain many items which are useful for measuring various aspects of system performance. They also contain some information which should be protected, such as I/O buffers. Currently all metering tools obtain their data by calling the ring-zero entry phcs_$ring_0_peek. This loophole is too general: it can be readily misused to read any ring-zero segment. This problem could be ignored when only a trusted few had access to call phcs_, but we now have on the order of a hundred people with good reasons for needing access.

The solution to this problem should make it possible to let any user read selected system data without endangering specific protected data. The set of users who are permitted to use the meters should be controllable (by an ACL) to suit individual installations.

Analysis

Due to haphazard design, some existing data segments contain both private and public data. These data should be separated into different segments because the segment is the fundamental unit of hardware access control. However, hardcore programs which use these segments may run in any user's process, so (without per-user ring brackets) the ACL's on the data segments cannot be used to discriminate among users. Therefore, access to

metering data has to be controlled by an inner-ring filter program. There are two alternatives at this point:

(1) Rearrange data according to protection requirements. Let the filter program decide per-segment whether or not to permit the access.

(2) Leave data intermixed. Create a more complicated filter program which knows what parts of which segments to protect.

It seems clear that the hardcore data should be grouped according to the access restrictions to be placed on it, so that finer distinctions can be drawn between public access, metering access, system-programmer access, and highly-privileged access. Such a global rearrangement would take too long (several months) to meet the immediate need, but I strongly recommend that any new or redesigned data bases be organized in this way.

For interim use I will choose (2) above, but how should the filter program decide what to protect? Well, several people have pointed out to me that the most critical hardcore data segments (typically I/O buffers) have fixed-size headers (essentially public) followed by some variable structure of per-device or per-user blocks (essentially private). Therefore, I propose to associate with each hardcore segment number Sn a limiter $L_{Sn}$ such that word i of the segment can be read in outer rings if and only if $0 \leq i < L_{Sn}$.

Proposed solution

There will be two "peek" gates accessible in the user rings: phcs_$ring_0_peek, the existing ring-zero gate, and metering_ring_zero_peek_, a new ring-one gate. The ACL of phcs_$ring_0_peek will restrict it to privileged system programmers. Everyone interested in reading the meters will be on the ACL of metering_ring_zero_peek_. metering_ring_zero_peek_ will call ring_zero_peek_filter_ in ring one, to validate the request. A new gate from ring one into ring zero, administrative_ring_zero_peek_, will call the hardcore entry ring_0_peek exactly as phcs_$ring_0_peek does; *.*.* will have access to this gate from ring one.

A user-ring subroutine called ring_zero_peek_ will be provided, to call either phcs_$ring_0_peek or metering_ring_zero_peek_ according to the user's privilege, and to map lack of access to call either gate into a suitable error code. The meter-printing programs will call this routine so as to work properly for whatever class of users is entitled to examine the given meters, and to produce a reasonable error message for any underprivileged users.

ring_zero_peek_filter_ will validate all requests against a table of read limiters stored in a segment in the libraries. This segment, >system_library_tools>ring_zero_meter_limits.table,

will have ring brackets of 5,5,5 so that it may be used by other programs. This data segment will be initialized each time the system comes up, by the command "initialize_peek_filter >system_library_1>ring_zero_meter_limits.ASCII" in the answering service exec_com file. The pathname designates an ASCII file containing reference names and corresponding numeric read limiters, one per line. Any segment not mentioned in this list is not accessible. A standard limiter list will be loaded from the system tape; the installation can specify its own if necessary by changing the pathname given with the initialize_peek_filter command.

A suggested list of limiters is attached (Appendix A). This list restricts access to the minimum required for the installed metering tools to be usable.

Schedule

In the following list, step 2 can overlap steps 3 and 4. Step 4 can reasonably be completed during January.

1) A dummy version of the user-ring interface ring_zero_peek_ will be installed immediately.

2) All twenty programs which call phcs_$ring_0_peek will be modified to call ring_zero_peek_. PDO has offered to perform this task.

3) The rest of the filtering mechanism will be installed, consisting of two gates, a ring-one program, the command for the initializer, and a data segment.

4) The initializer will start initializing the filter.

5) PDO will reduce access on phcs_ as they choose.

Please feel free to make comments or ask questions about this proposal, on line to Mabee.CompSys or by phone at 253-6004.

Appendix A - suggested limits file.

```
" This list controls access to metering (and related) data in
" hardcore.  Only data named in this list can be read.  The limit
" is given as 262144 if some program copies the entire segment.

" Last modified on 01/03/74 at 14:08:35 by R F Mabee.

" Name of segment              Limit     " Reason

tc_data                        262144    " traffic_control_queue.
sst_seg                        262144    " traffic_control_queue.
dseg                              256    " Used by several tools.
config_deck                    262144    " print_configuration_deck.
active_hardcore_data               74    " system_link_meters.
hcs_                           262144    " spg_ring_0_info_
hcs_.link                      262144    " and meter_gate.
phcs_                          262144
phcs_.link                     262144
hphcs_                         262144
hphcs_.link                    262144
imp_dim_gate_                  262144    " meter_gate.
imp_dim_gate_.link             262144
net_                           262144
net_.link                      262144
netp_                          262144
netp_.link                     262144
admin_gate_                    262144
admin_gate_.link               262144
xray_communications.link           54    " link_meters.
pds                              2540    " link_meters.
dsu170_seg                        256    " device_meters.
dsu270_seg                        256
dsu181_seg                        256
dsu190_seg                        256
bulk_store_mailbox                256
fsdct                            4096    " Can't get exact figure.
tty_buf                           112    " Clark's uninstalled tool.

" The limit on tc_data could be lowered to 256, to protect per-user
" time-used figures.  traffic_control_queue and print_apt_entry
" would then stop working.
```

Appendix B - new entry calling sequences.

This section should provide adequate temporary documentation of new entries, and can readily be turned into MPM writeups.

Name: ring_zero_peek_

This user-ring procedure is called to copy some specified data out of hardcore data segments. Some users are only permitted to copy certain data, others none at all. These restrictions are enforced by access lists on various gates; this module merely tries to decide which gate to call in order to obtain the requested data, and fabricates an appropriate error code if the user has insufficient privilege.

Usage

```
declare  ring_zero_peek_ entry (pointer, pointer,
                  fixed binary (19), fixed binary (35));
         call ring_zero_peek_ (from, to, len, code);
```

1) from is the address of the desired data in ring zero. (Input)

2) to is the address of the caller's data item which is to be copied into. (Input)

3) len is the size of the data item (i.e. length in words). (Input)

4) code is a standard error code, or zero. (Output)

Name: administrative_ring_zero_peek_

This is a gate from ring one to ring_0_peek in ring zero. For usage information, see description of ring_0_peek.

Name: metering_ring_zero_peek_

This is a gate from rings two through five to ring_zero_peek_filter_ in ring one. For usage information, see description of ring_zero_peek_filter_.

Name: ring_zero_peek_filter_

This module provides a way to allow users to examine various system-wide, hardcore data bases while maintaining security for all other hardcore data. The segment >system_library_tools>ring_zero_meter_limits.table specifies a read limiter for each hardcore segment; only that many words from each segment are supposed to be accessible.

ring_zero_peek_filter_ is used to copy data out of ring-zero segments. If the request is invalid, it is disregarded and an appropriate error code is set. This entry point is to be called through a gate.

Usage

```
declare  ring_zero_peek_filter_ entry (pointer, pointer,
                  fixed binary (19), fixed binary (35));
         call ring_zero_peek_filter_ (from, to, len, code);
```

1) from is the address of the desired data in ring zero. (Input)

2) to is the address of the caller's data item which is to be copied into. (Input)

3) len is the size of the data item (i.e. length in words). (Input)

4) code is a standard error code, or zero. (Output)

Name: initialize_peek_filter

This privileged command is executed by the initializer process to define the set of hardcore data items which constitute metering data. (See ring_zero_peek_filter_.)

Usage

```
initialize_peek_filter  pathname
```

1) pathname designates an ASCII file listing segment names and corresponding read limiters, one set per line. A partial sample is shown below.

Note: A standard list is provided on the system tape: >system_library_1>ring_zero_meter_limits.ASCII.

Example:

```
" Name of segment              Limit     " Reason

tc_data                       262144    " traffic_control_queue.
sst_seg                       262144    " traffic_control_queue.
dseg                             256    " Used by several tools.
config_deck                   262144    " print_configuration_deck.
active_hardcore_data              74    " system_link_meters.
hcs_                          262144    " spg_ring_0_info_
hcs_.link                     262144    " and meter_gate.
```