To: Distribution

From: T. H. Van Vleck

Date: December 3, 1976

Subject: Unattended Operation of Multics - Part II

This memorandum describes the detailed plan for unattended operation of Multics.

OVERVIEW

MTB-152 described the basic plan for unattended operation, which provided for a "flagbox" accessible to both BOS and Multics, consisiting of 36 one-bit switches. As actually implemented, the flagbox is a 16-word segment when viewed from Multics and a region of the toehold as seen from BOS: in addition to the switches it contains data items such as the hardcore DBR value which BOS uses for dumping Multics.

Flag Usage

The set_flagbox command and the get_flagbox command/active function are used to get and set the flags in the toehold while Multics is operating. Three flags have preassigned meanings and are known by keywords in these commands:

1. auto_reboot - TRUE if the system is to attempt to reboot itself after it has crashed.

2. booting - TRUE during bootload. It is turned off at the end of part 3 of system_start_up.ec, when bootload is over. This flag prevents the system from looping attempting to reboot if it crashes before it comes up.

4. rebooted - TRUE if the system has rebooted as a result of automatic operation.

5. unattended - TRUE if the system is not attended by an operator.

In addition, the RTB flag may be tested: it may have one of the values

MANUAL	manual XED 4000
CRASH	system crash
SHUT	normal shutdown
CALL	operator call to BOS

and the SSENB flag may also be tested to see if the storage system has been enabled.

OPERATIONAL INSTRUCTIONS

The important difference between the old and new way of running the system is that the system may be operated in several new modes:

automatic or manual

unattended or attended

The old way of doing things corresponds to manual, attended operation.

In automatic mode, the system will take a dump and reboot if it crashes. In manual mode, the dump may or may not be automatic, but the system won't reboot without positive operator instructions.

In unattended mode, the system assumes that no tape mount requests can be honored.

Changing Modes

The operator may turn on automatic mode as part of the bootload process. To do this, he invokes the AUTO runcom instead of typing BOOT.

If automatic mode was set by the use of AUTO, the operator may turn off automatic mode with the command

```
x auto off
```

and may re-enable it with the command

```
x auto on
```

but if AUTO was not used to boot the system, typing "x auto on" will have no effect.

Setting_Unattended_Mode

A new system function will be provided in admin.ec so that the operator can conveniently set the system into unattended mode. Typing

```
          x unattend
```

will invoke the following steps:

```
            sc_command deldev tape_(01 02 03 04 05 06 07 08)
            sc_command word login Unattended service
            set_flagbox unattended true
            set_flagbox auto true
            set_flagbox rebooted false
```

Installations with more or fewer than eight tape drives will modify the text of admin.ec. Tese steps leave backup and I/O daemon functions running; some sites may wish to modify the text of admin.ec to log some daemon processes out.

Returning_to_Attended_Mode

When the system has been placed in unattended mode, the operator may revert to attended operation by typing

```
          x attend
```

which will perform the following steps:

```
            sc_command adddev tape_(01 02 03 04 05 06 07 08)
            sc_command word login
            set_flagbox unattended false
```

Other operations such as logging in daemons may be added.

Rebooting_after_a_Crash

If the system reboots automatically after a crash, the most sensible thing to do is to assume that there is no operator present. Therefore, the following lines are added to system_start_up.ec:

```
          &if [and [get_flagbox unattended] [get_flagbox rebooted]]
          &then sc_command deldev tape_(01 02 03 04 05 06 07 08)
```

so that until the operator explicitly reattaches the tape drives or does an "x attend" no user process will hang waiting for a tape. In order to avoid reboot loops, the following lines are used:

```
        &if [and [get_flagbox unattended] [get_flagbox rebooted]]
        &then set_flagbox auto false
```

These commands prevent repeated attempts to reboot without operator intervention. If these lines are omitted from system_start_up.ec, the reboot loop will terminate when >dumps runs out of quota, copy_fdump fails, and FDUMP finds the dump partition full. Sites may add additonal actions to handle this situation. For example, a site may modify system_start_up.ec to submit a deferred absentee job which will enable rebooting if the system stays up for at least an hour.

What will Prevent Recovery

Many events could cause the system not to reboot after a crash. Among these are

- System loop or failure to return to BOS - in this case the operator will enter BOS via XED 4000, and the runcoms will ask the operator to issue a command or type EOM to continue with automatic recovery. Since switches must be thrown to cause the manual entry to BOS, the runcoms know that there is an operator present to answer the question.

- Switches are not 004000717200 - this event also indicates that there is an operator in the machine room. The runcoms will ask the operator to hit EOM before continuing with recovery. Thus, even if the operator's console fails, the system can be taken out of automatic recovery mode.

- The system_start_up.ec never finished - in this case the booting flag will still be on. The runcoms will take a dump and do an emergency shutdown, but will await operator instructions from the console. If the operator types an EOM the system will try to reboot.

- The auto_reboot flag is off - automatic mode is turned off before giving the operator a chance to type, so that if the operator just types BOOT the system continues in manual mode. Automatic mode may also be turned off by the set_flagbox command executed while multics is running. The runcoms will print a message that the system is in manual mode and await an EOM before recovering.

- Some disk volume cannot be accepted - in this case the initializer process will have typed a message and inhibited automatic startup. The system will hang at operator command level in ring 1 or ring 4, depending on when the error is detected.

- FDUMP failed - in this case the runcoms will attempt to take a dump to tape. Since this requires tape handling, the operator will be asked to mount the tape and the system will await an EOM from the console.

- Explicit call to BOS - If BOS is entered as a result of a call to hphcs_$call_bos, the system assumes this is due to operator intervention. The runcoms print a message and await console input. If the operator types GO the system resumes operation.

- Lock error during shutdown - If the shutdown state is not 4, after an attempted shutdown, the runcoms comment and await console input.

- Reboot loop - if the system attempts to reboot itself repeatedly, this may be a sign of some system problem which does not prevent answering service startup but crashes the system later. The standard system_start_up.ec will not reboot the system twice without operator intervention, because automatic mode gets turned off. If this plan seems to be too conservative for some sites, they may modify the system_start_up.ec to take other action.

Runcom Programming

In order to understand the following RUNCOM files, it is important to note that BOS RUNCOM is not recursive. That is, when one runcom calls another, the first runcom is entirely forgotten. When the runcom file is exhausted BOS reads the next command line from the terminal (or card reader).

The TTY command causes a line to be read from the typewriter and executed as a command. If the line is null, the runcom continues. If the line is a BOS command, the command is executed. But if the line is a runcom name, the current runcom is forgotten and the new one called.

RUNCOM FILES

auto.runcom

```
* automatic restart
*
if switch 777777777777 neq 004000717200 ngsw
flag 1 on            * set automatic mode
restar star          * boot system
```

restar.runcom

```
* restart multics - auto or manual
*
flag 2 on           * set booting
config p cpu mem    * document config
*
boot &1 &2 &3 &4 &5 &6 &7 &8 &9
*
rtb                 * system returned to bos
```

rtb.runcom

```
* come here after a boot has returned to bos
*
if rtb manual mrtb  * if manual rtb
if rtb call boscal  * if bos called
if rtb crash crash1 * if system crash
if shut eq 4 normxt * if shutdown complete no errs
write => crash during shutdown - eom for dump
flag 1 off          * leave auto mode
tty                 * await operator input
crash1              * who was that masked man
```

crash1.runcom

```
* execute if system crashed
*
fd355               * dump fnp
blast crash         * sorry losers
fdump               * take core dump
if fdump eq 1 tdump * eq 2 if wouldnt fit, happens often
if flag 1 tryagn    * if auto mode
write => system crashed - manual mode, eom for esd
tty                 * pause
tryagn              * try esd
```

tryagn.runcom

```
* try to restart
*
if not ssenb write => storage system not enabled - eom for esd anyway
if not ssenb tty     * await operator input
esd                  * emergency shutdown
if shut neq 4 write => esd failed - will salvage
if flag 2 noboot     * if a.s. didnt turn off we never came up
flag 4 on            * turn on rebooted flag
if flag 1 auto       * loop if auto mode
flag 4 off           * turn rebooted off again
write => manual mode - eom for auto restart
tty                  * await operator input
auto                 * reenter reboot loop
```

tdump.runcom

```
* tdump - take a dump to tape
*
write => fdump failed - mount dump tape on 1 and eom
tty                  * await operator input
dump
tape 1
proc all
eof
quit
write => label tape for printing - eom for esd
tty                  * await operator input
tryagn               * try to get started again
```

noboot.runcom

```
* died during bootload
*
write => crashed during boot - eom for reboot
flag 4 off           * turn off rebooted
flag 1 off           * if opr give other cmd not in auto
tty                  * await operator input
auto                 * reenter auto mode
```

normxt.runcom

```
* normal shutdown stop
*
write => normal shutdown - eom for auto restart
flag 1 off          * stop auto mode
flag 4 off          * turn off rebooted
tty                 * await operator input
auto                * reenter loop
```

boscal.runcom

```
* manual call to bos
*
write => bos called - eom for dump, or gogo
tty                 * await operator input
flag 1 off          * real crash turn off auto
flag 4 off          * turn off rebooted
crash1              * take cump
```

gogo.runcom

```
* restart interrupted multics
*
go
rtb
```

mrtb.runcom

```
* manual rtb comes here
*
write => manual rtb - eom for dump
flag 1 off          * stop auto mode
flag 4 off          * turn off rebooted
tty                 * await operator input
crash1              * take nice dump
```

ngsw.runcom

```
* switches ng
*
write => switches not 004000717200 - fix and eom
flag 4 off             * turn off rebooted
flag 1 off             * disable auto mode in case he type command
tty                    * await operator input
auto                   * reenter loop
```