HITMAN V10.0

OpenVMS Idle Process Killer &
System Monitor

By Saiga Systems

Quick links:

Saiga Systems Software Inc.
#215, 801 - 6th Street SW
Calgary, Alberta, Canada
T2P 3V8

Voice: 1-800-561-8876

Telephone	(403) 263-1151
Fax	(403) 263-0744
Internet	support@saiga.com
WWW	http://www.saiga.com/
listServer	cohort@saiga.com

May 1, 2003

The information in this document is subject to change without notice and should not be construed as a commitment by Saiga Systems Inc. Saiga Systems Inc. assumes no responsibility for any errors or omissions that may appear in this document. This is the first release of the Version 10.0 manual; all comments regarding errors, omissions, etc. in this manual will be gratefully received. No guarantee is made that the new items documented in this manual will not change in the final release.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

DEC, DECNET, Alpha AXP, VAX and OpenVMS are trademarks of HP. HITMAN and Cohort are trademarks of Saiga Systems Inc.

The READER'S COMMENTS form on the last page of this document requests your critical evaluation to assist us in preparing future documentation.

Table of Contents

Using HITMAN to manage your Alpha AXP or VAX

This chapter shows you how HITMAN helps you tighten security, free resources and get more use of the reousrces available on your Alpha AXP or VAX.

With HITMAN, your system handles more users because you:

- free memory
- clear dialup ports
- release limited license slots

by terminating or disconnecting idle processes and, optionally, setting connect time limits. If a process is not active within the time you specify, HITMAN gets rid of it. If a process is connected for the full amount of time you specify, HITMAN gets rid of it. No more terminals left logged in overnight. No more dialup lines tied up on weekends because some user forgot to log off.

And, if you are concerned about security, HITMAN will plug security leaks and safeguard your company's valuable data by disconnecting unattended terminals.

HITMAN keeps an eye on your system and tells you when anything unusual or potentially dangerous occurs. HITMAN tells you when:

- a disk drive logs an error
- disk free space falls below limits you have set
- key system processes (such as ERRFMT) are missing
- jobs such as the daily backup are missing from queues
- unauthorized privileged users appear on your system
- queues are stopped or stalled
- a user runs with a specified privilege such as SETPRV or BYPASS
- users bump up their priorities
- specified users log on (or off)
- someone logs into a restricted terminal
- users waste resources playing games
- a process is using excessive CPU cycles, buffered or direct I/Os

When HITMAN detects an event it records the event in a log file and can:

- take corrective action by submitting a batch procedure
- mail a message to a user or distribution list
- send an operator message to a specified operator class

Set it up and forget it

HITMAN is a "set it up and forget it" tool. Once you define your site-specific parameters, it kills idle users and keeps an eye on your Alpha AXP or VAX without making any demands on your time. It's like having an assistant who works 24 hours a day and always follows your orders. You will wonder how you ever got along without it!

A Deadly Game

With HITMAN, you finally have an effective way to control game playing on your Alpha AXP or VAX. With HITMAN's powerful AUTOHIT facility, you can terminate specified images regardless of activity. For example, when multi-user STAR TREK appears on your system, you can hit it immediately with or without warning regardless of activity. You can define any number of auto-hit images. HITMAN nails them whenever it sees them. No more resources wasted playing games during prime time!

Hackers Beware

You can use HITMAN to guard against unauthorized privileged users who appear unexpectedly on your system. If a user with any privilege in the ALL category appears on your Alpha AXP or VAX, and their username is not on HITMAN's list of authorized users, they are terminated immediately. Privileged accounts are automatically exempt from this scrutiny. Hackers beware - HITMAN is always on the job.

HITMAN is easy to learn and use

HITMAN is easy to learn and use since it looks and acts like the OpenVMS utilities you use every day. OpenVMS standards and conventions have been followed closely.

HITMAN commands look like regular DCL commands. Command defaults have been selected to minimize keystrokes and errors. And, you can use a menu-driven screen interface to modify any of HITMAN's parameters. HITMAN also provides comprehensive, on-line OpenVMS-style HELP. You can access the help at the DCL level or through the menu-driven, screen interface. With the choice of a screen interface or a DCL-like command line interface, plus on-line HELP, HITMAN is a snap to use.

HITMAN is safe to use

HITMAN is safe to use because it checks only interactive processes for inactivity by default. Detached, network and batch processes are automatically protected unless you tell HITMAN otherwise. And, to ensure it kills only truly inactive processes, HITMAN checks for Buffer I/O (BIO) and Direct I/O (DIO) activity as well as CPU time.

Your data is protected because HITMAN allows images to write their data buffers to disk and release their locks before it terminates them. You can even control the number of force exits sent to the process and the number of seconds the process is given to run-down before HITMAN terminates it.

HITMAN understands processes and sub-processes.

If an idle parent process has one or more active sub-processes, none of the processes get hit. On the other hand, if the whole family is idle, they all get terminated.

If you are using a software package which creates detached processes rather than sub-processes, you can link the different processes together through a common field such as UIC, server or username so HITMAN will consider them all part of the same family of processes.

You can even specify your own procedure for handling idle processes using HITMAN's /SPECIAL_EXIT table.

HITMAN is fully supported

You will always have someone to call if you have any problems with HITMAN. With an annual software support agreement, you get unlimited TOLL-FREE telephone support from our experienced software support staff as well as automatic software updates. Our support lines are open Monday to Friday from 8 AM to 4:30 PM Mountain Standard Time except normal business holidays. Just dial 1-800-561-8876 or e-mail us at support@saiga.com.

HITMAN is field proven

HITMAN is eliminating waste, plugging security leaks and monitoring systems at over 3,000 Alpha AXP/VAX/OpenVMS sites in the United States, Canada, Europe, Asia, the Caribbean and Australia. It's solidly engineered, extensively tested and fully field-proven by systems managers like yourself.

You are in Control

You tell HITMAN how long to wait before it strikes.

You set a system-wide maximum idle time which applies to every interactive process on your system. And, you can set different idle times for any number of individual users as well as UICs, images, terminals, identifiers, terminal server ports, processes and paths. You can even set idle times for DIALUP processes separate from LOCAL so you can free idle modems for other users! With Version 10 of Hitman these lists can be any size.

They Were Warned

HITMAN lets you warn users before you strike. This allows users to avoid termination if they are only momentarily inactive.

You can send users one or two warning messages. You can tell HITMAN when to send the messages as well as define the text of each message. You can also send a warning message repeatedly until the user becomes active or is terminated.

As with idle times, you can specify warning times for individual users, UIC's, images, terminal server

ports, identifiers, terminals, processes and paths; even DIALUP as opposed to LOCAL processes.

HITMAN sends a termination message to the users it kills. If your site needs tight security, HITMAN can also clear the screen before it sends the message.

And it can even mail terminated users a detailed message so they will know what happened when they log back in.

You control who gets hit
You can protect any number of users so they will not be hit even if they are completely inactive. You didn't really want to shoot the president's ALL-IN-1 process did you?

You can also protect UIC's and UIC groups. This allows you to protect projects or classes of users with a single command.

You can prevent data integrity problems by protecting sensitive database applications. HITMAN allows you to protect as many separate images or process names as you require regardless of who is running them.

If your users are really cranky, you can protect processes that are running any image at all. The only users who will get logged off are those sitting at the DCL prompt. This guarantees you will not cause problems for users.

Finally, HITMAN allows you to protect terminals and servers, and automatically protects any process running on the console terminal (OPA0:).

You can use wildcards when you protect users, UIC's, images, terminals, processes or server ports.

HITMAN, by default, protects non-interactive processes such as BATCH or DETACHED or NETWORK jobs.

They can't hide
Sneaky users cannot hide from HITMAN. Even if they use a small amount of resources (such as updating a screen clock) to look active, you can set default thresholds on CPU, BIO's and DIO's to catch them. They cannot hide from HITMAN! You can also set thresholds for specific images.

Test HITMAN before you turn it loose.
You can prevent user complaints by testing HITMAN on your system before you turn it loose. When you run HITMAN in WATCH mode, messages are written to the log file but users are not warned or killed. When you run it in WARN mode, processes are warned but they are not killed. HITMAN's log file will show what HITMAN would have done. When you have decided who to protect and what

warning times to set, you can switch HITMAN into HIT mode where users are warned and killed.

HITMAN V10 is the first version of Hitman to run on only later versions of OpenVMS
Because of changes made to the compilers that supported adding dynamic tables to HITMAN Version 10 it is the first Version of Hitman to have a minimum Version of OpenVMS requirement. Hitman V10 requires OpenVMS 7.1 or higher and it requires Version 7.3 or higher of the OpenVMS Fortran Run-time Libraries (RTL) which are distributed with the operating system. At the time this manual was produced Hitman Version 10 was only available for AXP systems. A VAX version is currently under development.

HITMAN runs on all Alpha AXP models. It also runs on Cluster systems. You can share parameter files across a cluster or have separate settings for each node.

Stores your site-specific information
HITMAN stores your site-specific information such as protected users, idle times and events in a parameter file so you will not have to re-enter it if you stop HITMAN or your system crashes. You can define prime and non-prime periods of the day and set up a different parameter file for each. For example, you might want to terminate idle processes only at night. HITMAN will automatically load the appropriate file when it passes from prime to non-prime or vice-versa.

Provides a complete record

You will always know what HITMAN is doing on your Alpha AXP or VAX because it writes complete details of hits, warnings and events to the log file on disk. The messages show the date, time, username, terminal and image name of each process warned or terminated.

You can use HITMAN's reporting facility to select records by user, date or type of message from the log file, sort them in order by user, date, etc. and list them on your printer. Or generate a /ASCII report of HITMAN's log that can easily be imported into your own graphics and report writing software.

HITMAN will also send messages to the operator console and all users enabled as security operators. If you do not want a lot of message traffic to your console, you can select only the types of messages you want to receive or change the class of operator HITMAN sends the message to. You can even turn off console messages completely if you wish.

HITMAN will not bog down your system

HITMAN consists of an interactive process which communicates with you and a detached process which monitors your system. On most systems the detached process uses less than 1% of your CPU resources. Most of the time, you will barely notice the resources it uses.

HITMAN requires less than 10,00 blocks on your disk. And, you can install it on any disk you

choose. Installation using VMSINSTAL is easy - it only takes minutes.

HITMAN, the safe, effective idle process killer and system monitor. You will wonder how you ever handled security and monitored resources without it!

Installing HITMAN

Installing HITMAN is simple using the VMSINSTAL utility provided by Digital. For complete installation instructions refer to Saiga Systems' Product Installation and Release Notes Guide. The release notes file included in the distribution kit also includes installation instructions as well as output from a sample installation and a sample upgrade. To extract these release notes from the VMSINSTAL kit for Hitman V10 use the following command; be sure to specify where the distribution files are located on your system at the spot indicated *kit-source*.

$ @SYS$UPDATE:VMSINSTAL HITMAN0100 *KIT-SOURCE* OPTIONS N

you will then be given the choice of viewing the release notes, printing the release notes, both viewing and print, or skipping this step and continuing with the installation. A sample run to extract release notes is included below:

Overview of HITMAN's Major Functions

The following pages provide an overview of HITMAN's primary functions and logic.

1. Use the DCL level or HITMAN's menu interface to maintain HITMAN's prime and, optionally, nonprime data files. These files contain all the site specific parameters.
2. Use the DCL level or HITMAN's menu interface to start HITMAN's detached process.
3. HITMAN performs a data collection which involves:
    1. HITMAN checks its' mailbox for messages and reads the current data file again, if any changes have been made.
    2. If /prime_enabled is in effect HITMAN determines if it should switch to the other data file and, if so, reads the new parameters.

```
$ @SYS$UPDATE:VMSINSTAL HITMAN0100 DKA0:[SHARES] OPTIONS N

        OpenVMS AXP Software Product Installation Procedure V7.3

It is 22-MAY-2002 at 13:22.
Enter a question mark (?) at any time for help.

* Are you satisfied with the backup of your system disk [YES]?
The following products will be processed:
  HITMAN0 V10.0

        Beginning installation of HITMAN0 V10.0 at 13:22
%VMSINSTAL-I-RESTORE, Restoring product save set A ...

    Release notes included with this kit are always copied to SYS$HELP.

    Additional Release Notes Options:
        1. Display release notes
        2. Print release notes
        3. Both 1 and 2
        4. None of the above

* Select option [2]: 2
* Queue name [SYS$PRINT]:
Job HITMAN0100_RELEASE_NOTES (queue SYS$PRINT, entry 59) started
* Do you want to continue the installation [NO]? NO
%VMSINSTAL-I-RELMOVED, Product's release notes have been moved to SYS$HELP.

        VMSINSTAL procedure done at 13:23
```

3. If /OFF is in effect HITMAN skips everything else and hibernates for /interval minutes then wakes up for the next data collection (item 3).
4. HITMAN collects data on all processes and updates its internal tables.
5. If /USER_EXIT is in effect and there are special_exits defined HITMAN calls the User Exit Module. The default user exit finds all processes with a current path name that matches a value in the file and submits the specified procedure if a match is found
6. A batch job is submitted for each of these processes and they are marked as protected so they are not terminated by the regular module.
7. HITMAN then executes its warning/killing module for the first time to kill users. It does the kill loop before the warn loop to minimize the possibility of users resuming activity after being found idle (all protected processes are ignored during this function). If a process should receive warning messages, and has not yet been warned, the first warning message will be sent in this loop and the process will not be terminated until the next data collection.
    i. It checks for processes that should be killed because they are running an image on the AUTOHIT list, logs and then kills them.
    ii. It checks for processes that should be killed because they are running with privileges and are not on the AUTHORIZED list, logs and then kills them. This check can be optionally disabled.
    iii. It checks for DECWindows users that have exceeded their idle time limit in all related processes then logs and submits the Hitman$decwindows_command_procedure procedure.
    iv. It checks for all users that have exceeded their idle time, logs and then kills them.
    v. It checks for DECWindows users that have exceeded their connect time limit in all related processes then logs and submits the HITMAN$DECWindows_command_procedure procedure.
    vi. It checks for all users that have exceeded their connect time, logs and then kills them.
8. HITMAN then executes its warning/killing module again to warn users (all protected processes are ignored during this function)
    i. It checks for DECWindows users that have exceeded the first or second warning times in all related processes then logs and warns them.
    ii. It checks for all users that have exceeded the first or second warning times, logs and then warns them.
    iii. It checks for DECWindows users that have exceeded their first or second connect warning times in all related processes then logs and warns them.
    iv. It checks for all users that have exceeded their first or second connect warning time, logs and then warns them.
9. HITMAN then loops through all defined events.
    i. If any of them are “true” HITMAN then

(1) Logs the event in the log file.
(2) Checks if mail should be sent and sends it if the number of times the event has been signalled is less than the value specified for messages in the event or if messages has been set to 99.
(3) Checks if operator messages should be sent and sends one if the number of times the event has been signalled is less than the value specified for messages in the event or if messages has been set to 99.
(4) Checks if there is an action procedure that should be submitted and if the number of times the procedure has been submitted is less then the number of times requested in the event HITMAN submits the procedure.

ii. HITMAN increments a counter for the event to track how many times in a row the event has been seen "true". If the event is "false" HITMAN sets this counter to 0.

10. HITMAN then hibernates for /interval minutes before waking up to perform the next data collection.

How HITMAN terminates processes

The warn and kill module in HITMAN takes the following steps to terminate processes regardless of why they are being terminated. Please remember that if a user is protected for any reason HITMAN will never terminate their process.

The termination qualifier (/termination=[delete_process][forced_exit][both]) controls which systems services are called while HITMAN is terminating users. The default is both and the default value for HITMAN$MAX_FORCE_EXITS is 2 if it has not been defined.

HITMAN loops through all processes in its process table checking for processes that should be terminated. This loop is repeated up to a maximum of HITMAN$MAX_FORCE_EXITS or until it has looped through /force_wait times, whichever is greater. HITMAN sleeps for 1 second between each loop.

For each process that is eligible to be hit HITMAN checks if they are at the DCL level (no current image running). If the processes image name is blank HITMAN considers it to be at the DCL level and calls the system service to delete that process. If there is an image name and HITMAN$MAX_FORCE_EXITS count has not been exceeded HITMAN calls the system service to send a forced exit. The /noimage_name qualifier can be used to tell HITMAN to only hit processes at the DCL level and ignore processes that have a current image.

During the last loop any processes that should be deleted and have not yet been deleted are. This kills processes that are running images that don't respond to forced exits.

New features of HITMAN V10.0

* The new /table_size qualifier allows users to configure how much table space the Hitman detached process allocates. Previously Hitman allocated space for 2,000 processes; users requiring more space needed to download a special kit built for more processes. Users running Hitman on workstations can now save memory by configuring Hitman to allocate space for significantly less than 2,000 processes. Refer to the /table_size qualifier for a detailed explanation and examples.

* New with Version 9 was the /summary qualifier used to generate a summary of any HITMAN log file. In Version 10 this report is also available through the menu interface.

* The single most customer requested enhancement since V6 of Hitman was released has been incorporated into V10. All protection lists, special idle and warn time lists, special connect time lists, the event list, the holiday list, the list of users allowed to set their own idle times, the list of users allowed to have privileges and the list of images that should be automatically killed are now dynamic. You can add as many entries to these tables as you require; they are no longer limited to 256 entries/table. Smaller or empty tables should use significantly less memory resources.

  NOTE: The menu interface, which uses the SMG routines provided with OpenVMS, does not support dynamically expanding tables. When using the menu to update any of these lists there is a restriction to 2,048 entries. Users requiring more entries than this in a specific table must update the table using the DCL interface.

* Process names can now be used in the idle and warn time lists as well as the connect time lists. Previously process names could only be used in the protection lists. Refer to the /process qualifier.

* With Version 3 of Cohort a list server was created for each of the 7 products included in Cohort; the Hitman list was called hitman@saiga.com. Recently we merged all 7 of these lists to create a single list cohort@saiga.com. Users wishing to be on this list should send an e-mail to cohort@saiga.com. This list is used to send product and patch announcements for Saiga Systems products. Users subscribing will receive no more than 1 e-mail per month. The list is moderated and interesting customer comments or the answers to frequently asked questions may also be posted. Users wishing to post a question to other Hitman users may send their question to the list server.

* Detailed termination debugging has been made available with a new termination logging option. When this option is enabled every step of process termination is logged to help track where problems may be occurring. Refer to the troubleshooting section.

* It is now possible to include a username in the events that check for absent (or present) processes or batch jobs. Using this functionality with absent events will cause the event to be triggered if the batch job or process is not found on the system or is not found on the system running under the specified username. Using this functionality with present events will causes the event to be triggered if the batch job or process is found on the system running under the specified username. In either case if the username is set to the wildcard character "*" or blank "" the event will behave as it did previously and will not consider username as a criteria when determining if a batch job or process is absent or present.

* It is now possible to create events to monitor your system for CPU or MEMORY errors. To create these events add a device error event and use CPU: or MEMORY: for the device name respectively.

* Many users have mentioned that receiving broadcast messages while in the menu can become inconvenient on systems where a large number of messages is typical. On the main menu an option has been added to disable broadcast messages for your current session, while in the menu.

* Unlimited special user exit routines can now be defined. V8 and V9 of Hitman allowed sites to set up only one special exit that bypassed Hitmans' process termination routine. This exit was configured using logical names to define the image (wildcards were supported) to watch for. When an idle process, running that image, was discovered instead of killing the process Hitman would use the values specified in a number of logical names to submit a batch job, under the specified username, to a specific queue so that site specific action could be taken. In V10 this user exit configuration data is now included in the permanent data files instead of using logical names. It is also possible to define as many user exit checks as desired. To make this functionality more useful we have changed the exit routine from checking for specific images to checking for specific paths (ie: disk:[directory]image.exe); file version number is ignored.

  The new /special_exit qualifier has been defined to add these exits. The keywords used with this qualifier are:

  path=image_to_use_exit_for
  node=node_to_watch_on
  job=job_to_submit_when_a_user_is_idle_in_this_image
  username=username_to_submit_batch_job_under
  queue=queue_to_submit_batch_job_to
  debug=t_or_f

  Refer to the section on user_exits for a more detailed explanation and examples.

  NOTE: The special exit functionality is enabled/disabled using the /user_exit qualifier. Be sure to enable this functionality when using special exits.

* It is now possible to set protections, idle and warn times or connect times for a specific image by its complete path (excluding version number). This change addresses a frequent customer request to allow protection of a production copy of an image while allowing testing with a development copy. Refer to the /path qualifier.

* It is now possible to turn off the mail username check for Hitman events. Any username entered in an event as a mail recipient, or in a mail distribution file, that begins with a colon (":") will be treated as literal and passed through to the OpenVMS mail callable interface exactly as entered. This should allow users with SMTP mail services on their system to have mail sent to nonOpenVMS mail accounts including, at many sites, internet mail accounts.

Learning to use HITMAN

This chapter shows you how to use HITMAN. If you have just installed a demo or have not used HITMAN before, please read this chapter and do the requested exercises. The text you enter is displayed in ***bold italics***.

NOTE During this tutorial, we will make a number of changes to the permanent data file. If you want to end the tutorial, you must recreate a default permanent data file. The procedure to do this is given at the end of the tutorial.

Tutorial

Verify that HITMAN logicals are defined

HITMAN uses five logicals to access data files and executables. Normally, these logicals are defined as part of the system startup. However, to ensure the logicals are defined, type the following command:

$ *@SYS$MANAGER:HITMAN_SYSTEM_LOGICALS*

This procedure is created as part of the HITMAN installation procedure. If the logicals were already defined, you will see the following messages:

%DCL-I-SUPERSEDE, previous value of HITMAN_CDU has been superseded
%DCL-I-SUPERSEDE, previous value of HITMAN_COM has been superseded
%DCL-I-SUPERSEDE, previous value of HITMAN_DAT has been superseded
%DCL-I-SUPERSEDE, previous value of HITMAN_DOC has been superseded
%DCL-I-SUPERSEDE, previous value of HITMAN_EXE has been superseded

To verify that the HITMAN logicals are defined, type the following command:

$ SHOW LOGICAL HIT*

You should see output similar to the following:

(LNM$PROCESS_TABLE)

(LNM$JOB_xxxxxx)

(LNM$GROUP_xxxxxx)

(LNM$SYSTEM_TABLE)

```
"HITMAN_CDU"     = "ddcu:[HITMAN.CDU]"
"HITMAN_COM"     = "ddcu:[HITMAN.COM]"
"HITMAN_DAT"     = "ddcu:[HITMAN.DAT]"
"HITMAN_DOC"     = "ddcu:[HITMAN.DOC]"
"HITMAN_EXE"     = "ddcu:[HITMAN.EXE.VAX]"
```

where "ddcu:" is the disk drive on which HITMAN is installed. This output indicates that the HITMAN logicals are properly defined. Please note that the HITMAN logicals **MUST** be defined in the SYSTEM table so that the detached process can access them.

Verify that your HITMAN or Cohort license has been installed

Saiga Systems uses a licensing utility that creates a logical name containing the license information. HITMAN first checks to see if a HITMAN license is installed, then checks for a Cohort and will fail to start if it doesn't find either license. To verify that your license is installed:

```
$ SHOW LOGICAL PKMS*
```

you should see a clearly defined logical for either HITMAN or Cohort. If you don't you can reload your license key by entering:

```
$ @SYS$MANAGER:SAIGA_LICENSE
```

and entering the values you are prompted for from the license key. The license utility is case sensitive, please enter the values exactly as they are printed on your license. Be sure to answer Yes when asked to load the license and if the utility should create a license startup file.

Verify that your process has the necessary privileges

HITMAN consists of an interactive program and a detached process which monitors your system. Both of these programs require OpenVMS privileges to run.

To modify or list HITMAN parameters, you need the privileges
SECURITY and SYSPRV.

To start the HITMAN detached process, you need either:

SETPRV

or

ALTPRI, CMKRNL, DETACH, EXQUOTA, OPER, PRMMBX, SYSNAM, SYSPRV and WORLD

To set your privileges correctly if you're not certain you have the necessary privileges you may:

$ hitman_com:set_prv

We recommend using the SYSTEM account which has all the necessary privileges to run HITMAN since the detached process will be started under the SYSTEM account during reboots.

If your process does not have all the necessary privileges, HITMAN will list the additional privilege(s) you need whenever you enter a command. Here is a sample error message indicating that the process does not have the DETACH privilege:

%HITMAN-E-REQDETACH, Operation requires DETACH privilege

To find out what privileges your process has, type the command:

$ SHOW PROCESS/PRIVILEGE

To give your process specific privileges, type the command:

$ ***SET PROCESS/PRIVILEGE=privilege_name***

For more information on the SET and SHOW commands, consult the OpenVMS DCL Dictionary.

Controlling HITMAN

By default, HITMAN waits 30 minutes before it kills an idle user. While this is acceptable for a production version, it is too long for the purposes of this tutorial. Please type in the following command:

$ @HITMAN_COM:TUTORIAL1

This command procedure puts HITMAN in WATCH or simulation mode in which HITMAN appears to "warn" idle users after 2 minutes of inactivity and "kill" them after 4 minutes. In WATCH mode, HITMAN writes records to the log file but does not affect processes. **NO PROCESS WILL BE WARNED OR KILLED DURING THIS TUTORIAL**. When you execute the command procedure, you will see the following message three times:

%HITMAN-I-NOTRUN, HITMAN is not running

This message is informational. It does **NOT** represent an error. Please ignore it.

You can control HITMAN using a menu-driven screen interface or a command line interface similar to DCL. The screen interface is easier to use but somewhat slower. The command line interface is fast and convenient but requires that you know the name and syntax of HITMAN qualifiers and limits you to command lines of no more than 256 characters (the DCL maximum) which can make it difficult to

define complex events at the command line level. Most people start with the screen interface and switch to the command line interface as they become more experienced with HITMAN. This tutorial uses the screen interface and gives the command line equivalent.

To access the HITMAN main menu, type the following command:

$ HITMAN/MENU

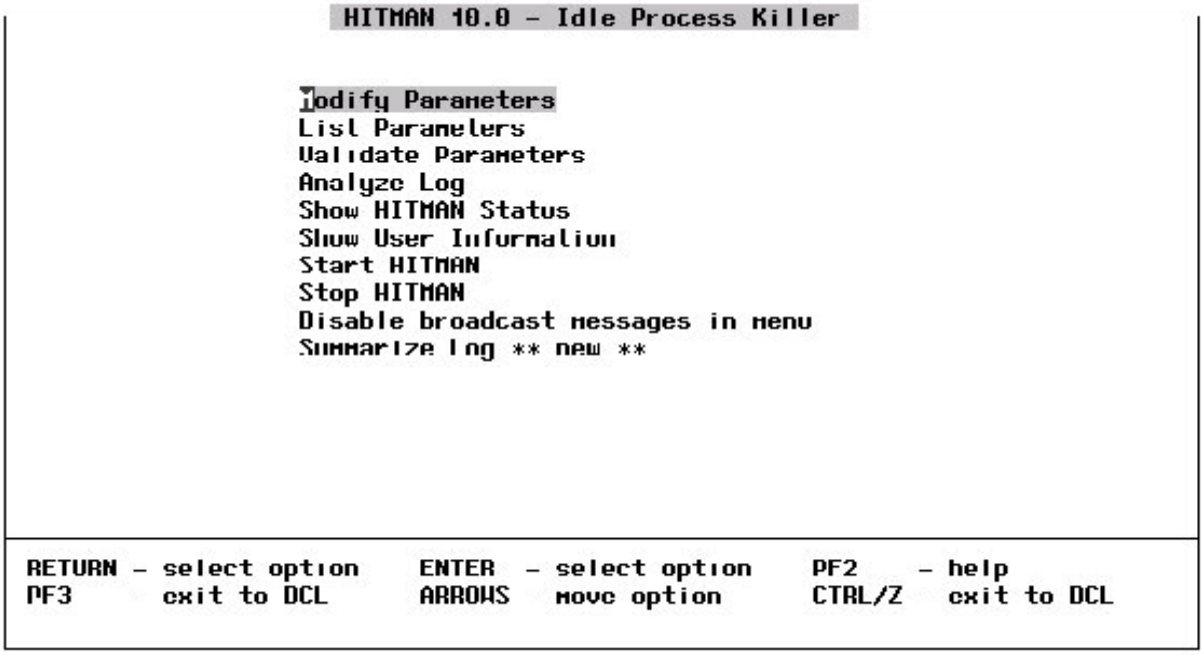
If you receive the following message:
%DCL-W-IVVERB, unrecognized command verb - check validity and spelling
\HITMAN\

HITMAN's command definition has not been installed. To add the command definition to your process table, type the following command:

$ SET COMMAND HITMAN_CDU:HITMAN

then re-enter the "HITMAN/MENU" command. The HITMAN main menu should be displayed on your terminal. You may wish to add the above command to the login.com file for any accounts which might be routinely used to control Hitman.

This is an example of the HITMAN main menu. Press the up and down cursor keys to move up/down within a menu. The current menu function is always highlighted in reverse video. Menus do not "wrap".



Pressing the down cursor at the bottom of menu or the up cursor at the top leaves the menu unchanged. Press <enter> to select a menu function.

Press [PF2] to get HELP. If you are in a menu, you will receive HELP on the menu and the currently highlighted option. If you are in a data entry screen, you will receive HELP on that screen.

Press [PF3] to cancel an operation or return to the previous menu.

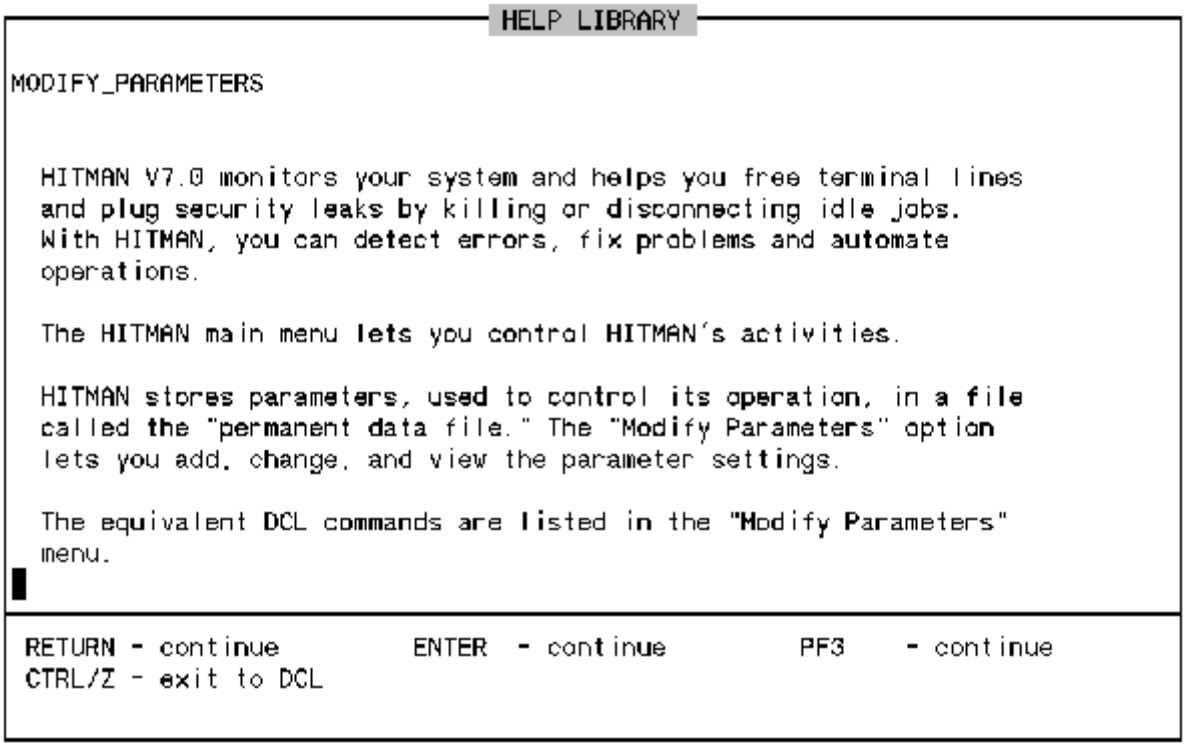
Press <CTRL/Z> to exit the HITMAN screen interface and return to the DCL $ prompt. HITMAN updates the permanent data file and exits.

There is a list of all the menus which can be called from the main menu near the end of this chapter.

Getting HELP
HITMAN has on-line HELP. To see a HELP screen at any point, press [PF2].

Because the highlighted menu selection is "Modify Parameters", HITMAN displays the HELP text for "Modify Parameters". To exit HELP, press <enter>.

To get help at the command line, type "HELP HITMAN" or "@HELP HITMAN".



Starting HITMAN

HITMAN consists of an image that executes under your process to set and modify parameters and a detached process which monitors your system. To start HITMAN (the detached process), select option "Start HITMAN" by pressing the cursor down key until the "Start HITMAN' option is highlighted in reverse video then pressing the <enter> key. You will see the following message:

Starting HITMAN, Please wait ...

After a few seconds, you will see the message:

Identification of created process xxxxxxx

showing the process identification (PID) of the detached process which monitors your system. HITMAN is now running on your system in WATCH mode with a default maximum idle time of 4 minutes. WATCH mode means that messages are written to HITMAN's log file but idle users are not warned or killed. No one knows HITMAN is running on your system. We recommend that you test HITMAN in WATCH mode for a few days so that you can see how it acts on your system.

NOTE: As part of its processing, HITMAN can submit batch jobs. To ensure these batch jobs have access to command procedures in the HITMAN directory, start HITMAN from an account which has SYSPRV as a default privilege.

Exit the "Start HITMAN" function by pressing the <enter> key. This returns to the HITMAN main menu.

To start HITMAN using the command line interface, you type:

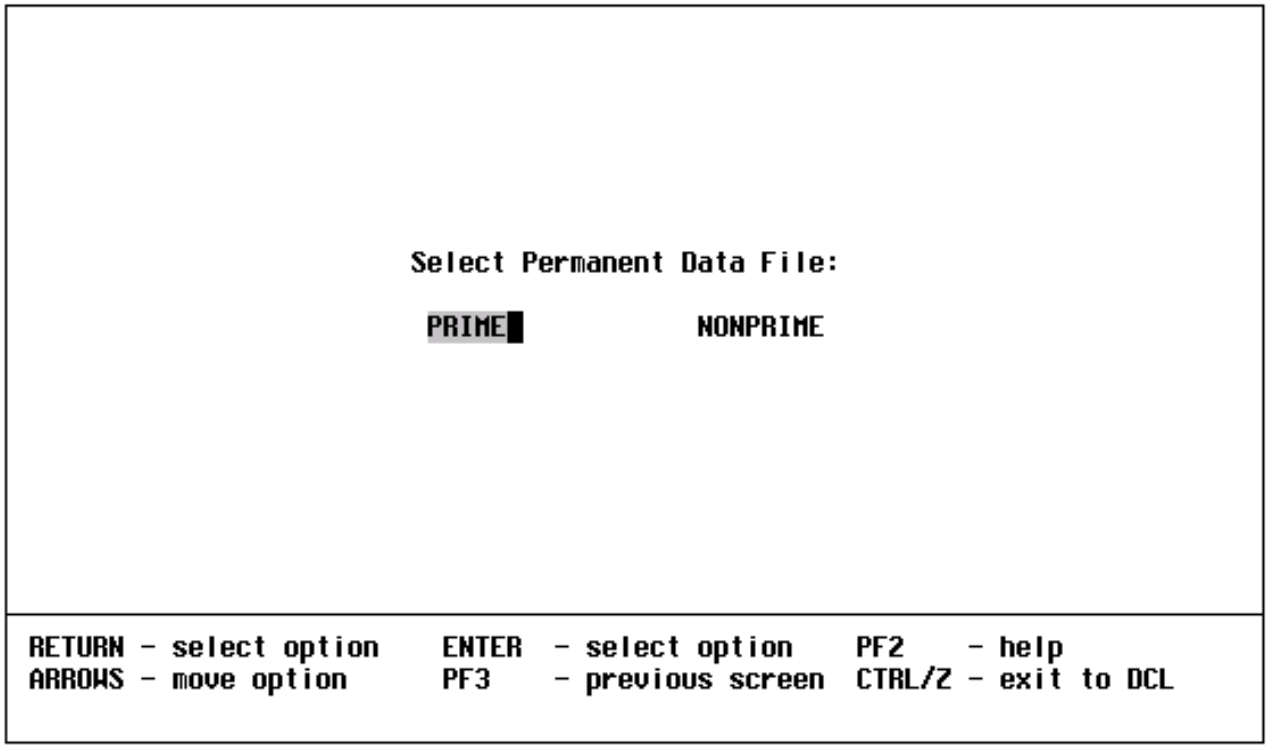
$ HITMAN/START

After a few seconds you should receive the following messages at your terminal:

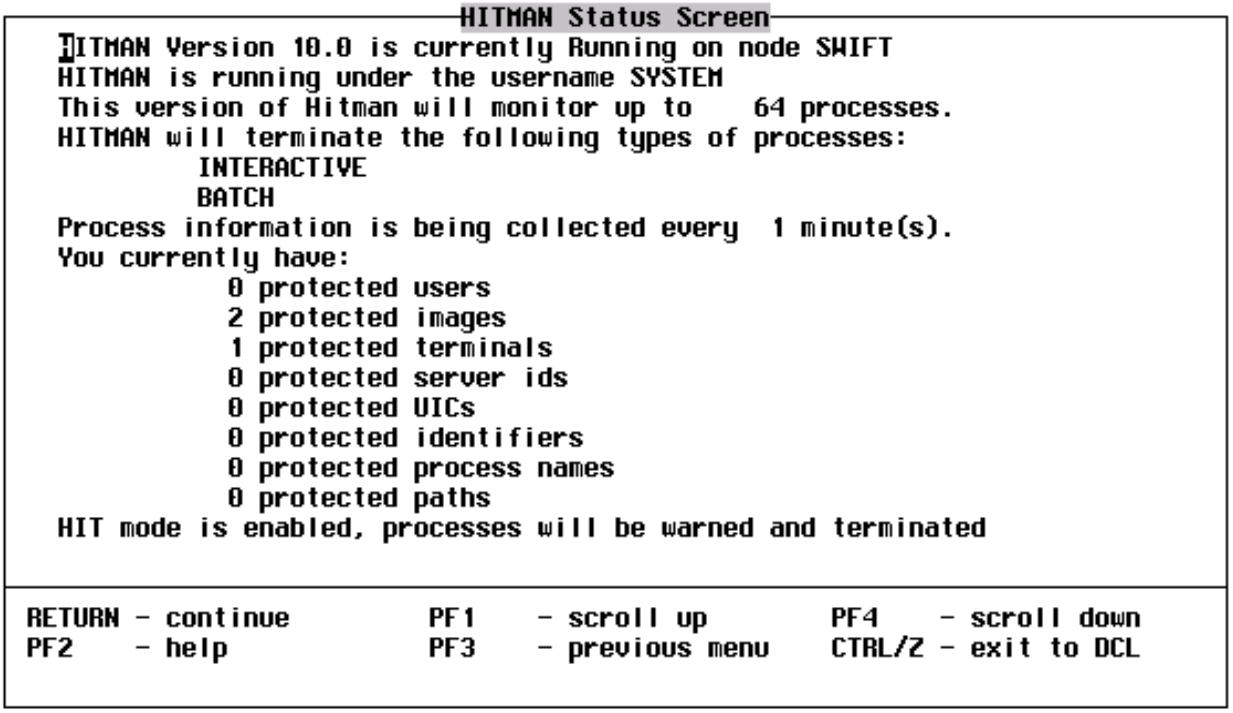
Showing HITMAN's status

The HITMAN status command allows you to determine if HITMAN is running and what parameters it is using. To see HITMAN status information, select option "Show HITMAN Status" by pressing the cursor up key until the "Show HITMAN Status" option is highlighted then pressing the <enter> key.

HITMAN stores parameter settings in a file (or files) called "permanent data files". When HITMAN asks you to select a permanent data file, press <enter> to select the prime file which is the default. The status screen should show that HITMAN V10.0.0 is running in WATCH mode on your system.



Exit the "Show HITMAN Status" function by pressing the <enter> key.



To show HITMAN status using the command line interface, you would type:

```
$ HITMAN/STATUS
```

Use the cursor up and cursor down keys or PF1 and PF4 to scroll through the output which is larger than the displayed window.

Analyzing HITMAN's log file

HITMAN writes a message in the log file every time it warns or kills a process. In the next step of the tutorial, we are going to analyze the HITMAN log file. We have set HITMAN's default idle time to 4 minutes with a first warning message at 2 minutes. Please wait at least 15 minutes to ensure that HITMAN has time to identify idle users and write a few messages in its log file. If you want to do other tasks while you are waiting, simply exit HITMAN by pressing [PF3]. HITMAN will continue to monitor your system.

Please wait at lesat 15 minutes
Remember, HITMAN is in WATCH mode.

No users will actually be warned or killed

Stopping HITMAN

Although you can analyze the log file while HITMAN is running, we are going to stop HITMAN while we look at the log file.

If you are at the DCL "$" prompt, type "HITMAN/MENU" to see the HITMAN main menu.

If you are at some menu other than the main menu, press [PF3] until you see the main menu.

To stop HITMAN, select option "Stop HITMAN" by pressing the cursor up/down keys until the "Stop HITMAN" option is highlighted then pressing the <enter> key. You will see the following message:

Stopping HITMAN, please wait ...

After a few seconds, you will see the message:

Press <enter> to continue ...

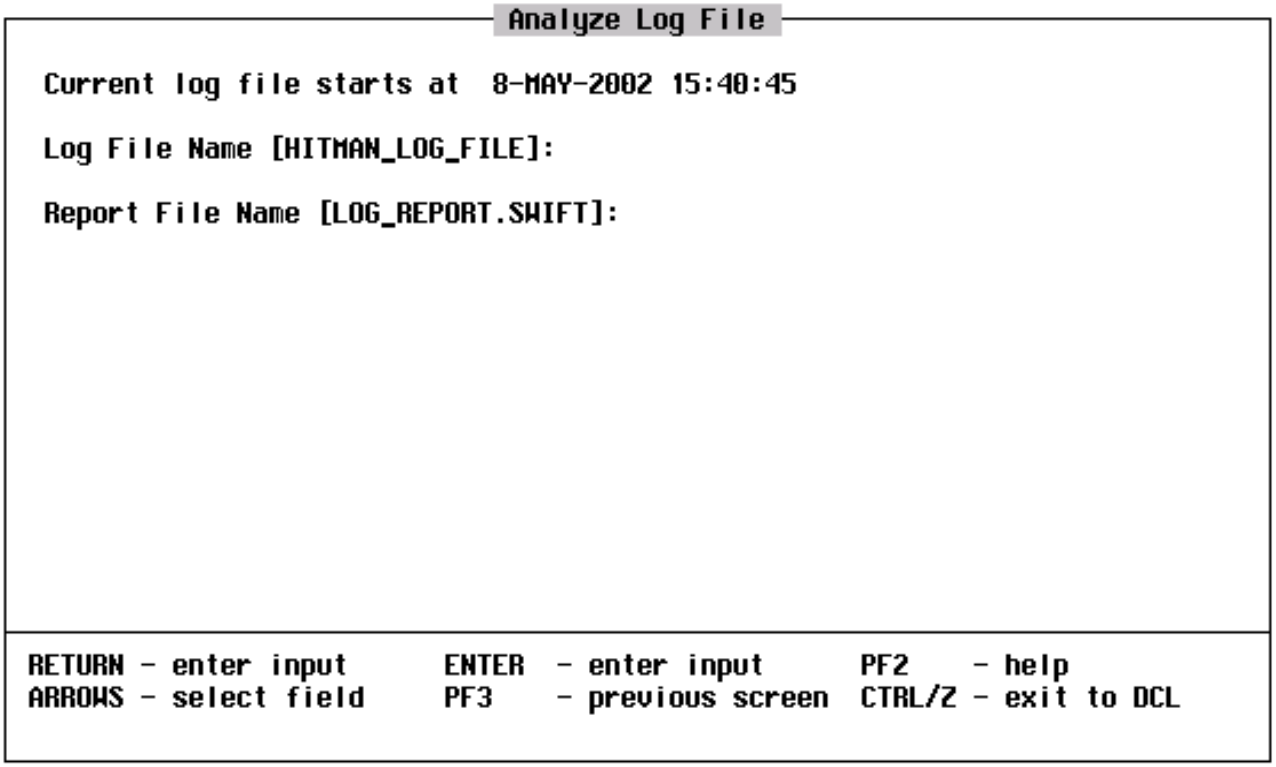
Exit the "Stop HITMAN" function by pressing the <enter> key. HITMAN is now stopped and the log file has been closed.

To stop HITMAN using the command line interface, you would type:

$ HITMAN/STOP

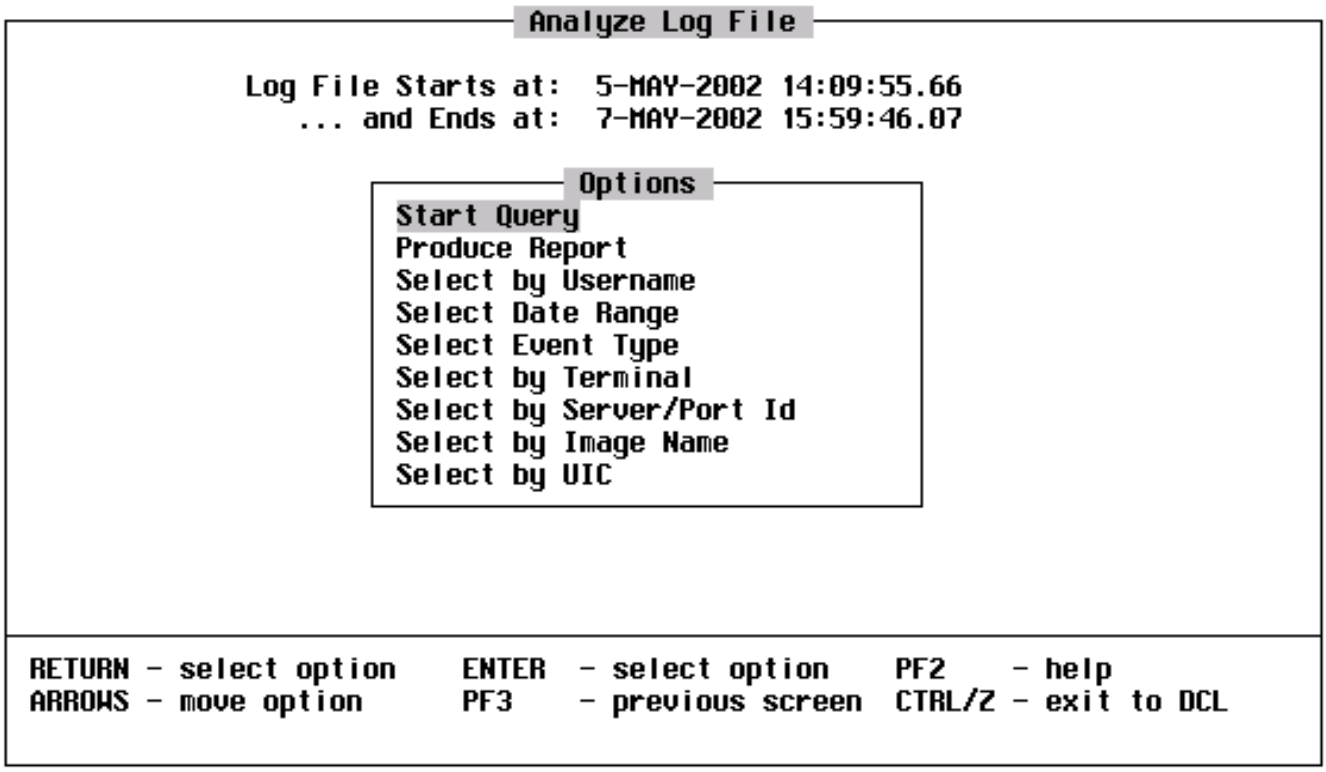
Now, we are going to look at the contents of the log file HITMAN generated.

Select option "Analyze Log" on the HITMAN main menu.



HITMAN will ask you for the name of the log file and the name of the output report file. Press <enter> to take both defaults. (Note that the log file extension is the same as the node name on which HITMAN is running. A HITMAN log file is specific to the node HITMAN is running on.)

Using the "Analyze Log" function, you can select records from the log file using criteria such as username, date and type of record. For the moment however, we will look at all the records. You can view the log file records at your terminal ("Start Query") or generate a file ("Produce Report") in the directory pointed to by the logical HITMAN_DAT. Select the "Start Query" option to see the records on your screen.



The following samples show what a first_warning log message and a termination message look like on the log report. Only part of this information is displayed on the screen when you specify "Start Query".

```
** 06-APR-2002 19:31:55.67 ************************ First Warning Message **
      Username: FRED               PID: 0000033F   Hit Mode: WATCH
  Process Name: FRED                               Terminal: LTA5098
    Image Name: GL_MENU                                 UIC: [00400,000401]
Server/Port Id: DS004/Port_7                         Subproc: 0
  Idle Minutes:  2     First:    2      Second:    0   Final:  4
      Set Idle:  F     Disconnect:  F        Disconnect Time: 0

** 06-APR-2002 19:31:55.67 *********** User Termination, Wait Time Exceeded **
      Username: FRED               PID: 0000033F   Hit Mode: WATCH
  Process Name: FRED                               Terminal: LTA5098
    Image Name: GL_MENU                                 UIC: [00400,000401]
Server/Port Id: DS004/Port_7                         Subproc: 0
  Idle Minutes:  4     First:    2      Second:    0   Final:  4
      Set Idle:  F     Disconnect:  F        Disconnect Time: 0
```

Of course, if no user was warned or killed during the short time HITMAN was running, there will not be any log messages. On the log report, the "Hit Mode" field indicates HITMAN is in WATCH mode so no process will actually be warned or killed even though the log messages say they were.

Press <enter> to exit the "Analyze Log File" and return to the main menu.

Another useful tool is the Log File Summary Report which summarizes all the records included in the log in a concise report. For example to summary the previous Hitman log file from the DCL level use the command:

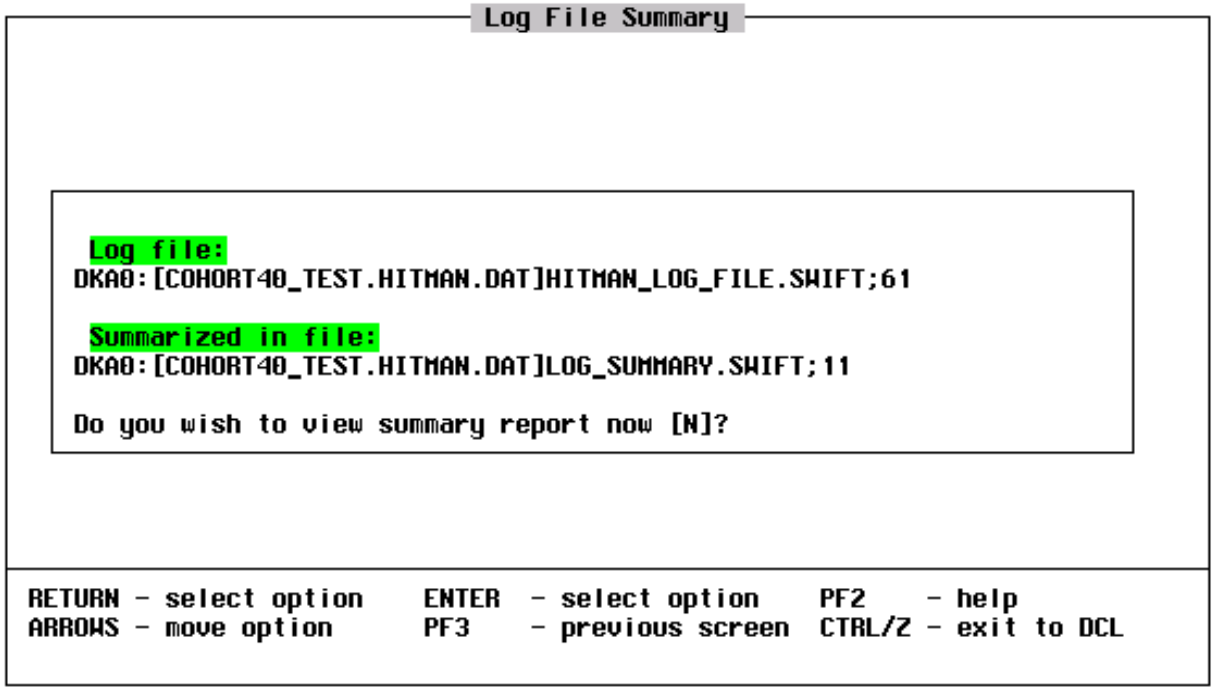
$ HITMAN /SUMMARY=HITMAN_DAT:HITMAN_LOG_FILE.node;-1

the output will be written to hitman_dat:log_summary.node.

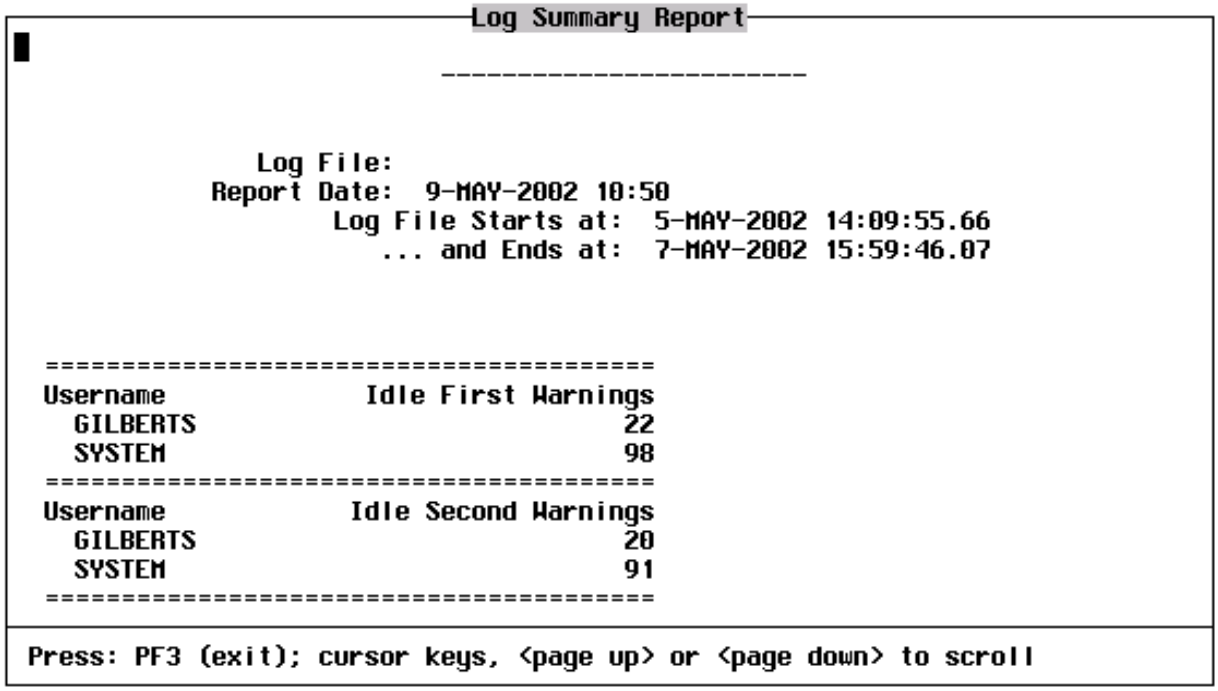
This report is also now available from the main menu. Select the last option on the main Hitman menu "Summarize Log ** New **". The next two screens will resemble the screens for analyzing the log and allow you to select the log file you wish to summarize as well as specify a report name if you don't wish to use the default name. Now selection criteria are available for a summary report so the only option is "Produce Report."

A screen is displayed showing the log file that was summarized and the name of the output report. You then have the option of exiting or viewing the summary report immediately within the menu.

If you answer "Y" the report will be loaded into the window and you may use the scroll keys to navigate through the report and view the summary information.



The <PF3> key or <CTRL/Z> can be used to exit the report display.



Listing Parameters

To list the HITMAN parameters, select the "List Parameters" option and select the prime permanent data file. View the parameters on your screen or generate a file which you can print. The report file is generated in the directory pointed to by the logical HITMAN_DAT. The list of HITMAN parameters fills several screens. HITMAN pauses after each screen to let you read the information.

To see the next screen, press [PF4].
To see the previous screen, press [PF1].

To scroll the screen up one line, press the cursor up key.
To scroll the screen down one line, press the cursor down key.

Exit the "List Parameters" function by pressing [PF3].

To list HITMAN parameters using the command line interface, you would type:

$ HITMAN /LIST(=list_type) [/OUTPUT=file] [/FILE=[prime][nonprime]]

Here is part of a sample HITMAN list:

```
HITMAN version 10.0.0              PRIME Permanent data file
/BIO_THRESHOLD  Less than   1 Bio(s) per interval is idle
/BROADCAST  Broadcast messages are sent to: USER11
/NOCLEAR  The screen is not cleared before the process is deleted
/NOCOMMON  Detached processes are not linked through a common field
/CONSOLE  Messages will be sent to the console for:
          ERROR
          EVENT
          LOG_FILE
          PROCESS
          TIME
/CPU_THRESHOLD  Less than    3 x 10 ms of CPU per interval is idle
/DECWINDOWS  DECWindows restart queue is MY_QUEUE
                        .
                        .
```

Although you can't scroll backwards and forwards using the command line interface, you can use HITMAN/LIST=list_type to select information and make the reports smaller. If you specify /output the output from the list will be directed to an output file instead of the terminal. Add /file=prime or /file=nonprime if you want to see parameters from the specific file.

HITMAN/LIST shows the qualifiers in **ALPHABETICAL** order within the following groups or list types: You can specify a list_type to limit the listing to one type. By default all the lists are shown.

Flags and values	/LIST=FLAGS
Lists	/LIST=LISTS
Messages	/LIST=MESSAGES
Events	/LIST=EVENTS

If a qualifier is negated ("/NOCONSOLE" rather than "/CONSOLE"), it still shows up under the original name rather than including "NO" as part of the sort.

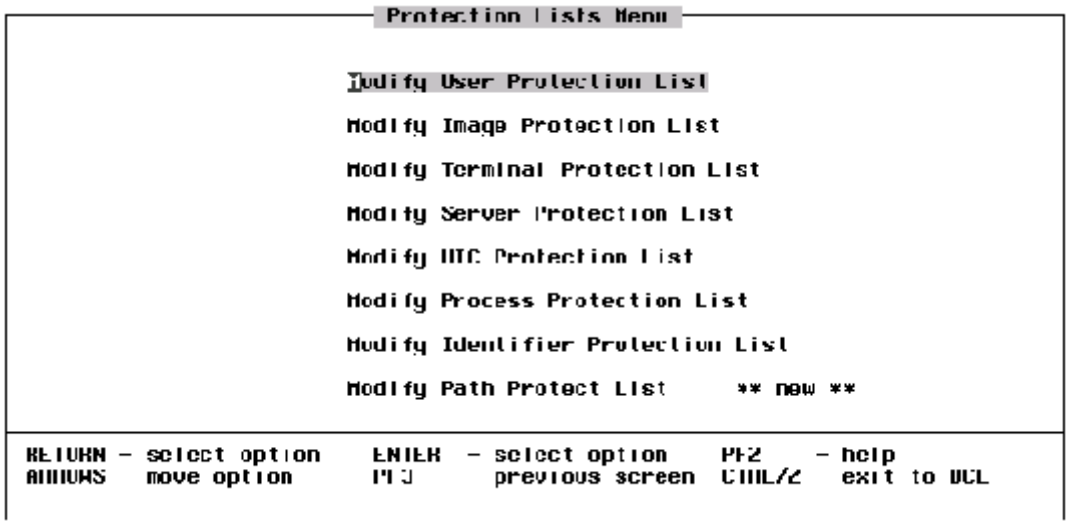
Modifying Parameters

HITMAN stores its parameters in a file called "PERM_DATA_PRIME.DAT" in the directory pointed to by the logical "HITMAN_DAT". (An initial file is included with the distribution kit.) You can also store a second set of parameters in a file called "PERM_DATA_NONPRIME.DAT".

When you modify a parameter, these files are updated and the detached process reads the file to get a copy of the new parameter settings. Thus, any change you make will take effect immediately. If you are running HITMAN on a cluster, all nodes using the parameter file will read the file at the beginning of the next data collection. You do not have to stop and re-start HITMAN when you modify a parameter.

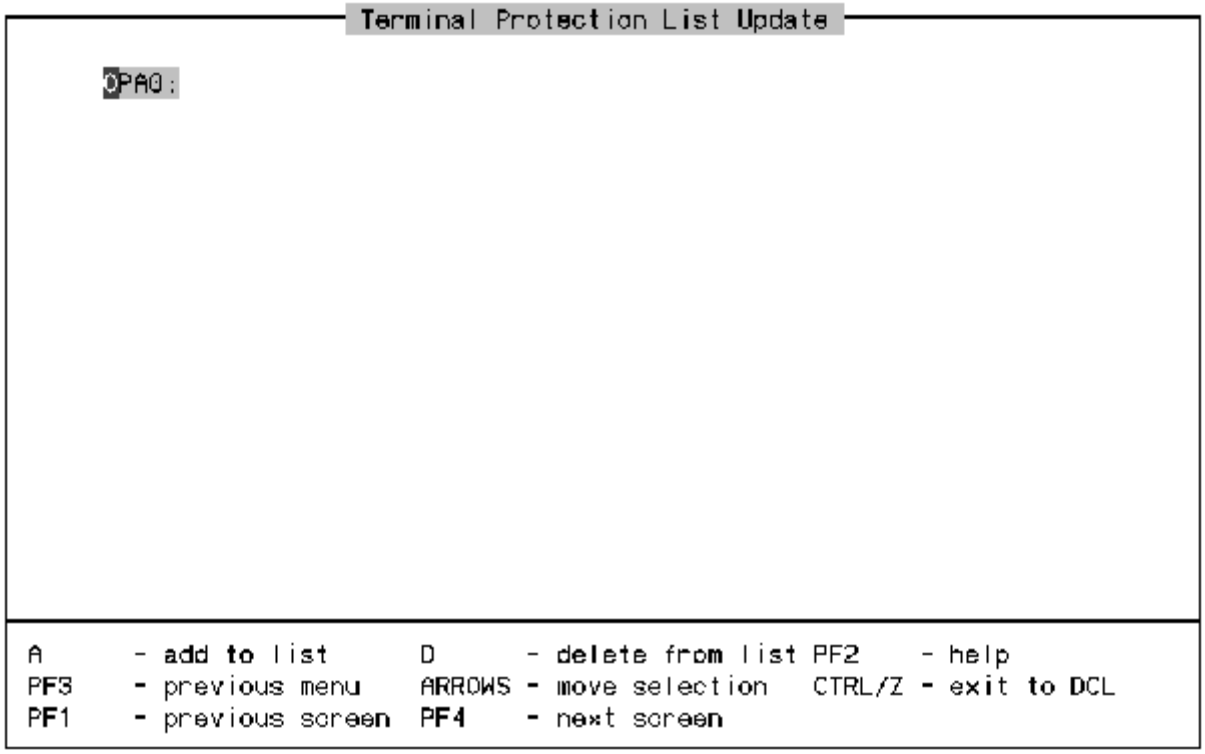
To change HITMAN parameters, select "Modify Parameters" and then select the PRIME file.

There are several menus within the "Modify Parameters" option. To start with, we are going to look at protection. "Protection" is the way you tell HITMAN who is to be exempt from being killed. Select the "Modify Protection Lists" option to display the Protection Lists Menu:



Modifying Terminal Protection Lists

Select the "Modify Terminal Protection List" option. The Terminal Protection List Update screen will appear:



Note that the console terminal "OPA0:" is already protected by default.

To remove a terminal from the protection list, you would use the cursor keys to move until the terminal is highlighted and press "D". HITMAN will request confirmation then remove the terminal from the list.

To add a terminal to the protection list, you would press "A". HITMAN will ask you to input the terminal id and put it on the protected terminal list.

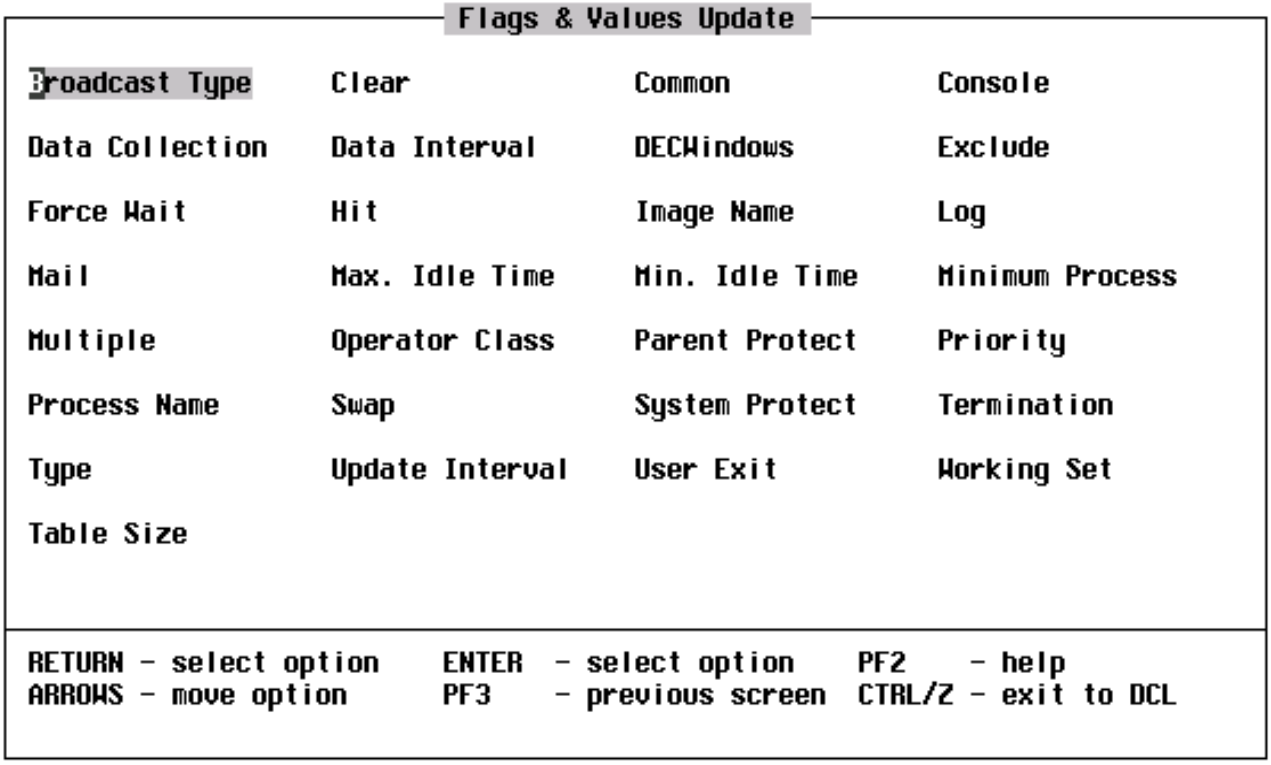
To cancel an add or delete operation, you would press [PF3].

Return to the Protection Lists Menu by pressing [PF3].

Return to the Permanent Data File Update Menu by pressing [PF3].

Modifying Flags & Values

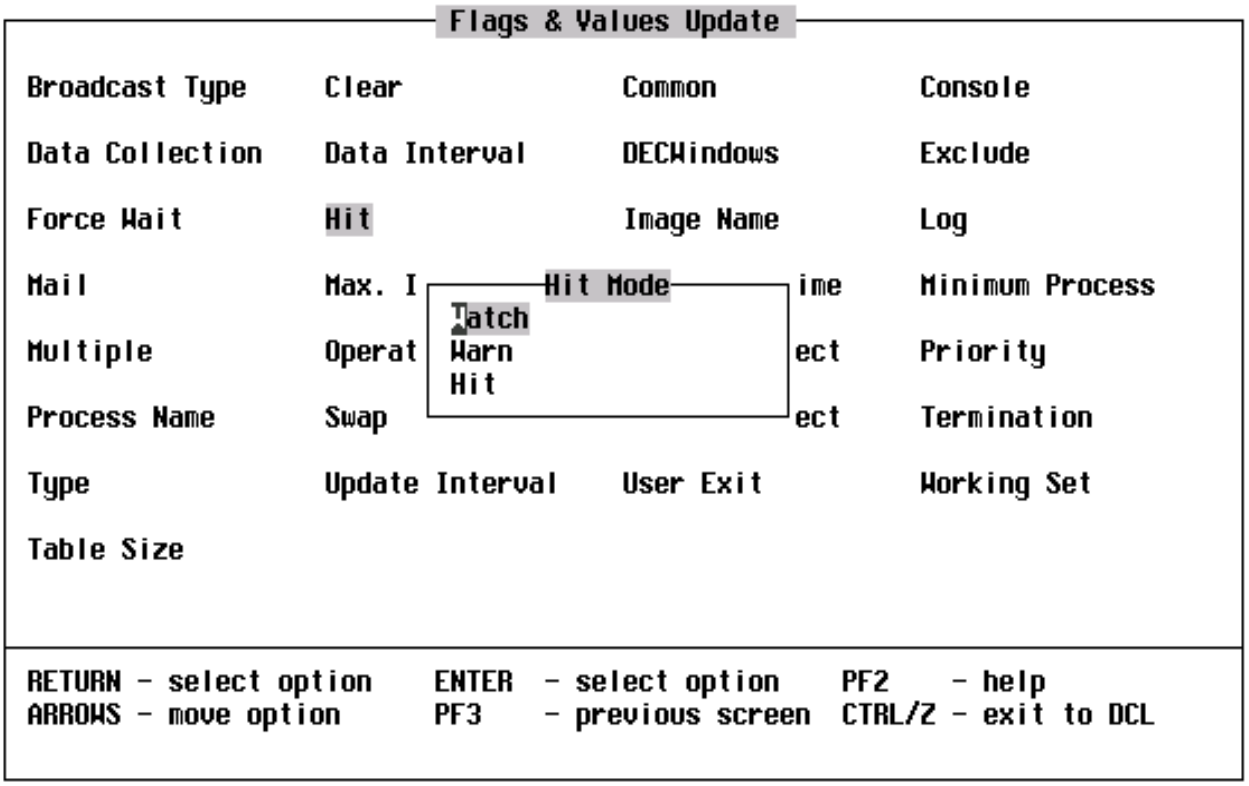
Select the "Modify Flags & Values" option. The Flags & Values Update menu will be displayed:



From this menu, you can view and set many different parameters. For the moment, press the cursor down key three times to highlight the HIT parameter, then press <enter> to select it. A window will pop up showing the three possible HIT modes:

WATCH	records idle process information but does not warn or kill them
WARN	records idle process information and warns idle users but does not kill them
HIT	records idle process information, warns idle users and kills them

The WATCH mode is highlighted to indicate that WATCH is the current setting. To select another setting you would press the cursor down key until the setting you want is highlighted, then press <enter> to change the setting. But, for the moment, leave HITMAN in WATCH mode by pressing [PF3] to leave the option unchanged.



Return to the Permanent Data File Update menu by pressing [PF3].
Return to the HITMAN main menu by pressing [PF3].

Displaying User Information
HITMAN allows you to see information on any process it is monitoring. If a user is not being terminated when you think they should (or vice-versa), you can use this function to find out why.

To display user information, select the "Show User Information" option on the main menu and select the default permanent data file. When you select the "Show User Information", you must enter a username (wildcards are not allowed) and specify a file name into which HITMAN will write the user information. HITMAN will generate a file containing all the information it has on all processes matching the username you specified. By default, the file is called "DUMP.DMP" and is created in the directory pointed to by the logical "HITMAN_DAT". Since the file is an ASCII text file, you can TYPE it, PRINT it or use a text editor to look at it. The contents of the user dump file are explained beginning on page 211.

To show user information using the command line interface, you would type:

```
$ HITMAN /DUMP /USER=username
```

If you want to explore a few additional menus, please do so now. Remember, press [PF3] to cancel an operation or return to the previous menu.

This completes our brief "tour" of HITMAN.

Recreating HITMAN's Default Permanent Data Files
To ensure you have a complete and correct permanent data file after this tutorial, or to recreate the default files if you believe your files have corrupted:

1. Stop HITMAN
2. Delete the existing permanent data files by entering the command:

```
$ DELETE HITMAN_DAT:PERM_DATA_*.DAT;*
```

3. Recreate the file(s) with all defaults by entering the command:

```
$ HITMAN /FILE=[PRIME][NONPRIME][BOTH] /LIST
```

The following chapters describe HITMAN in much greater detail. The next chapter will describe some common HITMAN configurations and how you can set them up for use at your site.

Some Common HITMAN Configurations

The following configurations represent a good portion of the calls we receive from users who are wondering how to set HITMAN up to do something a little "different." For sake of space the examples are given showing DCL level commands with an explanation; for users who prefer the menus, text is given that will enable them to find the correct menus. The configurations are presented in a question and answer format. Configurations are presented for the following:

How to set Hitman to use only one set of parameters 24 hours per day.

How to set Hitman to kill only a limited number of users for testing.

How to set Hitman to kill users only after hours.

How to copy your prime parameters to nonprime; useful if you've made a large number of changes that you wish to also be included in the nonprime file.

How can I share a copy of Hitman across multiple nodes but have different parameter files for each node.

How can I use Hitman to log everyone off the system before my nightly backup.

How can I set Hitman to protect a production copy of an image while terminating idle users in my test/development copy for testing purposes.

Question: How do I set HITMAN so that it will use only one set of parameters? I don't need different prime and nonprime parameters.

Answer: HITMAN provides two data files for flexibility. You define prime days and time to be whatever you want and HITMAN will consider everything outside of prime to be nonprime. You DO NOT define nonprime time. To have HITMAN use only one file, always the prime file, you can simply disable HITMAN's check for prime/nonprime time function.

Step	Description	Menu
1	Set HITMAN to use only the prime file 24 hours a day, 7 days a week by disabling the prime/nonprime check $ HITMAN/DISABLE_PRIME	***modify*** ***prime*** ***prime time*** ***disable***
2	Start HITMAN if it isn't already running. You do not need to stop HITMAN to change parameters. $ HITMAN/START	***Start***
3	Obtain a HITMAN status; it will indicate whether you are in Watch, Warn or Hit mode and that only the prime data file will be used 24 hours a day $ HITMAN/STATUS	***Status***

Question: How do I set up HITMAN, for testing purposes, so that it only kills a couple of users?

Answer: HITMAN has a built in feature that allows you to *reverse the function of one protection list making it a "hit" list.* Once it is set up everyone on the protection list will terminated and everyone else will be protected by default. You can take advantage of the username, image, terminal, UIC, server, process OR identifier protection lists. Only one list can be excluded.

Suppose you only want users XYZ* to be killed if they are idle. You can specify specific or wildcarded values for most protection lists.

Step	Description	Menu
1	Add XYZ* to the userprotection list $ HITMAN/PROTECT/USER=XYZ*	***modify prime protection username***
2	Remove any other users that are currently on the list $ HITMAN/PROTECT/NOUSER=([user1],[user2])	
3	Tell HITMAN to "exclude" the user protection list $ HITMAN/EXCLUDE=USER	***modify prime flags & values exclude***
4	Make sure that HITMAN is in HIT mode $ HITMAN/HIT=HIT	***modify prime flags & values hit***
5	Make the same changes to the nonprime file OR set HITMAN to use only the prime file 24 hours a day, 7 days a week	
6	Start HITMAN if it is not already running. $ HITMAN/START	***Start***
7	Obtain a HITMAN status; it will indicate that you are in HIT mode and that only x users on the username protection list will be hit $ HITMAN/STATUS	***Status***

NOTE: Exclude is frequently used for testing purposes. When you decide to turn off exclude and switch back to using the excluded list as a normal protection list be sure to delete the users in the excluded protection list and add back your original entries, if any.

Question: How can I set up HITMAN so that I only kill users after hours?

Answer: There are two distinct ways to do this depending on whether or not you want HITMAN to monitor events during the day.

HITMAN should monitor events during prime time (Monday-Saturday, 8am to 7 pm) and kill users between 7 pm and 8 am as well as all day Sunday.

Step	Description	Menu
1	Make sure prime time is defined correctly and that HITMAN is checking for prime/nonprime time $ HITMAN /PRIME_START=08:00 /PRIME_END=19:00 - /DAY=(WEEKDAYS,SATURDAY) $ HITMAN /ENABLE_PRIME	***modify prime prime time***
2	Protect everyone during prime time $ HITMAN /PROTECT /USER=* /FILE=PRIME	***modify prime protection username***
3	Optionally increase the data collection interval to lessen HITMAN's impact on the system during prime time. This will affect how often HITMAN checks your events; if it is critical your events be detected quickly we don't suggest you make this change. $ HITMAN /INTERVAL=5 /FILE=PRIME	***modify prime flags & values data interval***
4	Make sure HITMAN is in HIT mode in nonprime $ HITMAN /HIT=HIT /FILE=NONPRIME	***modify nonprime flags & values hit***
6	Start HITMAN if it is not already running. $ HITMAN /START	***Start***
7	Obtain a HITMAN status to confirm that HITMAN is running and what username it is running under. $ HITMAN /STATUS	***Status***

NOTE: HITMAN doesn't collect activity information for protected processes; when it switches from prime to nonprime it will be wait=x minutes before HITMAN terminates idle users. To terminate users sooner put HITMAN into watch mode during prime time instead; in WATCH mode HITMAN collects activity information and records the actions it would take to the log file but doesn't actually kill anyone. When HITMAN switches to nonprime, users that are eligible to be hit and have been idle for more than x minutes will receive one warning message then be killed on the next data collection.

If you don't need HITMAN to monitor for events during the day do the following:

Step	Description	Menu
1	Make sure prime time is defined correctly and that HITMAN is checking for prime/nonprime time $ HITMAN /PRIME_START=08:00 /PRIME_END=19:00 - /DAY=(WEEKDAYS,SATURDAY) $ HITMAN /ENABLE_PRIME	***modify*** ***prime*** ***prime time***
2	Put HITMAN into "watch" mode during prime time $ HITMAN /HIT=WATCH /FILE=PRIME this step is not strictly necessary but is good insurance in case data collection during prime time ever gets turned back on	***modify*** ***prime*** ***flags & values*** ***hit***
3	Tell HITMAN to not collect process data during prime time $ HITMAN /OFF /FILE=PRIME	***modify*** ***prime*** ***flags & values*** ***data collection***
4	Make sure HITMAN is in HIT mode in nonprime $ HITMAN /HIT=HIT /FILE=NONPRIME	***modify*** ***nonprime*** ***flags & values*** ***hit***
5	Start HITMAN $ HITMAN /START	***Start***
6	Obtain a HITMAN status; it will indicate that you are in HIT mode if it's currently nonprime time or WATCH mode if it's prime time. $ HITMAN /STATUS	***Status***

Question: I've just made a bunch of changes to HITMAN's prime file. Is there any way I can copy all the changes I've made to the nonprime file without having to enter them again?

Answer: The answer to this questions is - That depends. If your prime and nonprime parameters are quite close (for example they should have all the same events) then you can. If the parameters are going to be significantly different it may be easier and faster to configure the nonprime file manually.

After you have configured the prime file and included everything in it that you want to be common to prime and nonprime:

Step	Description	Menu
1	Make sure prime time is defined correctly and that HITMAN is checking for prime/nonprime time $ HITMAN /PRIME_START=08:00 /PRIME_END=19:00 - /DAY=(WEEKDAYS,SATURDAY) $ HITMAN /ENABLE_PRIME	***modify prime prime time***
2	Set your default directory to HITMAN's dat subdirectory and copy the prime parameters to be the nonprime parameters as well $ SET DEFAULT HITMAN_DAT: $ COPY PERM_DATA_PRIME.DAT PERM_DATA_NONPRIME.DAT	
3	Using the menu, or by adding /FILE=NONPRIME to DCL level commands, make any customizations necessary to the nonprime data	***modify nonprime***
4	Using the menu, or by adding /FILE=PRIME to DCL level commands, make any customizations necessary to the prime data	***modify prime***
5	Start HITMAN $ HITMAN /START	***Start***
6	If you want to compare the two sets of parameters to see the differences you can list the two sets of parameters then use DCL differences command to compare them. $ HITMAN /LIST /FILE=PRIME /OUTPUT=PRIME.LIS $ HITMAN /LIST /FILE=NONPRIME /OUTPUT=NONPRIME.LIS $ DIFFERENCES PRIME.LIS NONPRIME.LIS	

NOTE: When configuring HITMAN from the DCL level it is possible to update both the prime and nonprime files for any field that doesn't include a special character like ",".Add /FILE=BOTH to your DCL command and HITMAN will change both the prime and nonprime files.

Question: How can I set up HITMAN so that it has different parameters for different nodes but still share one installed copy?

Answer: It is quite easy to configure HITMAN this way by taking advantage of the PERM_DATA_PRIME and PERM_DATA_NONPRIME logical names.

Step	Description	Menu
1	Determine if there is going to be a number of parameters common to the nodes, such as protected users or images, or common events. If there is you should configure HITMAN normally on one node to begin.	
2	Determine a naming convention for the data files. We recommend calling them PERM_DATA_PRIME.node and PERM_DATA_NONPRIME.node.	
3	On each node define the logical names that direct HITMAN to use data files other than the default files. Be sure to add these definitions to SYS$MANAGER:HITMAN_SITE_LOGICALS.COM; this file is not deleted or replaced by subsequent installations/upgrades of HITMAN and preserves site specific logicals. $ DEFINE /SYS PERM_DATA_PRIME - HITMAN_DAT:PERM_DATA_PRIME.node repeat for nonprime.	
4a	If you are starting from a created data file simply copy it to the appropriate file(s). $ COPY PERM_DATA_PRIME.DAT PERM_DATA_PRIME.node $ COPY PERM_DATA_NONPRIME.DAT PERM_DATA_NONPRIME.node	
4b	To start from new default HITMAN data files enter the following command and answer YES when it asks if you want to create new data files. $ HITMAN /LIST /FILE=BOTH	***List Prime List Nonprime***
5	On each node in the cluster you may now customize the parameter files for that node using the DCL or menu interfaces	
6	On each node in the cluster you may now start HITMAN. $ HITMAN /START	***Start***

NOTE: NODE specific files such as HITMAN's log, error and output files always include a node name in the file name or type so multiple nodes can share the .dat directory. If you want to keep as much as possible in its default state you can create separate .dat directories for each node then redefine the HITMAN_DAT logical on each node to point to the new directory. Be sure to update HITMAN_SYSTEM_LOGICALS.COM on each node. You can copy the contents of the original HITMAN_DAT directory to each new directory to get started.

Question: Can I use HITMAN to get everyone off the system before my nightly backups?

Answer: HITMAN includes a little utility that we call "Mad Dog." This utility ignores activity, treats everyone as if they were idle, and warns them at 1, 2, 3 and 4 minutes then terminates them at five minutes. Mad Dog works by stopping and restarting HITMANafter directing it to a special permanent data file. After a few minutes it points HITMAN back to your regular data and restarts it.

Step	Description
1	Create the Mad Dog permanent data file. $ @HITMAN_COM:CREATE_MAD_DOG_FILE
2	Make any customizations to the Mad Dog parameters that you want. You can take advantage of all the same user protection features available in HITMAN. $ @HITMAN_COM:MODIFY_MAD_DOG_FILE will take you into the menu to Mad Dog's data file, select PRIME for the data file.
3	You may run Mad Dog at anytime by submitting it to a batch queue under a privileged account: $ Submit /Queue=node$batch Hitman_com:mad_dog /Noprint /Keep for safety purposes the Mad Dog procedure always adds the user that submitted it to the protection lists and it cannot be run interactively.
4a	Optionally you may set up a HITMAN time event to automatically submit Mad Dog before your backups. For example if you are running HITMAN with prime and nonprime files and want to log everyone off before your backup starts at 11 pm: $ HITMAN /EVENT=(TYPE=TIME, TIME=22:50, DAY=WEEKDAYS, - SUBMIT=(QUEUE=node$batch, JOB=HITMAN_COM:MAD_DOG.COM, - USER=SYSTEM, KEEP, NOPRINT)) /FILE=NONPRIME
4b	Alternatively, you can add the command $ @HITMAN_COM:MAD_DOG to your backup batch procedure at the top of the procedure so that as soon as HITMAN is finished terminating everyone your backup begins to run.

CAUTION: If you delete the Mad Dog job while it is running HITMAN will be stuck in Mad Dog mode. If this happens, stop HITMAN, deassign the logical names perm_data_prime and perm_data_nonprime and restart HITMAN.

A common change to Mad Dog is to edit the Mad Dog procedure and increase the Maddog_time symbol to a larger number of minutes. This keeps HITMAN in Mad Dog mode while your backup is running; anyone who logs on during the backup will be warned terminated within 5 minutes.

Using the menu-driven screen interface.

Type "HITMAN/MENU" to display the main menu.

Press the cursor up and down keys to move up/down within a menu. The current menu function is always highlighted in reverse video. Menus do not "wrap". Pressing cursor down at the bottom of menu or cursor up at the top leaves the menu unchanged.

Press <enter> to select a menu function.

Press [PF2] to get HELP. If you are in a menu, you will receive HELP on the menu and the currently highlighted option. If you are in a data entry screen, you will receive HELP on that screen.

Press [PF3] to cancel an operation or return to the previous menu.

Press <CTRL/Z> to exit the HITMAN MENU interface and return to the DCL $ prompt. HITMAN updates the permanent data file and exits.

The following page contains a list of all the menus which can be called from the main menu.

Important Note: Hitman Version 10 has changed most tables, such as protection tables and idle and warn time tables, to be dynamic. These tables may have any number of entries. Unfortanately the OpenVMS Screen Management libraries (SMG) do not easily support dynamic memory allocation. As a result all menu screens in which entries are being added/deleted from a list are limited to 2,048 entries; if you require more entries than this in a single table please use the DCL interface commands to add the extra entries.

Menu tree

Here is a "map" showing how HITMAN menus link together. You can use it to find where a particular function is located.

- MAIN
  - Modify Parameters
    - Modify Protection Lists
      - Modify User Protection List
      - Modify Image Protection List
      - Modify Terminal Protection List
      - Modify Server Protection List
      - Modify UIC Protection List
      - Modify Process Protection List
      - Modify Identifier Protection List
      - Modify Path Protection List ***New***
    - Modify Idle and Warn Times
      - Modify System Idle Time
      - Modify Dialup Idle Time
      - Modify UIC Idle List
      - Modify User Idle List
      - Modify Image Idle List
      - Modify Set Idle List
      - Modify Terminal Idle List
      - Modify Server Idle List
      - Modify Identifier Idle List
      - Modify Process Idle List ***New***
      - Modify Path Idle List ***New***
    - Modify Autohit List
    - Modify Authorize List
    - Modify Disconnect List
      - Modify User Disconnect List
      - Modify Image Disconnect List
    - Modify Connect Lists
      - Modify User Connect List
      - Modify UIC Connect List
      - Modify Dialup Connect List
      - Modify Image Connect List
      - Modify Terminal Connect List
      - Modify Server Connect List
      - Modify Identifier Connect List
      - Modify Process Connect List ***New***
      - Modify Path Connect List ***New***
    - Modify Exit Code List
    - Modify Threshold List ***New***
      - Modify System Thresholds ***New***
      - Modify Image Thresholds ***New***
    - Modify Flags & Values ***Updated***
      - Broadcast Type
      - Clear
      - Common
      - Console
      - Data Collection
      - Data Interval
      - DecWindows
      - Exclude
      - Force Wait
      - Hit
      - Image name
      - Log
      - Mail
      - Max. Idle Time
      - Min. Idle Time
      - Minimum Process
      - Multiple
      - Operator Class
      - Parent Protect
      - Priority
      - Process Name
      - Swap
      - System Protect
      - Termination
      - Type
      - Update Interval
      - User Exit
      - Working Set
      - Table size ***New***
    - Modify Message Text
      - Modify First Warning Text
      - Modify Second Warning Text
      - Modify Final Message Text
      - Modify Authorize Message Text
      - Modify Autohit Message Text

Modify Event List
Add Event to List
Absent Process
Present Process
Runaway Process
Privilege Check
Priority Change
Process State
User
Image
Terminal
Image Count
Absent Batch
Present Batch
Monitor Queue
Free Blocks
Device Error
Node
Repeating
Time
Startup
Delete Event from List
Modify Event in List
Resequence Event List
Modify Prime Time
Modify Holiday List
List Parameters
Validate Parameters
Analyze Log
Show HITMAN Status
Show User Information
Start HITMAN
Stop HITMAN
Disable broadcast messages in menu ***New***
Summarize Log ***New***

Using the command line interface

Here are some examples for changing various parameters using the command line:

To disable messages to the operator's console, type:

```
$ HITMAN/NOCONSOLE
```

HITMAN messages will not be sent to the operator's console.

To protect user FRED, type:

```
$ HITMAN/PROTECT/USER=FRED
```

FRED will not be killed, no matter how long he is idle, or what images he runs.

To authorize user MOE to have privileges.

```
$ HITMAN/AUTHORIZE/USER=MOE
```

User MOE is authorized to have privileges in the ALL category.

To protect the image EDT, type:

```
$ HITMAN/PROTECT/IMAGE=EDT
```

Any process running EDT will not be warned or killed.

To hit any user who runs the image OTHELLO, type:

```
$ HITMAN/AUTOHIT/IMAGE=OTHELLO
```

NOTE: When you change a parameter, HITMAN tells the detached process to read the parameter file. If the detached process does not exist at this point, you will get the "HITMAN is not running" message:

```
%HITMAN-I-NOTRUN, HITMAN is not running
```

This message is informational. It does not represent an error and can be ignored.

Getting users to accept HITMAN

This chapter shows you how to overcome objections and help your users to accept HITMAN.

Initially, your users will probably resent being terminated when they are idle. However, after a few weeks, users often come to rely on being killed by HITMAN when they forget to log off. If you turn HITMAN off for some reason, the number of interactive processes on your system will explode because users assume they will be killed but aren't.

You can improve user acceptance of HITMAN by moving slowly, stressing the benefits of having HITMAN on the system and keeping users informed. No one likes to find out there's an idle process killer on the system by having their process deleted. Keep your users informed and help them understand that everyone's response will be better when resources are not being wasted on idle processes.

To ensure a smooth introduction of HITMAN on your system:

1. Start HITMAN in WATCH mode to see who it would kill. Then adjust your wait, warn and protection parameters to give users lots of time. Although some users may have valid reasons for being idle for 10 minutes, very few have valid reasons for being idle for two hours!

2. Put HITMAN in WARN mode and suppress the final warning message. If you use the default warn and termination HITMAN messages, users will assume they have been killed and get confused. Instead, use warning messages which encourage users to resume activity or log off. Here's a sample warning message:

    ?U?, your terminal has been ?M? for ?I? minutes. Please resume activity or log off.

Assuming FRED has been idle 45 minutes and /FIRST_WARN=45, this message would appear:

    FRED, Your terminal has been idle for 45 minutes. Please resume activity or log off.

3. Put HITMAN in HIT mode and test it on a single user to ensure everything works as you expect. To test HITMAN on a single user, type the following command:

    $ HITMAN/PROTECT/USER=username/EXCLUDE=USER

where "username" is the test user. This command protects all other users EXCEPT the one specified. Thus, only the test user will be warned or killed. Everyone else will be not be warned or killed.

4. When you are sure everything is operating to your satisfaction, remove the single user restriction using the commands:

$ HITMAN/PROTECT/NOUSER=username
$ HITMAN/NOEXCLUDE

5. Over a few weeks, slowly reduce wait and warn times until you reach the times you want.

Remember, move slowly, stress the benefits and keep your users informed. Faster response and access to limited licenses is to everyone's advantage.

Setting maximum idle times

This chapter shows you how to set the maximum amount of time a process can be idle on your system before HITMAN takes action.

HITMAN checks your system at regular intervals (the default is 1 minute). Each time it checks your system, HITMAN determines if a process has been idle and increments the process's idle time by the number of minutes in the data collection interval. Once a process has accumulated the maximum idle time you have specified (the "wait" time), HITMAN warns and/or kills it.

Your minimum wait or warn time must be at least twice the data collection interval. Thus, with a data collection interval of 1 minute, your minimum wait or warn time should be 2 minutes or more.

In any case, wait and warn times must be multiples of the data collection interval. Thus, if you have a data collection interval of 5 minutes, your first warn time should be 10 minutes.

You can set a system-wide maximum idle time which applies to every interactive process on your system. And, you can set higher or lower times for specific users, images, terminals, servers, UIC's, identifiers, processes, paths (users running a specific instance of an image) and DIALUP processes (as opposed to directly connected processes).

Setting a system-wide idle time

Use the /WAIT=time qualifier by itself to indicate how many minutes of inactivity constitutes an "idle" process. The default is /WAIT=30 which means that after 30 minutes of inactivity, HITMAN considers the process idle. If you specify /WAIT by itself, it applies to every interactive process on your system. For example, the command:

$ HITMAN/WAIT=45

specifies that the maximum idle time for all interactive processes is 45 minutes.

Setting idle times for users

Use the /USER=(...)/WAIT=n qualifiers to set the maximum idle time for one or more users. Wildcards are allowed. You can define as many user idle times as you require. For example, the command:

```
$ HITMAN/USER=LARRY/WAIT=35
```

specifies a maximum idle time of 35 minutes for user LARRY. This idle time over-rides the system-wide maximum idle time.

Setting idle times for images

Use the /IMAGE=(...)/WAIT=n qualifiers to set the maximum idle time for one or more images. Wildcards are allowed. You can define as many image idle entries as required.

Setting idle times for processes by name

Use the /PROCESS=(...)/WAIT=n qualifiers to set the maximum idle time for one or more processes by their process name. Wildcards are allowed. You can define as many process idle times as you require.

Setting idle times for UIC's

Use the /UIC=(...)/WAIT=n qualifiers to set the maximum idle time for one or more UIC's. Enter UIC's in the normal OpenVMS format and enclose them in double quotes ("[100,651]"). Wildcards are allowed. You can define as many UIC idle times as you require.

Setting idle times for terminals

Use the /TERMINAL=(...)/WAIT=n qualifiers to set the maximum idle time for one or more terminals. Wildcards are allowed. You can define as many terminal idle times as you require.

Setting idle times for servers

Use the /SERVER=(...)/WAIT=n qualifiers to set the maximum idle time for one or more server id's. Wildcards are allowed. You can define as many server idle times as you require.

Setting idle times for paths

Unlike the image idle times when specifying a path idle time you include the disk, directory and filename. Use the /PATH=path/WAIT=n qualifiers to set the maximum idle time for a path. Wildcards are not allowed. You can define as many path idle times as you require.

Setting idle times for identifiers

Use the /IDENTIFIER=(...)/WAIT=n qualifiers to set the maximum idle time for one or more identifiers. Wildcards are not allowed. You can define as many identifier idle times as you require. Any process having the identifier will receive the corresponding identifier.

Setting idle times for DIALUP processes

Use the /DIALUP/WAIT=n qualifiers to set the maximum idle time for dialup (as opposed to local) processes.

Allowing users to set their own idle times

Use the /USER=(...)/ALLOW command to allow selected users to control their environment by setting their own maximum idle time using the command "SET_IDLE". HITMAN uses the idle time they specify for their process and any subprocesses they create. For example, if user FRED enters the command:

```
$ SET_IDLE 25
```

HITMAN will set his maximum idle time to 25 minutes assuming you have entered the command:

```
$ HITMAN/USER=FRED/ALLOW
```

Users cannot set idle times for any process other than their own. To see their current idle time setting, users can enter the following command:

```
$ SET_IDLE ?
```

HITMAN will display their current idle time setting.

NOTE: You must define "SET_IDLE" as a global symbol with the command:

```
$ SET_IDLE :== "$HITMAN_EXE:SET_IDLE"
```

SET_IDLE.EXE must have a protection of W:E and be installed with the privilege CMKRNL.

```
$ INSTALL :== $INSTALL/COMMAND_MODE
$ INSTALL
INSTALL> ADD HITMAN_EXE:SET_IDLE /PRIVI=CMKRNL
INSTALL> EXIT
```

Users must type "SET_IDLE 25" rather than "SET_IDLE=25" since DCL assumes you want to redefine the symbol SET_IDLE when it receives the "=" character.

You can control which users are allowed to use the "SET_IDLE" command as well as the minimum and maximum times they set. For example, the following commands:

```
HITMAN/USER=(LARRY,M*)/ALLOW
HITMAN/MIN_IDLE=10/MAX_IDLE=60
```

allow LARRY and any username starting with "M" to set their own idle times. However, the

second command limits the minimum idle time they can set to 10 minutes and the maximum to 60 minutes. If a user specifies a time outside the range or if users not on the /ALLOW list enter the SET_IDLE command, it is rejected with an error message.

The lowest value HITMAN allows for /MIN_IDLE is twice the data collection interval.

NOTE: The user-specified SET_IDLE command uses the per-process, site-specific longword JPI$_SITESPEC returned by the system service $GETJPI to store the user-specified idle time. If you have any other software on your system using this longword, there may be conflict errors. Normally the SET_IDLE program will detect this automatically since it can detect "foreign" values in the longword as opposed to the values it uses. If a third party product acts strangely when you start use SET_IDLE, check with the vendor to see if they use JPI$_SITESPEC.

How HITMAN applies different idle times

Since several special idle times can apply to any given process, HITMAN uses the following rules (the most important rule is first) in selecting the applicable idle time:

1. If the user specifies a time using SET_IDLE, use that time.

2. If a /USER time matches the process username, use the /USER time.

3. If a /UIC time matches the process UIC, use the /UIC time.

4. If an /IDENTIFIER time matches a process identifier, use the /IDENTIFIER time.

5. If a /IMAGE time matches the process image name, use the /IMAGE time.

6. If a /TERMINAL time matches the process terminal, use the /TERMINAL time.

7. If a /SERVER time matches the process server ID, use the /SERVER time.

8. If there is a /DIALUP time, and the process type is DIALUP, use the /DIALUP time.

9. Finally, if no other idle times are applicable, use the /WAIT time.

Lowering idle times when HITMAN is running

When you change a wait time to a value that is LOWER than the current wait time, there may be processes which have already accumulated that amount of idle time. HITMAN warns them at the next data collection interval and kills them at the following data collection interval unless they become active. If you have a shorter wait time in the NONPRIME file than in the PRIME, or the opposite, you may see a number of processes get terminated shortly after HITMAN switches data files because this change suddenly makes them idle for longer than the wait time.

How HITMAN decides a process is idle

A process is idle during a data collection interval if it:

- Uses less than the threshold amount of CPU
  AND
- Uses less than the threshold amount of buffered I/O's
  AND
- Uses less than the threshold amount of direct I/O's

OR

- Uses less than the threshold amount of CPU specified for the image it is running
  AND
- Uses less than the threshold amount of buffered I/O's specified for the image it is running AND
- Uses less than the threshold amount of direct I/O's specified for the image it is running

All the above conditions must be true for the process to be considered idle. If any are false, the process will be considered active and HITMAN will reset the number of minutes idle to zero.

Dealing with processes which use very little resources

Some processes use small amounts of CPU time and Buffered I/O's even when they are truly idle. You can set thresholds so that HITMAN will still consider these processes idle.

By default, if a process uses less than 30 milliseconds of CPU time per data collection interval, HITMAN considers it idle.

Use the /CPU_THRESHOLD, /BIO_THRESHOLD and /DIO_THRESHOLD qualifiers to specify thresholds appropriate to your site. With HITMAN Version 8 you can set special thresholds for individual programs (images); this should be useful with programs that use a lot less, or a lot more, resources when idle than the system defaults.

Dealing with processes which use a lot of resources

Some processes use substantial amounts of CPU time and Buffered I/O's even when they are truly idle. A common example is a menu screen which displays the time and updates it continuously. You can set thresholds for individual images to help compensate for background resource usage; HITMAN can determine that these processes are idle without affecting other users.

HITMAN will check processes to see if the image they are running has special thresholds. If it does HITMAN will use those threshold values; otherwise it will use the system default thresholds.

Use the /CPU_THRESHOLD, /BIO_THRESHOLD and /DIO_THRESHOLD with the /IMAGE qualifier on the same line to specify thresholds appropriate for a specific program.

Caution: Modifying threshold values can be risky. If thresholds are set too high active users may suddenly appear inactive and be terminated even though they are working. When making a large to a threshold, or thresholds, you may wish to run Hitman in Watch mode awhile and review Hitman's log file to make sure no users would have been terminated that shouldn't have been. When you are satisfied that the threshold values are resulting in termations for the offending processes without terminations of active users you can reset Hitman to Hit mode.

Warning idle users

This chapter shows you how to warn users to give them a chance to become active before HITMAN kills their process.

You can have HITMAN send one or more warning messages to idle processes before their processes are killed. This allows users to respond if they are thinking rather than truly inactive.

You can also turn off all warning messages so idle processes will be hit without warning.

You can set system-wide warning times which apply to every interactive process on your system. And, like maximum idle times, you can set higher or lower warning times for specific users, images, UIC's, terminals, servers, identifiers and DIALUP processes (as opposed to directly connected processes). Please note that warning times must be less than wait time.

Use the /FIRST_WARN=n and /SECOND_WARN=n qualifiers to indicate when the first and second messages are sent. With the default settings of /FIRST_WARN=20 and /SECOND_WARN=25, HITMAN sends idle users the first warning message after 20 minutes of inactivity. After 25 minutes of idle time, HITMAN sends idle users the second warning message.

Use the /MULTIPLE qualifier to send the second warning message every data collection interval after the second warning until the process becomes active or is terminated.

If you specify /FIRST_WARN and /SECOND_WARN by themselves, they apply to every process on your system that is eligible to receive these warnings. For example, the command:

```
$ HITMAN/FIRST_WARN=10/SECOND_WARN=15
```

specifies that all idle eligible processes will receive a first warning after 10 minutes of idle time and a second warning after 15 minutes of idle time.

You can also specify first and second warning times for:

- users	/USER
- images	/IMAGE
- UIC's	/UIC
- identifiers	/IDENTIFIER
- terminals	/TERMINAL
- servers	/SERVER
- DIALUP processes	/DIALUP
- processes by name	/PROCESS
- image paths	/PATH

For example, the command:

$ HITMAN/IMAGE=RX109/SECOND_WARN=55/WAIT=60

specifies a second warning time of 55 minutes for image RX109. This second warning time overrides the system-wide second warning time.

If users have set their own idle times using the "SET_IDLE" command, HITMAN sends the first warning message two (2) data collection intervals and the second warning message one (1) data collection interval before they are terminated. Users always receive at least one warning message before they are terminated regardless of their idle time setting.

Since several special warning times can apply to any given process, HITMAN uses the same order of precedence used for maximum idle times.

Suppressing warning messages

You can suppress the first warning message using the qualifier /NOFIRST_MESSAGE. You can suppress the second warning message using the /NOSECOND_MESSAGE. If you suppress both warning messages, idle users will not receive any warning before being killed.

Handling terminals set to /NOBROADCAST

If users have their terminals set /NOBROADCAST or have typed <CTRL/S>, the warning message cannot be displayed and a record is written to the log file indicating what happened. However, users will still be logged out if they exceed their idle time.

Specifying message text

Specify the text of the first and second warning messages as well as the final warning message using the qualifiers:

/FIRST_MESSAGE="message text"
/SECOND_MESSAGE="message text"

/FINAL_MESSAGE="message text"

If you do not specify message text, HITMAN uses the default message text for the first and second warnings:

*** HITMAN Warning, You have been ?M? for ?I? minutes ** ?L?

and the default text "** HITMAN Termination, You have been ?M? for ?I? minutes ** ?L?" for the final message sent when the process is terminated.

The maximum length of the messages is a 200 character string after variables have been substituted.

Inserting variable information into messages

Use message variables to make your messages more site specific. These variables have three characters and begin and end with a questions mark (?). The following is the list of variables you can use and the amount of space they require:

Variable	Gets Replaced by
?F?	first warning time (3 characters)
?H?	product name "HITMAN" (15 characters)
?I?	number of idle minutes accumulated (3 characters)
?L?	insert a carriage return/line feed
?M?	mode of termination (Connected or Idle, 10 characters)
?R?	number of idle minutes remaining (3 characters)
?S?	second warning time (3 characters)
?T?	current time and date (23 characters)
?U?	username receiving the message (up to 12 characters)
?W?	wait time (3 characters)

Here is an example of how these variables are used:

$ HITMAN -
/FIRST_MESSAGE="Hey ?U?, you have been ?M? for ?I? minutes?L?"

After 20 minutes of idleness, user FRED receives the following message:

Hey FRED, you have been idle for 20 minutes

You can specify any number of variables in a message.

Sample wait and warning messages

Here are some sample wait and warning messages:

/FIRST_MESSAGE
"Your process has been ?M? for ?I? minutes. Please resume activity?L?
or log off so others can use the system."

/SECOND_MESSAGE
"Your process has been ?M? for ?I? minutes. Please resume activity?L?
or log off to prevent security problems and allow others to use the system.?L?
After ?R? minutes your process will be terminated."

/FINAL_MESSAGE
"Your process was ?M? for ?I? minutes. To prevent security problems and?L?
allow others to use the system, your process has been terminated."

Controlling message actions

Specify qualifiers with each message to change the action of the message. If you specify /BELL, HITMAN will ring the terminal bell when it sends the message to the user terminal. If you specify /ALARM, HITMAN will send a warning message to the system console when it sends the message to the user terminal.

Protecting processes

This chapter shows you how to protect processes so HITMAN will not terminate them when they are idle.

Processes automatically protected by HITMAN

HITMAN automatically protects the following types of processes:

- parent processes with an active sub-process
- real-time processes (priority > 16)
- processes in a resource wait (MWAIT) state
- processes which HITMAN has disconnected

If a process has spawned a subprocess which is active, the idle parent process is NOT killed. Conversely, an idle subprocess is not killed so long as the parent process is active. The whole process tree must be idle before the parent process or any of its subprocesses are killed. In addition, if any of the process tree is running a protected image, on a protected terminal, etc., HITMAN protects all the processes.

In addition to HITMAN's automatic protection, you can tell HITMAN to leave processes untouched regardless of activity using HITMAN's protection lists. This function allows selected users to handle their own affairs, free from interference.

Protecting users

Use the /PROTECT/USER=(username,...) qualifiers to indicate which usernames are not to be touched. Wildcards are allowed. Hitman can protect any number of users.

Protecting images

Use the /PROTECT/IMAGE=(image,...) qualifiers to indicate which images are not to be touched. Wildcards are allowed. Hitman can protect any number of images. The DECNET images RTPAD and LTPAD are automatically protected. When adding images to this list specify only the filename portion of the executable. Use the DCL SHOW PROCESS/CONTINUOUS/ID= command to get a display of the image a user is running.

Protecting UIC's

Use the /PROTECT/UIC=("[n,n]",...) qualifiers to indicate which UIC's are not to be touched. Wildcards are allowed. Hitman can protect any number of UIC's.

Protecting identifiers

Use the /PROTECT/IDENTIFIER=(i,...,i) qualifiers to indicate processes having these identifiers are not to be touched. Wildcards are not allowed. Hitman can protect any number of identifiers.

Protecting terminals

Use the /PROTECT/TERMINAL=(terminal,...) qualifiers to indicate which terminals are not to be touched. Wildcards are allowed. Hitman can protect any number of UIC's. The console terminal OPA0: is automatically protected.

Protecting servers

Use the /PROTECT/SERVER=("server ID",...) qualifiers to indicate which server IDs are not to be touched. The quotes around server ID are required if the server ID contains special characters. Wildcards are allowed. Hitman can protect any number of server IDs. The server ID includes the server name and port name as displayed by a show user/full command.

Protecting paths

Use the /PROTECT/PATH=path qualifiers to indicate which image paths are not to be touched. Unlike an image protection a path includes the disk and directory of the image specification so it applies to only one, specific copy of the image. Wildcards are not allowed. Hitman can protect any number of paths. The path includes the device and directory as well as the image name (no file version number) as shown in the bottom line of the show process/continuous output.

Protecting processes by name

Use the /PROTECT/PROCESS=("process name",...) qualifiers to indicate which processes are not to be touched. The quotes around process name are required if the process name contains embedded blanks or special characters. Wildcards are allowed. Hitman can protect any number of processes.

NOTE: Users can change their process name easily and without restrictions. If users know which process names are protected, they can "acquire" protection by changing their process name. Please restrict access to the HITMAN parameters if you are using process protection.

Protecting system processes

Use the /SYS_PROTECT qualifier to protect all interactive system processes (UIC group is less than or equal to MAXSYSGROUP). See the chapter on COMMANDS for more information.

Protecting detached processes

HITMAN protects detached processes by default. Use the /COMMON=common_field qualifier to protect an idle, interactive parent process by linking it to an active, detached process through a common field such as username, UIC or server ID. See the chapter on COMMANDS for more information. You can enable HITMAN to terminate detached processes using the /TYPE=DETACHED command

Protecting network processes

HITMAN protects network processes by default. Use the /COMMON=common_field qualifier to protect an idle, interactive parent process by linking it to an active, network process through a common field such as username, UIC or server ID. See the chapter on COMMANDS for more

information. You can enable HITMAN to terminate network processes using the /TYPE=NETWORK command.

Protecting all processes EXCEPT for a few

Use the /EXCLUDE qualifier to reverse the protection for a specified list. Normally, the entries in a list are protected. When you specify /EXCLUDE for the list, everyone in the list is eligible to be hit and everyone not in the list is protected. For example,
the command:

```
$ HITMAN /PROTECT /USER=FRED /EXCLUDE=USER
```

protects everyone but FRED. Use the /EXCLUDE qualifier when you only have a few people you want to hit and many people you want protected. You can choose to exclude by the username protection list, as in this example, or by the identifier, image, process, server, terminal or UIC protection lists.

NOTE: When you use /EXCLUDE on a list, all other protection lists are disabled. You may only have ONE list excluded at a time.

Protection versus AUTHORIZE and AUTOHIT

All these qualifiers override /AUTOHIT and /AUTHORIZED qualifiers as well as idle checks. A protected user is allowed to run without interference from HITMAN.

Killing or disconnecting idle processes

This chapter shows you what actions you can specify when HITMAN has classified a process as "idle".

If you run HITMAN in WATCH mode (/HIT=WATCH), warning and termination records are written to the log file but users are not warned or killed. Since the log record contains the HIT mode setting, you can always tell whether the process was actually killed. Use WATCH mode to test HITMAN on your system.

If you run HITMAN in WARN mode (/HIT=WARN), warning and termination records are written to the log file and the user is warned but not killed. Use WARN mode to "gently" remind users that they should remain active or log off.

If you run HITMAN in HIT mode (/HIT=HIT), warning and termination records are written to the log file and users will be warned and either disconnected or killed.

If you enable the disconnect feature by specifying /USER/DISCONNECT or /IMAGE/DISCONNECT commands, HITMAN disconnects the process from its terminal. Otherwise, it terminates the process. NOTE: This feature is only available in the VAX version of Hitman.

Killing processes
Use the /TERMINATION qualifier to specify what you want HITMAN to do when it is terminating processes.

By default HITMAN kills processes by executing the system service $FORCEX once or twice if the process is executing an image and then executing the system service $DELPRC. OpenVMS will then automatically run-down any executing image, release locks and remove the process from the system.

If the process is running an image and the image cannot be forced to exit HITMAN sends a second forced exit. If the image still does not exit, the process is killed. If the process is waiting for input and is terminated, the process gets a read error and then does an image run=down.

The /FORCE_WAIT qualifier defines the wait between the $FORCEX and $DELPRC. HITMAN only uses the amount of time it requires to do a $FORCEX. If an image rundown is not complete by this time, HITMAN will delete the process and write a log record indicating this. If your site is handling many ALL-IN-1 users, you may want to increase the /FORCE_WAIT time to 10 seconds.

Use the qualifier /TERMINATION=FORCE_EXIT to force an image exit and run-down without

killing the process. This is useful for systems with captive accounts.

Disconnecting processes (VAX only)

Disconnecting terminals prevents security problems when users leave terminals unattended while still allowing them quick, secure access to their existing process.

HITMAN can disconnect idle processes from their terminals instead of terminating them. Users simply log back in through the normal OpenVMS facility and attach to the image. Data integrity is guaranteed; there is no data loss of any kind! Terminal disconnect allows the system manager to specify very low idle times. This sharply reduces the "window of opportunity" during which terminals are unattended and could be used by someone other than their owner.

If the user does not re-connect to the terminal within a time limit you set with the /TIMEOUT qualifier, HITMAN terminates the process automatically. The SYSGEN parameter TTY_TIMEOUT (default 15 minutes) controls how long OpenVMS waits before terminating a disconnected process. If you don't specify /TIMEOUT, OpenVMS deletes the process after TTY_TIMEOUT minutes.

Terminal disconnect is an optional feature. If you do not specify any disconnect commands, HITMAN automatically terminates idle processes.

NOTE: You must have the OpenVMS "virtual terminal" option in effect to be able to use the HITMAN disconnect feature. For more information on virtual terminals, please refer to the appendix "Virtual Terminals".

Specify which images and/or usernames are to be disconnected rather than terminated:

```
$ HITMAN/USER=(LARRY,M*,SHEMP)/DISCONNECT/TIMEOUT=15
$ HITMAN/IMAGE=ING*/DISCONNECT
```

By default, HITMAN terminates disconnected processes within 30 minutes if the user does not re-connect. Use the /TIMEOUT qualifier to change this limit.

You can choose to have processes running HITLOCK protected or treat them as normal processes.

Sending a termination message

HITMAN sends a termination message to the process before it deletes it.

The /FINAL_MESSAGE qualifier specifies the text of the message.

The /NOFINAL_MESSAGE qualifier suppresses the final message.

Clearing the screen

The /CLEAR qualifier clears the screen before the message is sent. This prevents anyone from reading the previous contents of the screen.

Sending mail to terminated users

The /MAIL qualifier sends a mail message to the user telling them that their process was terminated. HITMAN uses the OpenVMS callable mail interface to send mail quickly and efficiently. Previous versions of HITMAN used a batch queue and the MAIL.COM procedure.

You can provide a site-specific text message to be appended to the user mail message that is sent. Insert your message in the ASCII text file CUSTOM.TXT which is appended to each mail message that is sent to a user.

You can send the mail message to a distribution list as well as to the terminated user. If the file DISTRIBUTE.DIS exists, HITMAN uses it as a mail distribution list. DISTRIBUTE.DIS must be an ASCII text file containing a list of usernames. The usernames must be in a vertical column and start at the left column. For example, the following file:

```
FRED
LARRY
```

would send the mail message to the two users listed as well as the terminated user.

CUSTOM.TXT and DISTRIBUTE.DIS must be in the directory pointed to by the logical HITMAN_DAT.

New to Version 10 of Hitman is limited support for nonOpenVMS mail addresses. At sites using hooks into SMTP mail and other mail packages you may enter a “literal” mail address in a distribution list. Any mail recipient beginning with the colon character “:” will be treated as literal and Hitman will not verify that it is a valid OpenVMS username.

Handling DECSERVERS

If you are using HITMAN with users logged in through a DECSERVER and you want them logged out at the DECSERVER level, type the following commands at the Local prompt to set the port and server characteristics:

```
Local> Define Server Inactivity 30
Local> Define Port X Inactivity Enable
```

The first command defines the amount of time, for the server, a user can remain at the Local prompt before being logged out. The second command enables "Port X" to be timed out due to inactivity. By default, the port is disabled for inactivity timeout. This feature only works if there are no incoming sessions logged in to that port. If a user has an incoming session, the port will not

be timed out. If you are using a dial-out modem connected to a port, it will be timed out anyway. You may also use port settings to pass the dialup bit to VMS so that you can take advantage of HITMAN's special wait and warn or connect times for dialup processes.

To issue these commands for ports other than your own, you must be able to do a "SET PRIVILEGED", which means you need the DECSERVER privileged password.

How HITMAN handles DECWindows

This chapter explains how HITMAN handles DECWindows. Although there are two variants of DECWindows (DECWindows and DECWindows/MOTIF), HITMAN's DECWindows handling applies mainly to MOTIF users as this is the more widely used option.

HITMAN, by default, links the multiple processes associated with DECWindows together (equivalent to /COMMON=DECWINDOWS). This results in processes like the window manager and session manager being treated as active while an active process, such as a DECTerm session, exists. Since there is a bit of processing overhead associated with this feature, sites that are not running DECWindows can eliminate this overhead by defining the system logical HITMAN$NODECWINDOWS to be "TRUE".

```
$ DEFINE /SYSTEM HITMAN$NODECWINDOWS "TRUE"
```

DECWindows MOTIF

MOTIF users should define the logical Hitman$DECWindows_command_procedure in the system logical table to point to the command procedure they want submitted when HITMAN takes action on an idle user. ***HITMAN submits the job to the batch queue specified by the /DECWINDOWS qualifier.*** There are two sample procedures in the HITMAN_COM directory: Pausesession.com brings up the MOTIF pause session window and Endsession.com cleans up any processes in the DECWindows session, stops the session manager and displays the MOTIF login window. If the logical isn't defined Pausesession.com is submitted.

In a multi-user MOTIF environment, ensure the /DECWINDOWS batch queue has a high /JOB_LIMIT value because the PAUSESESSION.COM command procedure remains executing in the queue until the user specifies their password and is reconnected. Failure to ensure a high /JOB_LIMIT may result in pause sessions jobs pending in the queue. When a process slot is available and the pause session executes, the user is then unexpectedly paused. Although they can enter their password and resume with no negative effects, it is jarring to most users to be paused unexpectedly. The ENDSESSION.COM procedure is only in the queue until the LOGIN window comes up so the /JOB_LIMIT can be lower.

Standard DECWindows

In the standard DECWindows environment, HITMAN submits the DECW$STARTUP command procedure with the parameter "RESTART" whenever it detects an idle DECWindows user. ***HITMAN submits the DECW$STARTUP procedure to the batch queue specified by the /DECWINDOWS qualifier.*** This procedure, supplied by Digital, cleans up the associated processes, stops the session manager and then displays the login window. DECW$STARTUP is only suitable for the workstation environment, multi-user standard DECWindows users should define the system logical HITMAN$NODECWINDOWS to be "TRUE". With this logical defined HITMAN cleans up idle DECTerm sessions, etc. but does not submit the DECW$STARTUP procedure.

In a clustered environment with multiple workstations you can enter a logical name instead of a queue for the batch jobs to be submitted to. Simply define that logical name on each node to be a queue that runs on that node:

```
$ HITMAN /DECWINDOWS=DECW$BATCH
$ DEFINE/SYSTEM DECW$BATCH NODE$BATCH
```

and add the last line to SYS$MANAGER:HITMAN_SITE_LOGICALS.COM so it gets redefined during system startup.

Killing users after a specified amount of time

With HITMAN's connect time feature, you can limit the amount of time users may be on the system in a single session. Sites with a limited number of dialup lines can use this feature to prevent users from monopolizing a phone line by forcing them to log off when they reach their connect time limit. To continue, they must redial the system and log back in.

HITMAN can warn and terminate users after a specified period of time regardless of activity. The /CONNECT qualifier controls this function. It is specified in combination with the /DIALUP, /IDENTIFIER, /IMAGE, /SERVER, /TERMINAL, /UIC, /PROCESS, /PATH and /USER qualifiers to build lists of connect time-restricted users, ports, etc. When /CONNECT time is in effect, a user is warned at the times specified by /FIRST_WARN and /SECOND_WARN and is then terminated at the /WAIT time regardless of activity.

To limit connect time to 20 minutes on terminal ports TTA0: through TTA7: enter:

```
$ HITMAN /TERMINAL=TTAx /CONNECT /FIRST_WARN=12 -
      /SECOND_WARN=16 /WAIT=20
```

for each terminal port. If TTA0: through TTA7: are the only TTA devices on your system you can wildcard the terminal specification:

```
$ HITMAN /TERMINAL=TTA* /CONNECT /FIRST_WARN=12 -
      /SECOND_WARN=16 /WAIT=20
```

The system wait and warn times are for idle processes only and do not apply to connect times. To establish a system-wide connect time limit of 1 hour, enter:

```
$ HITMAN /USER=* /CONNECT /FIRST_WARN=50 /SECOND_WARN=55 -
      /WAIT=60
```

HITMAN's protection tables override connect time limits in exactly the same way they override idle time limits. HITMAN never hits protected processes.

Using HITMAN to Terminate Non-interactive Processes

By default, HITMAN considers only interactive processes eligible to be terminated. However, HITMAN monitors and terminates any combination of the following process types:

INTERACTIVE
DETACHED
BATCH
DIALUP
NETWORK

Several third party software products create processes which are not the standard INTERACTIVE type. These processes frequently show up as DETACHED or NETWORK. You can use HITMAN to manage these types of processes. We recommend that you run HITMAN in the default mode, monitoring only INTERACTIVE processes, unless there are processes on your system that are not being hit because they are not INTERACTIVE. The sections on "Solving HITMAN problems" and "Getting information on what HITMAN is monitoring" (page 211) explain how to determine what the type of a process is.

PLEASE BE CAREFUL!

Use this function with caution since it can cause problems on your system if it is used incorrectly. For example, making NETWORK processes eligible to be terminated may enable HITMAN to terminate DECNET processes that may have limited activity but are necessary for a Cluster to run. Making DETACHED processes eligible to be terminated may enable HITMAN to terminate critical SYSTEM processes such as the JOB_CONTROL and OPCOM processes. Run HITMAN in WATCH mode for several days with the new /TYPE options in effect before analyzing the log to determine if any processes of the newly eligible types need to be protected. The HITMAN protection lists always take priority over any other option.

```
$ HITMAN/HIT=WATCH
$ HITMAN/TYPE=(INTERACTIVE,NETWORK)
```

Before enabling HITMAN to terminate DETACHED processes, we **strongly** recommend that you use HITMAN's /SYS_PROTECT qualifier. This protects all processes on the system in the UIC groups from 1 to the current value of the sysgen parameter MAXSYSGROUP. Run HITMAN in WATCH mode and analyze the log to make sure no other processes outside these UIC groups could be terminated that could create problems for your site.

```
$ HITMAN/SYS_PROTECT
$ HITMAN/HIT=WATCH
$ HITMAN/TYPE=DETACHED
```

Using HITMAN on a cluster

This chapter shows you how HITMAN runs on Cluster systems.

Each node runs a separate copy of HITMAN which only checks for idle processes on the node on which it is running. If you do not install HITMAN on a common drive, you must install it on each node separately.

LOG FILE

Each node in the cluster must have its own log file. The log file is not set up for shared sequential access and idle processes are specific to a single node. HITMAN creates a unique log file name by using the node name as the file extension.

PERMANENT DATA FILE

You can share the permanent data file across the cluster or set up a separate permanent data file for each node. Any number of nodes can read a common data file without problems. Only one node can write to the file. HITMAN will signal an error if more than one node tries to write to the file. When you change the permanent data file, all nodes will read it at the start of the next data collection.

ERROR and OUTPUT LOG

These files are created by the detached process if an error occurs. They are specific to the node HITMAN is running on.

NODE NAME

HITMAN uses the node name defined in SCSNODE as a file extension to make the node-specific files unique. If the node name does not appear as the file extension for the log file, check that SCSNODE is defined in SYSGEN.

VAX and AXP

HITMAN must be installed on at least one VAX and one AXP in a mixed cluster, however, they can be installed to the same directory (only the executables are different in this case and they are installed in separate directories). HITMAN parameters can be shared between VAX and AXP systems; as explained above all node specific files will include the node name somewhere in the file name.

Mixed Versions of OpenVMS

HITMAN may be run in a cluster with mixed versions of OpenVMS, however, between major releases (5.x and 6.x or 6.x and 7.x) the executable images are incompatible. You can create a separate exe directory for one of these versions (the one that HITMAN wasn't installed under), redefine HITMAN_EXE on that node and relink HITMAN to build executables for that version of OpenVMS. Simply redefine HITMAN_EXE on the other nodes running that version to point to

the new executables. The parameter files can be shared in a mixed cluster with no problems.

Using different parameters based on time or date

This chapter shows you how to use HITMAN's prime/non-prime data files to control HITMAN's activities during various times of the day as well as on weekends and holidays.

Many sites are busy during working hours and relatively idle during the rest of the day and on weekends. To accommodate this, HITMAN allows you to have one set of parameters during working hours ("prime") and a different set during the rest of the day ("non-prime").

If you do not specify prime times, HITMAN uses the prime data file settings continually.

Here are a few of the ways HITMAN users utilize prime/non-prime:

- have short idle times during the day and longer ones during non-prime
- have HITMAN in WATCH mode during prime and in HIT mode during non-prime
- protect some users during the day but hit them during non-prime if they’re idle.

To define the prime period, specify a prime start, /PRIME_START, and end, /PRIME_END, for each day (/DAY) of the week. **HITMAN considers times outside this range to be non-prime. You do not need to specify non-prime as well as prime.** For days such as weekends (SATURDAY and SUNDAY), you can designate the entire day as non-prime, /NODAY=WEEKEND. You can specify up to 30 dates such as Christmas or January 1st which HITMAN is to treat as non-prime regardless of the day of the week, /HOLIDAY=date.

The distribution kit contains sample prime and non-prime data files. By default, prime time starts at 8 a.m. (/PRIME_START=8:00) and ends at 5 p.m. (/PRIME_END=5:00) for weekdays (/DAY=WEEKDAY) with weekends being non-prime.

You must specify the three qualifiers /DAY, /PRIME_START and /PRIME_END together. You must not omit any of them.

The prime start and end times are stored in both files so HITMAN handles the transition automatically. You do not have to do anything other than set the prime times and designate the non-prime days. HITMAN automatically loads the appropriate file. When HITMAN switches times it does not reset its accumulated process information. As a result, if you use a lower wait time in non-prime, as soon as HITMAN switches to non-prime they may be eligible to be hit. In this case HITMAN sends the user **one** warning if they have not received any warnings and then terminates the user's process.

NOTE: When you modify a permanent data file, be sure to specify which data file (prime/non-

prime) you want to modify. At the command line, use the /FILE=PRIME or /FILE=NONPRIME qualifiers to specify which data file you want to modify or list. If you do not specify, HITMAN assumes you want to change whichever file is **current** (based on your /PRIME_START and /PRIME_END settings) at the time you make the request. If you want to make the same change to both files, specify /FILE=BOTH when you enter the command. When you change prime start and end times, HITMAN writes the values into both files automatically since they must always be the same in both files.

Specifying times

Specify times using a 24-hour clock in the format "HH:MM". For example, "8:00" refers to 8 AM and "17:00" refers to 5 PM. When you enter a time, you must specify both hours and minutes (for example "8:30") but you cannot specify seconds or fractions of seconds. When you specify a single digit hour, you do not have to include a leading zero ("9:00" is equivalent to "09:00").

Specifying dates

Specify dates using the convention "DD-MMM-YYYY". For example, "12-MAY-2002". You must specify the entire date **including the year**. For days less than 10, you do not have to include the leading zero ("4-MAY-2002" is equivalent to "04-MAY-2002").

Enabling and Disabling prime/non-prime processing

You enable prime/non-prime processing with the qualifier /ENABLE_PRIME. To disable prime/non-prime processing, use the qualifier /DISABLE_PRIME. When prime/non-prime processing is disabled, HITMAN uses the prime data file settings continuously rather than checking the time and loading the prime and non-prime files as appropriate.

Setting up prime/non-prime files

Here's a procedure to set up prime and non-prime files using your existing permanent data file:

1. Set your default to the HITMAN data directory

```
$ SET DEFAULT HITMAN_DAT
```

2. Copy your existing permanent data file as non-prime

```
$ COPY PERM_DATA_PRIME.DAT PERM_DATA_NONPRIME.DAT
```

3. Define your prime start and end times

```
$ HITMAN/PRIME_START=07:00/PRIME_END=17:00/DAY=ALL
```

4. Set parameters in the non-prime file

```
$ HITMAN/HIT=WATCH     (changes whichever file is current)
```

$ HITMAN/FILE=NONPRIME/WAIT=n/FIRST_WARN=n/SECOND_WARN=n

.

.

5. List both files to ensure you have the correct times and settings

$ HITMAN/FILE=PRIME/LIST
$ HITMAN/FILE=NONPRIME/LIST

How HITMAN determines whether to use prime or non-prime

HITMAN uses the following logic at the start of each data collection cycle to determine if the system time is prime or non-prime:

1. If the day is a holiday, the time is non-prime.

2. If the day of the week is designated non-prime, the time is non-prime.

3. If the time is outside the prime start/end for that day of the week, the time is non-prime.

4. Otherwise, the time is prime.

Controlling HITMAN's operations

This chapter shows you how to control HITMAN's operations including setting the data collection and log file update intervals and allowing automatic response to system load.

Starting HITMAN automatically when your system boots

To start HITMAN automatically when your system boots, add the following commands to the site-specific startup file SYS$MANAGER:SYSTARTUP_V5.COM (SYS$MANAGER:SYSTARTUP_VMS.COM under OpenVMS V6.x):

```
$ @SYS$MANAGER:HITMAN_STARTUP.COM
```

This command procedure:

- executes the HITMAN_PKMS_START procedure to load HITMAN's license
- defines the HITMAN logicals in the SYSTEM table
- installs the command definition in the process table
- starts HITMAN

Handling multiple parameter updates

If someone else modifies the permanent data file from the DCL prompt or the menu interface while you are modifying the file, HITMAN will notify you that you are about to overwrite the changes that have been made since you read the file. HITMAN will write the old permanent data settings to PERM_DATA_PRIME.MODIFIED in the HITMAN_DAT directory. HITMAN will then write out its parameters, with all of your modifications, as it was when you entered this menu.

Choosing a data collection interval

HITMAN collects data on processes, checks for events, warns and kills idle processes and then hibernates for a given number of minutes. The HITMAN default of 1 minute intervals will work in most cases. Use the /INTERVAL qualifier to specify the time between data collections. The "best" interval for your system depends on your wait and warning times and whether you check for critical events such as device errors or low free blocks. Here are some sample data collection intervals:

WARN/WAIT	**INTERVAL**
25 or more	5
20 - 24	4
15 - 18	3
10 - 14	2
5 - 9	1

The first column refers to your earliest warning time (or your earliest wait time if you have

warnings turned off). Remember, all your wait and warn times have to be a multiple of the data collection interval. There are numbers within these ranges which are prime and therefore are not multiples of anything.

Choosing a log file update interval

The /UPDATE_INTERVAL qualifier specifies the interval at which the log file is updated. The default is 15 minutes. This should be adequate for most sites. Log messages are buffered until the log file is updated. Once the log file has been updated, you can view the log messages in the file. If you want to watch what HITMAN is doing, set this interval down. If you want to run with minimum resource load, set this interval up. The /LOG qualifier will close and reopen the log file so you can look at it. Otherwise, it will be updated every update interval and when you stop HITMAN.

Turning data collection on and off

The /ON qualifier turns data collection on. This is the default.

The /OFF qualifier turns data collection off. HITMAN remains on the system but does not collect any data, check for events or kill idle users. Use this qualifier to make HITMAN inactive without stopping it.

Making HITMAN sensitive to system load

Use the /MINIMUM_PROCESS qualifier to turn data collection on or off depending on your system load. HITMAN remains on the system but only collects data if there are a specified minimum number of processes on the system. For example:

```
$ HITMAN /MINIMUM_PROCESS=12
```

HITMAN will only be active if there are 12 or more interactive processes on the system. Use this qualifier to have HITMAN "wake up" when the load increases and leave your system alone when it is lightly loaded.

Setting HITMAN's priority

To ensure accurate data collection, the HITMAN detached process runs at a higher priority than the processes it is monitoring. The default is 6 as opposed to the normal interactive default of 4. Unless you have very unusual requirements, do not alter the HITMAN priority. Use the /PRIORITY qualifier to specify the priority at which HITMAN runs on your system. If you plan to use the Hitman Runaway Process event Hitman must be at priority 5 or higher - at priority 4 a runaway process will consume so much of the CPU that Hitman will not be able to get enough resources to submit it’s action procedure in response.

Controlling swapping

To collect information on a process, HITMAN makes the process current. If your system has many outswapped processes, HITMAN may cause unnecessary swapping resulting in system

resource wastage. Use the /NOSWAP qualifier to prevent HITMAN from swapping processes in. Note however, that HITMAN cannot collect information on processes unless it swaps them in. As a result, HITMAN cannot accurately check for idleness when /NOSWAP is specified. Processes which are actually idle may not be flagged as idle if HITMAN skips collecting data on them because they are swapped out.

Changing HITMAN's name

You can have HITMAN run in "stealth" mode on your system. Use the /NAME qualifier to change HITMAN's process name to prevent users from seeing it on your system. When you change HITMAN's process name, the new name is written in both the prime and non-prime data files.

If you have changed the names of the permanent data or log files by reassigning the logicals, you cannot use this option.

Analyzing HITMAN's log file

This chapter shows you how to monitor HITMAN's activity by analyzing HITMAN's log. HITMAN records every warning, process termination and event in a log file. Each log record contains information such as: username, image, terminal, server, process name, date & time record was written. The log file is a sequential, unformatted FORTRAN file. You must use a program to analyze it. You cannot TYPE it to our screen or look at it using a text editor.

You can analyze the log file by typing "HITMAN/MENU" to activate the screen interface and selecting the "Analyze Log" function. With this function, you can select records by one or more of the following criteria: username, date range, event type (warning, termination, event occurrence, etc.), terminal name, server/port ID, image name or UIC and view or report them.

The Hitman log file summary report provides a complete summary of the contents of a log file. It includes how many times each user received warnings and was terminated as well as how many times each event was detected, etc. This report is particularly useful when running Hitman in watch mode since it summarizes all the actions Hitman would take.

At the command line, use the HITMAN/REPORT qualifier to create an ASCII version of the entire log file. Use the SEARCH command or text editors to find information in this file.

At the command line use the /ASCII qualifier to create an ASCII export version of the log that is suitable for importing into many popular database, spreadsheet and reporting packages. The default output format is comma delimited with text strings enclosed in double quotes ("). You can change the delimiter character by defining the logical HITMAN$DELIMITER to be the desired character. The following tables list the fields contained in this ASCII dump format and provide translation tables for the two code fields - log record type and event type.

Use the qualifier /LOG to close the existing log file and start a new one. Use the qualifier /NOLOG to close the log file without starting a new one. Please note that if you use this qualifier, you will not have any record of HITMAN's activities on your system.

When HITMAN starts up, it opens a new version of the log file in the directory pointed to by the logical HITMAN_DAT. To open the existing log file, define the logical HITMAN$LOG. Type the following command:

```
$ DEFINE /SYSTEM HITMAN$LOG "KEEP"
```

By default, the log file is called HITMAN_DAT:HITMAN_LOG_FILE.{node_name}. To change its name, define the system logical name HITMAN_LOG_FILE to be the file name you want. For example:

```
$ Define /System Hitman_log_file Dua1:[Fred]hit_log.{Node Name}
```

causes the new log file to be opened as "DUA1:[FRED]HIT_LOG.{node name}".

ASCII Log Fields

Field Name	Field Type	Len	Table
Log Message Type	Integer		A
Date (MM/DD/YY)	Character	8	
Time (HH:MM)	Character	5	
PID	Character	8	
Username	Character	12	
Process Name	Character	15	
UIC Group, Member ([00000,000000])	Character	14	
Image Name	Character	39	
Terminal	Character	12	
Server	Character	64	
First Warning Time (minutes)	Integer		
Second Warning Time (minutes)	Integer		
Wait/Connect Time (minutes)	Integer		
Number of idle minutes	Integer		
Number of Subprocesses	Integer		
Hit Mode (T or F)	Flag		
Set Idle Mode (T or F)	Flag		
Disconnect Mode (T or F)	Flag		
Disconnect Time (minutes)	Integer		
Event Type	Integer		B
Event ID	Integer		
Event Queue Name	Character	80	
Event Device Name	Character	40	
Event Free Blocks	Integer		
Event Process Name	Character	15	
Event User Logged In (T or F)	Flag		
Event User Logged Out (T or F)	Flag		
Event Current Priority	Integer		
Event Node Name	Character	15	
Event Description	Character	50	
Event Action Flag (T or F)	Flag		

Event Action Procedure Name	Character	39	
Event Action Queue Name	Character	80	
Time Event; Month	Character		
Time Event; Day	Character		
Time Event; Hour	Integer		
Time Event; Minute	Integer		
Runaway Event; CPU Threshold	Integer		
Runaway Event; BIO Threshold	Integer		
Runaway Event; DIO Threshold	Integer		
Runaway Event; Number of Data Collections	Integer		
All Events; Number of Times to Submit Action Procedures	Integer		
All Events; P8 Value	Character		
State Events; Process State	Integer		C
Node Events; Watch Node	Character		
State Events; Count	Integer		
All Events; Number of Times Action Submitted	Integer		

Table C, Process States

Value	Process State	Value	Process State
1	COLPG	8	HIBO
2	MWAIT	9	SUSP
3	CEF	10	SUSPO
4	PFW	11	FPG
5	LEF	12	COM
6	LEFO	13	COMO
7	HIB	14	CUR

Table A, Log Message Type

Type of Log record	Value
Started new log file	0
terminated unauthorized privileged user (AUTHORIZE)	1
terminated image (AUTOHIT)	2
IDLE termination	3
IDLE first warning	4
time stamp	5
dump requested	6
IDLE first warning	7
IDLE second warning	8
IDLE multiple warnings	9
CONNECT termination	10
CONNECT first warning	11
CONNECT second warning	12
CONNECT multiple warnings	13
Event dump	14
Dump summary	15
Dump=list	16
Sent force exit	18
Delete process	19
Clean Delete Process	20
Time Delete Process	21
Special User Exit submitted	22
Parameter file switch (PRIME<->NONPRIME)	48
Event record	57

Table B, Event Type

Event Type	Value
Absent Batch	1
Device Error	2
Image	3
Absent Process	4
Present Batch	5
Priority Change	6
Privilege Check	7
Terminal	8
User	9
Check Free	10
Monitor Queue	11
Queue Stopped	12
Queue Stalled	13
Present Process	14
Time	15
Startup	16
Runaway	17
Default	18
Process State	19
Node	20
Repeating	21
Image Count	22

Sending messages to the system console

This chapter shows you how to control the messages HITMAN sends to the system console.

HITMAN sends a message to the system operators when:

it starts up	HITMAN/START
it shuts down	HITMAN/STOP
it changes its name	HITMAN/NAME
it creates a new log file	HITMAN/LOG

it encounters an error such as:
- NOBROADCAST set on an idle user's terminal

a process is warned

a process is killed for:
- being idle
- exceeding its connect time limit
- running a restricted image
- having an unauthorized privilege

an event occurs

it confirms it is still running (hourly time stamp message)

Some of these messages are controlled by the setting of other qualifiers. For example, if /NOALARM is specified with the autohit message, no console message will be sent when a process is terminated for running a restricted image.

Use the /CONSOLE=(type,...) qualifier to tell HITMAN to send different types of messages to operators. The default is to enable all five of the following types.

ERROR	Send error messages to the console
LOG	Send a message every time a new log file is opened
FILE	Send a message every time HITMAN switches data files from prime to nonprime or back to prime
PROCESS	Send a message every time HITMAN warns or kills a user
TIME	Send a message every hour to indicate HITMAN is running

Use the /NOCONSOLE qualifier to suppress all messages to operators or /CONSOLE=([NOERROR,] [NOEVENT,] [NOFILE,] [NOPROCESS,] [NOTIME]) to disable only specific types of console messages.

Use the /OPERATOR_CLASS qualifier to tell HITMAN which class of operator to send messages to. By default, HITMAN sends messages to the class SECURITY.

Normally, the system console is enabled for all message classes. However, if it is not enabled for

the class you are sending HITMAN messages to, you will not receive any HITMAN messages at the console.

Time stamp

In addition to normal messages, HITMAN sends an hourly time stamp message to confirm that it is still running. Use this if you want positive confirmation that HITMAN is running on your system.

The /CONSOLE=TIME qualifier tells HITMAN to send the time stamp message to operators.

Use the /CONSOLE=NOTIME qualifier to suppress the time stamp message to the console. This is the default.

If /NOCONSOLE is specified, it overrides all /CONSOLE values and no messages are sent to the console. In order to send time stamp messages to the console /CONSOLE=TIME must be specified.

To receive HITMAN messages at your terminal, type:

$ REPLY /ENABLE=message_class

To suppress HITMAN messages at your terminal, type:

$ REPLY /DISABLE=message_class

where message_class is the same as that specified in /OPERATOR_CLASS.

Here are some sample HITMAN console messages:

This message is created when HITMAN starts up, or when a new log is started using HITMAN/LOG.

```
%%%%%%%%%%%  OPCOM   7-APR-2002 16:48:44.03  %%%%%%%%%%%
Message from user LARRY
HITMAN - A new log file has been created
```

This is the "no privilege" message.

```
%%%%%%%%%%%  OPCOM   7-APR-2002 16:49:56.28  %%%%%%%%%%%
Message from user LARRY
HITMAN - Terminated User MOE      (_TXA7:MOE), Lack of Authorization
```

This is the "time stamp" message.

```
%%%%%%%%%%%  OPCOM   7-APR-2002 17:00:32.17  %%%%%%%%%%%
Message from user LARRY
This is HITMAN, see you in an hour
```

Checking HITMAN's status on your system

This chapter shows you how to get information about HITMAN as it monitors your system.

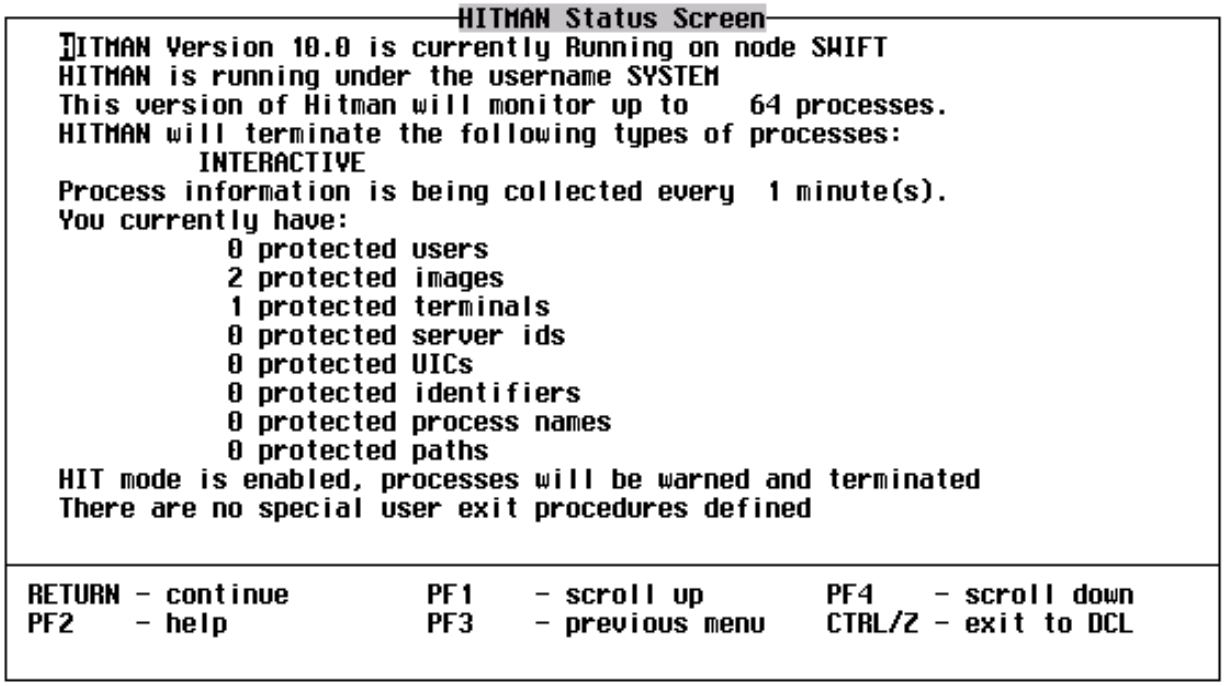
The HITMAN status command allows you to determine if HITMAN is running and what parameters it is using. If HITMAN is not operating as you think it should, do a status report to see if anything unusual is happening. If HITMAN is not running on your system the status report will clearly indicate this on the first line and information displayed will reflect the parameters that would be in effect IF Hitman were running at the current time.

To see HITMAN status information, select option "Show HITMAN Status" on the main menu. When HITMAN asks you to select a permanent data file, press <enter> to select the default.

To show HITMAN status using the command line interface, type:

$ HITMAN /STATUS

The following lines show a sample HITMAN status screen. The following page explains the contents of the screen.



The output is displayed on the screen. If the output is more than 1 full screen you can move up and down through the status display using the cursor up and cursor down keys. The first couple of lines of the screen show you:

whether HITMAN is running
the version of HITMAN you are running
the number of processes Hitman is configured to monitor
the node HITMAN is running on

Then the next 2 to 6 lines will show you what types of processes are currently being checked for activity.

The next line shows you how often HITMAN checks your system. The status report will also indicate if data collection is disabled by /OFF or /MINIMUM_PROCESS.

The next seven lines show you how many protected users, images, etc. you have. The status report will indicate if you are using /EXCLUDE qualifiers to reverse the role of protection lists.

The line following the protection lines shows you what mode HITMAN is in. If HITMAN is not hitting idle users, ensure that it is in HIT mode.

Following that, HITMAN shows you whether it is using prime or non-prime values.

Following the prime/non-prime information, the status report will produce a variety of informational messages indicating the "unusual" options such as /AUTOHIT are active.

Checking the Status of Events

The $HITMAN/DUMP=EVENT comman can be used to check the status of events to help debug events that do not appear to be working as expectedd. This command creates or appends to HITMAN_DAT:EVENT.DMP a report that shows each defined HITMAN event and whether or not HITMAN currently sees that event as true.

Checking the Status of all Processes on your System

The $HITMAN/DUMP=SUMMARY command can be useful in verifying that no processes that should be protected are now eligible to be killed after extensive changes to the parameter file(s) have been made. This command creates or appends to HITMAN_DAT:SUMMARY.DMP a report that shows each process on your system that HITMAN is monitoring, whether or not the process is idle and whether or not the process is protected from being killed.

Killing restricted images automatically

This chapter shows you how to terminate unwanted images even if they are active.

You can have HITMAN kill processes running specified images regardless of activity. This is known as an "autohit". This function could be used to eliminate game playing on your Alpha AXP or VAX during peak periods. For example, type "HITMAN/AUTOHIT/IMAGE=OTHELLO/FILE=PRIME", to terminate the image "OTHELLO" during prime time.

The /AUTOHIT qualifier defines the images which are to be stopped as soon as they are detected on your system. For example, the command:

```
$ HITMAN/AUTOHIT/IMAGE=(STAR_TREK,ADVENTURE)
```

tells HITMAN to kill the images STAR_TREK and ADVENTURE as soon as it sees them on your system.

To remove an image from the autohit list you specify /noimage with the autohit qualifier. For example, the command:

```
$ HITMAN/AUTOHIT/NOIMAGE=(STAR_TREK)
```

would remove STAR_TREK from the autohit list.

NOTE: Users running images specified in the qualifier /AUTOHIT/IMAGE=(imagename) will not be hit by HITMAN if they are protected by the image(s) being in the /PROTECT/IMAGE=(imagename) list or an entry in any other protection list (user, terminal, server, etc.). Protection takes precedence over autohit.

HITMAN does not provide the facility to do autohits on users or terminals since these can be handled readily with existing OpenVMS utilities. To prevent specific users from logging in, use AUTHORIZE to DISUSER them. To prevent specific terminals from being used, set the terminal to /NOTYPEAHD.

Suppressing the /AUTOHIT message

You can suppress the AUTOHIT message using the qualifier /NOAUTOHIT_MESSAGE.

/NOBROADCAST

If terminals set /NOBROADCAST or users have typed <CTRL/S>, the AUTOHIT message is not displayed and an error message is written to the HITMAN_ERROR file. However, processes

is still terminated.

Specifying message text

You can tell HITMAN to send a message when a process is killed. Specify the text of the AUTOHIT message using the qualifier:

```
/AUTOHIT_MESSAGE="message text"
```

If you do not specify message text, HITMAN uses the default message text:

```
*** PROCESS IS RUNNING A RESTRICTED IMAGE ** ?L?
```

The maximum length of the messages is 200 characters.

You can use message variables in AUTOHIT messages. For more information on message variables, see the chapter "Warning idle users".

You can specify qualifiers with each message to change the action of the message.

If you specify /BELL, HITMAN rings the terminal bell when it sends the message to the user terminal.

If you specify /ALARM, HITMAN sends a warning message to the system console when it sends the message to the user terminal.

If you specify /KILL, the process will be terminated. If you specify /NOKILL, the process is only warned.

Killing unauthorized privileged users automatically

This chapter shows you how to use HITMAN to guard against unauthorized users gaining privileges on your system. You can tell HITMAN to monitor your system and stop processes which have privilege but are not on your list of authorized users. Use this function as a second line of defense against hackers who appear on the system with privilege without your knowledge. This check is made every data collection interval when HITMAN wakes up..

HITMAN checks for processes with privileges under the ALL category (a subset of available privileges defined under OpenVMS). Users with one or more of these privileges are considered privileged: BYPASS, CMEXEC, CMKRNL, DETACH, DOWNGRADE, LOG_IO, PFNMAP, PHY_IO, SETPRV, SYSNAM, SYSPRV or UPGRADE. Since there are multiple privilege masks in effect for any process HITMAN allows you to specify which mask(s) should be checked. Define a system logical, HITMAN$PRIVS, to be a value from 1 to 4 to control which mask(s) are checked. The different modes are as follows:

1 This is the default if HITMAN$PRIVS is not defined and behaves exactly as HITMAN did in versions 6 and 7. This option checks the process privilege mask. It will, however, terminate any users running a privileged image IF that image also requests the privilege for the process. If the image doesn't request the privilege be granted to the process HITMAN will not detect the additional privileges.

2 This option will not detect users as privileged if they are running a privileged image that has requested privilege(s) for the process. This is less secure but may eliminate a situation where users are being detected with privileges unexpectedly.

3 This is a more secure variation of option 1. It detects if users **can** request privileges and acts on them even if they have not yet requested those privileges for their process. It adds a check of the authorized privilege mask.

4 This is a more secure variation of option 2. It also checks the authorized privilege mask and acts on processes that may request privileges.

Use the /AUTHORIZED/USER qualifiers to identify users who can run on the system with privilege. Any user with one or more privileges in the ALL category, who is not on the authorized list is stopped as soon as their process is detected.

If there are no users in the authorized list or if /AUTHORIZED/USER=ALL is specified, HITMAN does not make any privilege checks.

If /AUTHORIZED/NOUSER=ALL is specified, all users on the system are checked for privilege. Any user with a privilege in the ALL category is stopped.

LOGINOUT

During the process of logging in, a process has temporary privileges. HITMAN ignores any process running the image LOGINOUT.

MAXSYSGROUP

If the UIC group number for a process is less than or equal to the SYSGEN parameter MAXSYSGROUP, the process is treated like it has the privilege SYSPRV by default, and is therefore a privileged user. HITMAN automatically ignores these users. It checks the value of MAXSYSGROUP every time it checks for privilege. Any change you make to that parameter will take effect in HITMAN immediately.

Specifying AUTHORIZE message text

You can tell HITMAN to send a message when a processes is killed. You can specify the text of the AUTHORIZE message using the qualifier:

```
/AUTHORIZE_MESSAGE="message text"
```

If you do not specify message text, HITMAN uses the default message text:

```
*** PROCESS IS UNAUTHORIZED FOR PRIVILEGE ** ?L?
```

The maximum length of the message is 200 characters.

Suppressing the /AUTHORIZE message

You can suppress the AUTHORIZE using the qualifier /NOAUTHORIZE_MESSAGE.

/NOBROADCAST

If users have their terminals set /NOBROADCAST or have typed <CTRL/S>, the AUTHORIZE message is not displayed and an error message is written to the HITMAN_ERROR file. Processes, however, are still terminated.

You can use message variables in the AUTHORIZE message. For more information on message variables, see the chapter "Warning idle users".

You can specify qualifiers with the message to change the action of the message. If you specify /BELL, HITMAN rings the terminal bell when it sends the message to the user terminal. If you specify /ALARM, HITMAN sends a warning message to the system console when it sends the message to the user terminal. If you specify /KILL, the process is terminated. If you specify /NOKILL, the process is only warned.

If the logical HITMAN$AUTHORIZE is defined, HITMAN will disuser the user account when it terminates the process.

Files you need to use HITMAN

This chapter shows you how HITMAN files are used and the amount of disk space they require.

HITMAN DIRECTORY

The HITMAN directory contains all executables, object, documentation, command procedures, log files and the parameter file. Here is its structure:

[HITMAN	
.CDU]	Command Definition
.COM]	DCL Command Procedures
.DAT.AXP]	AXP Data Files
.DAT.VAX]	VAX Data Files
.DEF.AXP]	AXP Include Files, SOURCE licenses only
.DEF.VAX]	VAX Include Files, SOURCE licenses only
.DOC]	Documentation and HELP text
.EXE.AXP]	AXP Executables
.EXE.VAX]	VAX Executables
.LIS]	Compile Listing Files, SOURCE licenses only
.MAP]	Compile Map Files, SOURCE licenses only
.MSG]	Error Message text and object
.OBJ.AXP]	AXP Object Library
.OBJ.VAX]	VAX Object Library
.SRC.AXP]	AXP FORTRAN Source, SOURCE licenses only
.SRC.VAX]	VAX FORTRAN Source, SOURCE licenses only

It requires 5,500 blocks. If you have source, that directory requires an additional 5,000 blocks.

LOG FILE

The size of the log file is dependent on the number of inactive users, the options selected for HITMAN, the number of events and the length of time it is left open. The log file expands dynamically as messages are written to it. Normally, it is a few dozen blocks but on large systems with many events, it can get quite large.

The number of blocks initially allocated to the HITMAN log file is determined by the RMS parameter extend size. To find the value of this parameter, type "$ SHOW RMS".

If the extend size is zero, RMS computes its own default allocation. Commonly, this amount is 32 blocks but it can differ from system to system. When the file is closed, it is truncated to the amount actually used.

NOTE: We recommend you close the existing log file and create a new one at least every two weeks to keep the file size to a reasonable number of blocks. You can set up a TIME event to do this automatically and even include a PURGE/KEEP in the command procedure.

The log file is normally created and accessed in the directory HITMAN_DAT. It is normally called HITMAN_LOG_FILE.{node name}. If you wish to move this file to a different location, define the logical HITMAN_LOG_FILE to the location you want, before you start up HITMAN.

```
$ DEFINE/SYSTEM HITMAN_LOG_FILE-
  ddcu:[FRED.LOG]HITMAN_LOG_FILE.{node name}
```

This will cause the log file to be created and accessed in the directory ddcu:[FRED.LOG]. It will be called HITMAN_LOG_FILE.{node name}.

PERMANENT DATA FILE

Permanent data files are typically only a few blocks, although they can get larger as you protect more users, images, and define more events. You will always have a prime permanent data file PERM_DATA_PRIME.DAT, and a non-prime data file, PERM_DATA_NONPRIME.DAT, though you may only use the prime.

A permanent data file is normally created and accessed in the directory HITMAN_DAT. It is normally called PERM_DATA_PRIME.DAT. If you wish to move this file to a different location, define the logical PERM_DATA_PRIME to the location you want before you start up HITMAN.

```
$ Define /System Perm_data_prime ddcu:[fred.dat]perm_data_prime.dat
```

This will cause the permanent data file to be created and accessed in the directory ddcu:[FRED.DAT]. It will be called PERM_DATA_PRIME.DAT. You must specify the complete file specification. Define PERM_DATA_NONPRIME to change the location or name of the nonprime data file.

ERROR LOG FILE

The error log file is created by the detached process when it writes to SYS$ERROR. This file is called HITMAN_ERROR.{node name} and is in the HITMAN_DAT directory. Since this file records only exceptional events it is normally very small; if you discover this file is becoming large you should stop hitman, edit this file and fix whatever error is causing the error messages.

OUTPUT LOG FILE

The output log file is created by the detached process when it writes to SYS$OUTPUT. This file is called HITMAN_OUTPUT.{node name} and is in the HITMAN_DAT directory. Since this

file records only exceptional events, it frequently mirrors the contents of the error file, it is normally very small; if you discover this file is becoming large you should stop hitman, edit this file and fix whatever error is causing the error messages.

Logicals you need to use HITMAN

To use HITMAN, you must define several logicals. You can also define several logicals to control HITMAN's operations. Here is a description of each logical that may be defined on your system.

HITMAN_CDU - This logical points to the [HITMAN.CDU] directory. Used to access the HITMAN command definition table. HITMAN_CDU is optional.

HITMAN_COM - This logical points to the [HITMAN.COM] directory. Used to access the HITMAN DCL command procedures. HITMAN_COM is optional.

HITMAN_DAT - This logical points to the [HITMAN.DAT] directory. Used to access the data files and command procedures for mailing. It must be defined in the system logical table. HITMAN_DAT is required.

HITMAN_DOC - This logical points to the [HITMAN.DOC] directory. Used to access the screen interface help files. It must be defined in the system logical table. HITMAN_DOC is required.

HITMAN_EXE - This logical points to the [HITMAN.EXE] directory. It is used to access the HITMAN executables. It must be defined in the system logical table. HITMAN_EXE is required.

HITMAN_LOG_FILE - This optional logical is used to redirect where HITMAN stores the log file. When defined, it points to the log file HITMAN will use. It must be defined in the system logical table. HITMAN_LOG_FILE is optional.

PERM_DATA_PRIME - This optional logical is used to redirect where HITMAN stores the prime permanent data file. When defined, it points to the current prime permanent data file. It must be defined in the system logical table. PERM_DATA_PRIME is optional.

PERM_DATA_NONPRIME - This optional logical is used to redirect where HITMAN stores the non-prime permanent data file. When defined, it points to the current non-prime permanent data file. It must be defined in the system logical table. PERM_DATA_NONPRIME is optional.

HITMAN$LIST - This optional logical is used to override the output of a HITMAN/LIST command. If this logical is not defined, HITMAN pauses at the end of each screen of output. If you want HITMAN to list the information without pausing, type:

```
$ DEFINE/SYSTEM HITMAN$LIST "GO"
```

HITMAN$LOG - This optional logical is used to continue using the same log file when you start HITMAN. It must be defined in the system logical table. By default, HITMAN creates a new log file each time it is started. If this logical is defined, HITMAN opens the existing log file rather than creating a new one. If you want HITMAN to use the existing log file, type:

```
$ DEFINE/SYSTEM HITMAN$LOG "APPEND"
```

HITMAN$KEEP - This optional logical is used to keep the output log generated by a mail batch job if you are having trouble getting event mail to work. It must be defined in the system logical table. If this logical is not defined, HITMAN deletes the log file. If you want HITMAN to keep the batch log, type:

```
$ DEFINE/SYSTEM HITMAN$KEEP "KEEP"
```

HITMAN$AUTHORIZE - This optional logical is used to disuser a user account in your SYSUAF when a process is terminated for unauthorized privilege. It must be defined in the system logical table. If this logical is not defined, HITMAN does not alter the SYSUAF record. If you want HITMAN to disuser unauthorized users, type:

```
$ DEFINE/SYSTEM HITMAN$AUTHORIZE "DISUSER"
```

HITMAN$NODECWINDOWS - This optional logical is used to instruct HITMAN to not submit the DECW$STARTUP procedure to the /DECWINDOWS queue. As a result HITMAN cleans up idle DECWindows processes but never terminate the session manager.

```
$ DEFINE/SYSTEM HITMAN$NODECWINDOWS "TRUE"
```

HITMAN$SUSPEND - This optional logical is used to instruct HITMAN to create the detached process in such a way that it is ineligible to be suspended. This prevents a privileged user from suspending the HITMAN process; in effect turning off HITMAN without removing the process from the system.

```
$ DEFINE/SYSTEM HITMAN$SUSPEND "NO"
```

HITMAN$DECWINDOWS_COMMAND_PROCEDURE - This optional logical points to the command procedure HITMAN submits in the event of an idle DECWindows user being detected. If this logical is not defined HITMAN will use the HITMAN_COM:PAUSESESSION command procedure by default.

```
$ DEFINE/SYSTEM HITMAN$DECWINDOWS_COMMAND_PROCEDURE -
      HITMAN_COM:PAUSESESSION.COM
```

HITMAN$DELIMITER - This logical tells HITMAN what ASCII character to use as the delimiter for the /ASCII dump of HITMAN's log file. If this logical is not defined HITMAN will use the comma character "," by default.

```
$ DEFINE/SYSTEM HITMAN$DELIMITER ";"
```

HITMAN$NOPROTECT_DISCONNECT - This logical tells HITMAN to not protect disconnected processes, which is the default action. Define it in the system logical table to be "TRUE".

```
$ DEFINE/SYSTEM HITMAN$NOPROTECT_DISCONNECT "TRUE"
```

HITMAN$MAX_FORCE_EXITS - tells HITMAN the maximum number of forced exits to send to a process before terminating it. If the process exits cleanly to DCL before this number of forced exits have been sent HITMAN will terminate it then. Complex applications sometimes exit more cleanly if this logical is defined to be 2 or 3.

```
$ DEFINE/SYSTEM HITMAN$MAX_FORCE_EXITS 3
```

HITMAN$PRIVS - tells HITMAN how it should determine if a user is privileged for purposes of the /Authorized check. It should be defined to be 1, 2, 3 or 4. If this logical is not defined HITMAN will use a default value of 1. For an explanation of how each of these different values controls privilege checking please refer to the section on “Killing unauthorized privileged users automatically.”

HITMAN$MAX_TERM - tells HITMAN the maximum number of idle users it can terminate in one data collection interval. If this logical is not defined HITMAN will terminate all idle users eligigle to be hit that it discovers during a data collection. This logical is useful at sites where a large number of users become idle at one time or that are running applications that take a substantial amount of resources to run-down; using this logical you can minimize the impact of process termination on your system resources.

Defining logicals during system startup

The following logical names:

HITMAN_CDU
HITMAN_COM
HITMAN_DAT
HITMAN_DOC
HITMAN_EXE

are defined in the HITMAN system logicals command procedure which can be executed by typing:

```
$ @SYS$MANAGER:HITMAN_SYSTEM_LOGICALS
```

this procedure is also executed automatically by HITMAN's startup command procedure during system startup.

Define any other required logicals using the DEFINE command and add the definitions for them to the command procedure HITMAN_SITE_LOGICALS.COM in SYS$MANAGER. This procedure is new with Version 8 of HITMAN and any logicals defined in it will not be alterred or lost during future upgrades. The site specific logicals procedure is executed automatically by HITMAN's startup procedure during system startup.

NOTE: Defining any of the HITMAN logicals to be themselves may result in HITMAN entering an infinite loop trying to translate that logical. For example:

```
$ DEFINE/SYSTEM HITMAN$NODECWINDOWS -
      HITMAN$NODECWINDOWS
```

will cause HITMAN to enter an infinite processing loop.

Privileges you need to use HITMAN

This chapter shows you the privileges you need to do various HITMAN operations.

To start the detached HITMAN process, you need the following privileges:

- CMKRNL
- DETACH
- EXQUOTA
- OPER
- PRMMBX
- SECURITY
- SYSNAM
- SYSPRV
- WORLD

HITMAN will acquire these privileges automatically if you have SETPRV. Otherwise, you must give your account the necessary privileges by typing:

```
$ @HITMAN_COM:SET_PRV
```

NOTE: To ensure more precise data collection, HITMAN runs at priority 6 rather than the default priority of 4. To start the detached process with a base priority higher than the default priority, specified by the /PRIORITY qualifier, you require ALTPRI privilege.

If you want to change a HITMAN parameter or list the parameter file, you require SECURITY and SYSPRV.

NOTE: As part of its processing, HITMAN can submit batch jobs. To ensure these batch jobs have access to command procedures in the HITMAN directory, start HITMAN from an account which has SYSPRV as a default privilege.

Using HITMAN events

HITMAN provides a set of special functions, called events, to help you manage your systems better. Events fall into two major categories; 1) those that detect special conditions or errors when they occur and 2) time events to help automate system management. Each type is described separately.

Using HITMAN events to detect errors and fix problems

With HITMAN, you can define conditions you want to check for (such as device errors, stopped or stalled queues, users with potentially dangerous privileges, low free space on a disk, and others) plus the actions to be taken when the condition is detected. An event is usually one "condition/action" combination.

During each data collection, HITMAN checks for all the defined events and takes the action you have specified. You can specify as many events for HITMAN to monitor as you require.

When an event occurs, HITMAN records the event in its log file and takes one or more of the following actions:

- send an operator message
- mail a message to a user or a distribution list
- submit a batch job to take corrective action.

HITMAN will tell you when:

- queues are stopped or stalled (MONITOR_QUEUE)
- jobs such as the daily backup are missing from queues or are submitted under the wrong username (ABSENT_BATCH)
- a disk drive, the CPU or Memory logs an error (ERROR)
- disk free space falls below limits you have set (CHECK_FREE)
- unauthorized privileged users appear on your system (PRIV_CHECK)
- a user runs with a specified privilege such as BYPASS (PRIV_CHECK)
- users bump up their priorities (PRIO_CHECK)
- key system processes (such as ERRFMT) are missingor running under the wrong username (ABSENT_PROCESS)
- specified users log on (or off) (USER)
- someone logs into a restricted terminal (TERMINAL)
- users waste resources playing games (IMAGE)
- a process consumes a significant amount of system resources (RUNAWAY)
- a process is in an unusual state (STATE)
- a large number of processes are computable at one time (STATE)

- a node has dropped out of the cluster (NODE)
- a particular batch job has been submitted or submitted under a specific username (PRESENT_BATCH)
- a particular process is running or running under a specific username (PRESENT_PROCESS)
- a queue is not running (it has been stopped or has stalled) (MONITOR_QUEUE)
- when more than a specified number of users are running a specific program (IMAGE)

There are also events to help you manage your system by:

- Taking action whenever HITMAN starts (STARTUP)
- Taking action every specified number of minutes (REPEATING)
- Taking action at a specific time every day, weekday or on a particular day of the month (TIME)

Defining events

The following pages show examples of the various events being entered from the DCL prompt and the menu interface. We recommend you use the menu interface because it displays prompts for the items necessary to set up or modify an event. This makes it easier to use and understand when you are creating or modifying events. Version 8 includes the ability to modify most event fields, individually or globally, from the DCL level.

When adding events there are a number of data items common to all events as well as event specific values. Many of these data items have default values which are used if they are omitted when adding them through the DCL interface. The menu interface will display default values and allow you enter a different value. The common items found in all events control the actions taken when the event is true.

Specifying Node

When adding HITMAN events the menu always allows you to enter a node on which the event should be signalled. It is also possible to enter a nodename from the DCL level. Unless you want the actions specified for an event to be taken on every node that is capable of detecting the event you should enter a nodename. This tells HITMAN to take action on an event only if it's detected on the node you specified. For instance adding a check free space event for a disk that is mounted on five nodes with HITMAN using a common permanent data file and running on all five nodes; if the event is set to send 3 messages and you didn't specify a node you could get a total of 15 console and mail messages for the 1 event (5 nodes x 3 messages).

Specifying an Event Description

You can specify a text description of each event you set up. This description is then used as part of the text for the mail message sent to the users you specify. The description is also submitted as the second parameter (P2) to the action procedure. The more events you set up the more helpful this field will be when reading event mail. Make sure your text is descriptive of the problem the

event detects.

Specifying Actions

You can specify several different actions to be taken when an event is true. They include any combination of sending an operator message, sending a mail message and submitting an action procedure to batch. These actions are described in the following sections.

Sending an operator message

Use the OPERATOR qualifier to send a message through OPCOM to an operator class you specify. The syntax is:

OPERATOR=class

If you do not specify a class, messages are sent to the class specified by the /OPERATOR_CLASS qualifier used to send warn and termination messages.

When adding events through the menu interface use your cursor keys to highlight /OPERATOR and then enter the class.

Sending a mail message

Use the MAIL qualifier to mail a message to a specified username. The syntax is:

MAIL=user

You can specify either a user name or a mail distribution list to send a message to several users. Distribution lists must start with the character "@". If they do not contain a logical or directory specification, HITMAN assumes the distribution list is in the directory pointed to by the logical HITMAN_DAT. Usernames must be either a valid OpenVMS username or a “literal” mail name to take advantage of mail hooks that can be built into the system (for example SMTP); literal usernames may contain any characters but must begin with the colon “:.”

HITMAN creates the mail message text in a temporary file named in the following format:

28D9EF600096B265MAIL_TEMP.TXT.

The number is the OpenVMS date and time the file was created. These files are created in the directory pointed to by HITMAN_DAT.

HITMAN also looks for the distribution list DISTRIBUTE_EVENT.DIS in the HITMAN_DAT directory. If it exists, HITMAN sends mail to all the users specified in that distribution list as well as the user or distribution list specified with the event.

You can specify an additional text message to be appended to the standard event mail message.

The text file EVENT_CUSTOM.TXT is appended to the regular mail message HITMAN before sending the mail. The EVENT_CUSTOM.TXT file exists in the HITMAN_DAT directory but is empty by default. You can edit this custom message using an editor.

When you enter an event through the menu interface use your cursor keys to highlight /MAIL then enter the username or distribution list filename.

Specifying a process state for the process state event

The process state event requires a process state; choose from the following states.

State	Description	State	Description
COLPG	Collided Page	MWAIT	Miscelleneous Wait State (includes RWAST)
CEF		PFW	Page-fault wait
LEF	Local event flag	LEFO	Local event flag; outswapeed
HIB	Hibernating	HIBO	Hibernating outswapped
SUSP	Suspended	SUSPO	Suspended outswapped
COM	Computable	COMO	Computable outswapped
FPG		CUR	Current process

Submitting a batch job

Use the SUBMIT qualifier to submit a batch job. The syntax is:

SUBMIT=(JOB=xxx,QUEUE=xxxx,USER=xxxx,KEEP,PRINT)

You must specify the job name. If you do not specify a queue, HITMAN will use SYS$BATCH.

You must design and add the actions to be take within the DCL batch job. HITMAN passes information about the event to the batch job via parameters. The first four parameters (P1 to P4) are always:

P1 event type
P2 description
P3 operator_class
P4 mail user

where:

event type — event type "ABSENT_BATCH" (for monitor queue this parameter will be either "QUEUE_STOPPED" or "QUEUE_STALLED")
description — description entered by the user
operator_class — operator class the message was sent to
mail_user — user/distribution list the mail message was sent to

The following page contains a table showing what parameters are passed as P5, P6, P7 and P8 for

the various event types. P8 is a user defined parameter that is stored within the event for each of the event types. This parameter can be used to pass site specific or event specific information to the action procedures.

Event Type	P5	P6	P7	P8
ABSENT BATCH	queue	job		User defined
PRESENT BATCH	queue	job		User defined
USER	user	terminal	image	User defined
TERMINAL	user	terminal	image	User defined
IMAGE	user	terminal	image	User defined
IMAGE COUNT	image	count		User defined
ERROR	device			User defined
CHECK_FREE	device	free_blocks		User defined
ABSENT PROCESS	process_name			User defined
PRESENT PROCESS	process_name	pid		User defined
PROCESS STATE	process_name	pid	state	User defined
NODE	watched node			User defined
MONITOR QUEUE	queue			User defined
PRIO_CHANGE	priority	user	pid	User defined
PRIV_CHECK	priv_mask	user	pid	User defined
RUNAWAY	pid			User defined
TIME				User defined
REPEATING				User defined
STARTUP				User defined

queue	queue name "SYS$BATCH"
job	job name "DAILY_BACKUP"
user	username "FRED"
image	image name "HITMAN_SET"
terminal	terminal "TTA3:"
device	device name "DUA0:"
free_blocks	number of free blocks specified in the event

process	process name "LANDRYT"
priv_mask	text list of privileges "(SYSPRV,OPER)"
pid	the 8 byte process ID
state	state the process was in
count	number of processes seen running the image

Use this information to identify which event is involved and take corrective action. To help you get started there is a sample event batch procedure for each type of event in the [HITMAN.DAT] directory. For example, ABSENT_BATCH.COM is the sample procedure showing the use of an ABSENT_BATCH event. These procedures assign the job parameters to variables. Any actual actions you want to take must be added to these jobs.

When you enter an event through the menu interface use your cursor keys to highlight the /Action flag and then enter the name of the command procedure to be submitted, queue that the job should be submitted to, whether or not to keep the log and whether or not to print the log.

Specifying the number of notifications

Use the MESSAGES qualifier to control how many times HITMAN notifies you of the event. You can specify that you be notified from 1 to 98 times. If you specify 99, HITMAN will signal the event every data collection until it is no longer true. The default number of messages is 3.

Deleting events

To remove an event from the list, type:

```
$ HITMAN/NOEVENT=(ID=event_id)
```

Checking the Status of Events on Your System

To generate a dump of all your events and their current status. Enter:

```
$ HITMAN /DUMP=EVENT
```

the output is appended to a file DUMP_EVENT.DMP in the HITMAN_DAT directory and includes the event id and type, the event description, an event status and the number of times the event has been signalled, if it is currently true. This command can help you debug new events that don’t seem to be working as expected.

Debugging Events

In order to debug events it's important to understand how they work. HITMAN treats events like questions that can be answered true or false. Once an event, such as a CHECK_FREE event, is true (meaning the disk has less than the threshold number of blocks free) it remains true until the event is reset by an appropriate change. The change that will reset an event is specific to each type of event. For instance a CHECK_FREE is reset when there are more than the threshold number of blocks free on the device. ERROR events are reset when the system reboots and the error counter

is set to zero or they are triggered again if the error counter increments. ABSENT_PROCESS events are reset when the process is found on the system. It's important to keep this in mind when debugging events because the action procedure, if any, is only submitted at the time the event first becomes true.

HITMAN checks events when it wakes up to do a data collection every /INTERVAL number of minutes. The larger your interval the longer it takes for HITMAN to detect a condition or reset itself once the condition is no longer true. During debugging wait at least one complete data collection interval after each change to be sure you give HITMAN a chance to properly check the event.

To get an event status for all events from HITMAN. At the DCL level enter

$ HITMAN /DUMP=EVENTS

to create this report in the HITMAN_DAT diretory in a file called dump_event.dmp. Sample output is included below:

```
********************* Event Dump: 22-MAY-2002 12:07:35.86*********
DEFAULT Event    1
                Description: DEFAULT RECORD
                     Status: False
            Signalled Count:    0
STARTUP Event    2
                Description: quickly submit a short idle job
                     Status: False
            Signalled Count:    0
REPEATING Event    3
                Description: brief idle job
                     Status: False
            Signalled Count:   49
REPEATING Event    4
                Description: brief idle job
                     Status: False
            Signalled Count:   49
ABSENT BATCH Event    5
                Description: Daily Backup Job Missing
                     Status: True
            Signalled Count:    3
PRESENT PROCESS Event    6
                Description: DEFAULT RECORD
                     Status: False
            Signalled Count:    0
$
```

Looking at this file will show you whether or not HITMAN has detected an event to help isolate whether the problem is with the event or with a function under the event such as mail or submitting

the action procedure. A sample of the output from this report is included below; an event that is False was not been seen as True during the last data collection and no action was taken. The signalled count indicates how many times this event has been seen as true since Hitman was last started or, in the case of systems with prime and nonprime data files, since Hitman last changed data files.

The most common problems with events are:

- Specifying a generic queue for mail or action procedures may send the job to a node that it's unable to run on. Enter a queue that executes on the proper node
- Incorrectly specifying the image or procedure name creates an event that gets detected everytime HITMAN starts or never gets detected. Specify only the filename portion for either of these and be sure to enter it correctly
- Starting HITMAN under an account without the necessary default privileges causes batch jobs to fail because they can't access their own command procedure. We recommend that HITMAN run under the SYSTEM or an equally privileged account
- Making changes to an event too quickly during debugging may prevent HITMAN from reading the newly changed event and taking the appropriate action before the event is changed again. Always wait at least 1 data collection interval between changes to events to give HITMAN time to wake up and read the changes before you make additional changes
- Wildcarded queue monitoring events may be difficult to debug. The event may have been previously triggered by a problem with a different queue than the one you are using for testing. Once an event is true it will not signal another problem until it has been reset. If you have a large number of queues we don't advise creating a queue event with just an asterisk (*) for the queue; grouping your queues logically into multiple events may help

If you have problems getting an event to work, take the following steps:

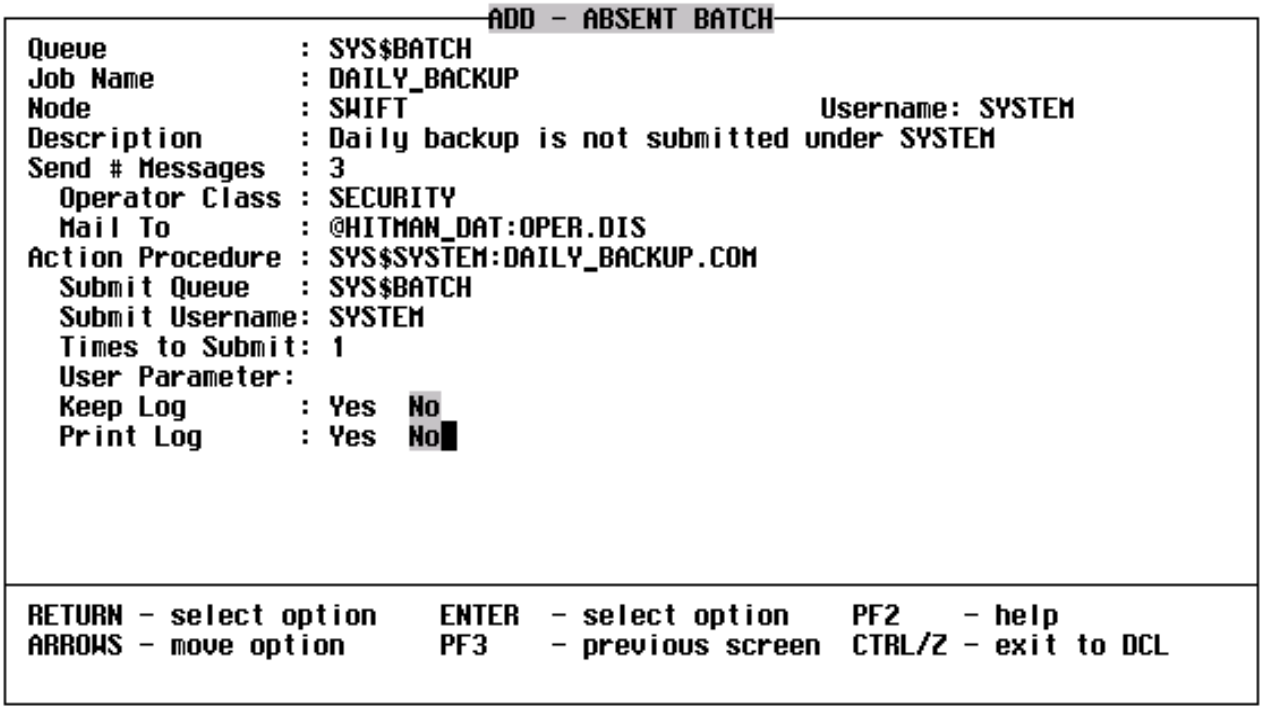
- Verify the common problems detailed above are not responsible for the problem.
- Verify the values you entered for the event specific fields like devices, terminals, usernames, job names, etc.
- Turn on console messages if they aren't already turned on and enable operator messages with /Operator for the event. Then enable yourself as an operator so you'll receive the same messages as the console. If you receive a message from the event, you know that it's being detected and the problem is in the mail or action procedure setup.
- If you've made a number of changes, stop and restart HITMAN. This guarantees that HITMAN isn't missing the event because it has already detected it and believes it has already been signalled. Normally HITMAN will detect changes to the permanent data files when it wakes up to do a data collection.
- If you added the event through the DCL interface, delete the event and add it back using the menu. We suggest this because the menu interface will prompt you for all the

possible items you may need to specify to set up your event. It will also remember the values you have entered and use them for defaults if you are planning to enter more than one event in a session. When entering events through the DCL interface, especially if you are creating a DCL command procedure to add multiple events, be thorough; specify all the appropriate parameters with complete values.

If the event is still not working as expected generate a listing of the event and call Saiga Systems technical support at 1-800-561-8876 or e-mail us at support@saiga.com.

Checking for the absence of batch jobs (ABSENT_BATCH)

The ABSENT_BATCH event checks for a batch job that should be in a queue but is not. Version



10 also adds the functionality to check that the batch job is submitted/running under a specific username; if this fields is used (not blank or not "*") the job will be considered absent if it is not found or not found under the specified username. Leaving this field blank or inserting an asterisk "*" will cause this event to work exactly as it did in previous versions.

For example, to send a console message to the operator if the daily backup procedure is not in SYS$BATCH under username SYSTEM and mail messages to all the operators in the operator distribution list you would enter an event like the one above in the menu interface or type:

```
$ Hitman/event=(Type=absent_batch, user=system, -
      Description="Daily Backup Job Missing", node=swift, -
      Queue=sys$batch, job=daily_backup, messages=3, -
      Operator=security, Mail="@hitman_dat:oper.dis")
```

In addition to the event TYPE, you must specify a QUEUE and a JOB plus the actions you wish to take. You should specify an execution queue rather than a generic queue. HITMAN only searches the specified queue. If you enter a generic queue, only the generic queue will be

searched, not the execution queues it points to. You can specify NODE if you wish a single node to check for and report the event. You can include a DESCRIPTION to remind yourself what the event was for, it also gets included in the mail message sent.

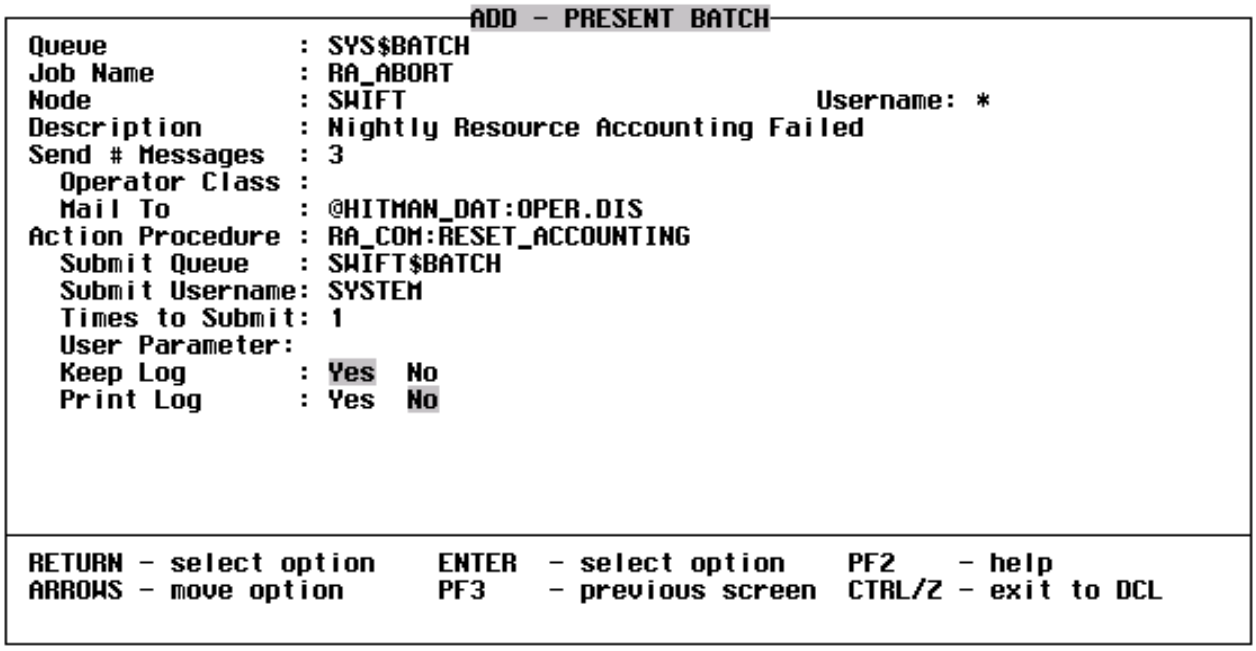
In the above menu example the event has also been updated to attempt to resubmit the daily_backup job. If this event were ID 20 you could update it at the DCL level to make this change with the following command:

```
$ Hitman /event=(id=20, submit=(job=sys$system:daily_backup.com,-
$_ queue=sys$batch, times=1, keep, noprint, user=system))
```

If you want Hitman to automatically submit a batch job for you at a certain time on specific days you can set up a time event (refer to the section on time events at page 128).

Checking for the presence of batch jobs

Use the PRESENT_BATCH event to check for a batch job that should NOT be in a queue but is. This event is the exact opposite of the ABSENT BATCH event. With Version 10 you may now specify a username if you only wish the event to be triggered if the batch job is present and submitted/running under a specific username. Leave this field blank or enter an asterisk "*" if you wish the event to function the same way it did in previous versions of Hitman.



For example, to notify the operators when the accounting submits a batch job because it aborted and then submit a batch job to reset it you could create an event like the one above; an asterisk (*) is used for the username field because we don't care what username this job is submitted under. The DCL:

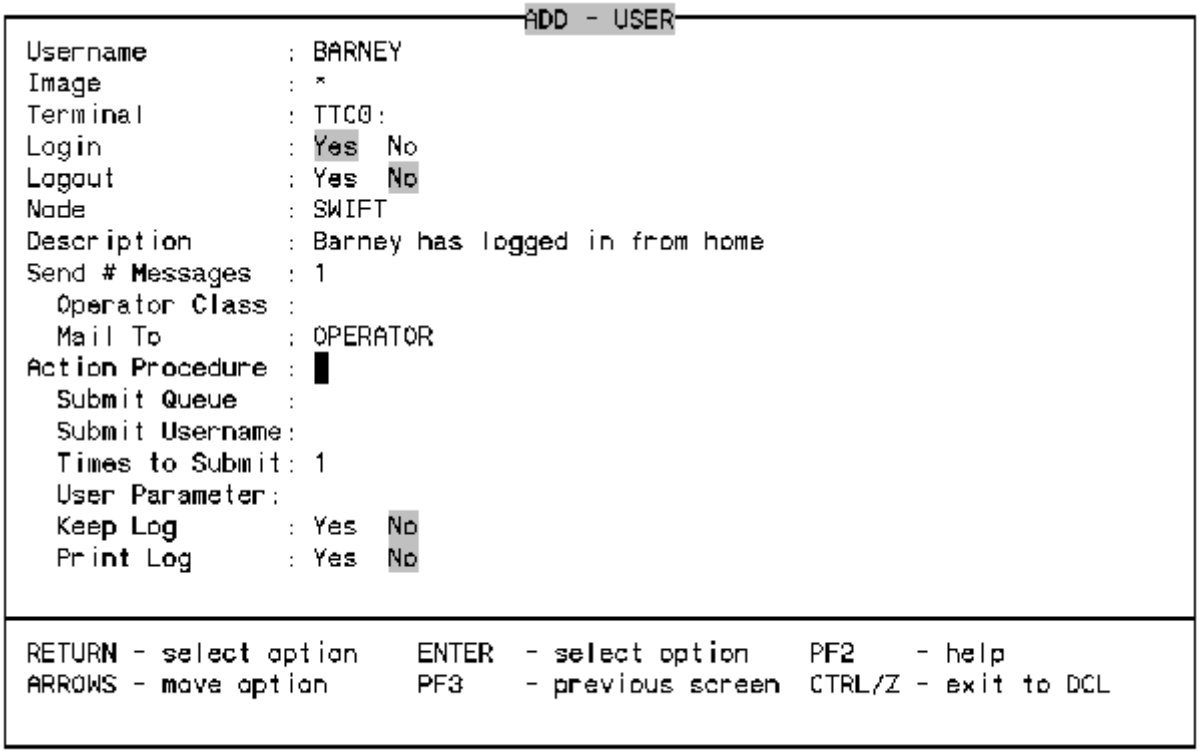
```
$ Hitman /Event=(Type=present_batch, Queue=sys$batch, user="*", -
_$ Job=ra_abort, Node=swift, Mail="@hitman_dat:oper.dis",-
_$ Messages=3, Description="Nightly Resource Accounting Failed",-
_$ Submit=(Job=ra_com:reset_accounting, Queue=swift$batch, Keep, -
_$ User=system, Noprint, Times=1))
```

In addition to the event TYPE, you must specify a QUEUE and a JOB plus the actions you wish to take. You should specify an execution queue rather than a generic queue. HITMAN only searches the specified queue. If you enter a generic queue, only the generic queue will be searched, not the execution queues it points to. You can specify NODE if you wish a single node to check for and report the event. You can include a DESCRIPTION to remind yourself what the event was for, it also gets included in the mail message sent.

Checking user status

These three event types are identical. The order of the menu prompts is changed to reflect the choice of event type but the same items are entered for each. Use these events to keep an eye on your users. You can check for a user:

- logged in (present on the system)
- logged out (absent from the system)
- running a specified image
- using a specified terminal
- running a specified image on a specified terminal



For example, to mail OPERATOR a message user BARNEY logs in through terminal TTC0: (a terminal with a modem attached), type:

$ Hitman /Event=(Type=user, User=barney, Terminal=ttc0:, Image=*, In, -
_$ Mail=operator, Description="Barney Has Logged in from Home", Messages=1)

In addition to the event TYPE, you must specify a USER plus the actions you wish to take. You can specify either IN or OUT to detect the presence or absence of the user on the system. If you specify IMAGE and/or TERMINAL the event is only true when all of the qualifiers are true. You can use wildcards on username, terminal and image name. You can specify NODE if you wish a single node to check for and report the event. You can include a DESCRIPTION to remind yourself what the event was for, it is included in the mail message sent.

Checking for terminal use

Use the TERMINAL event to check if a terminal is being used. You can also specify a user and image for this event. This event is similar to the USER event except that it emphasizes terminals rather than users.

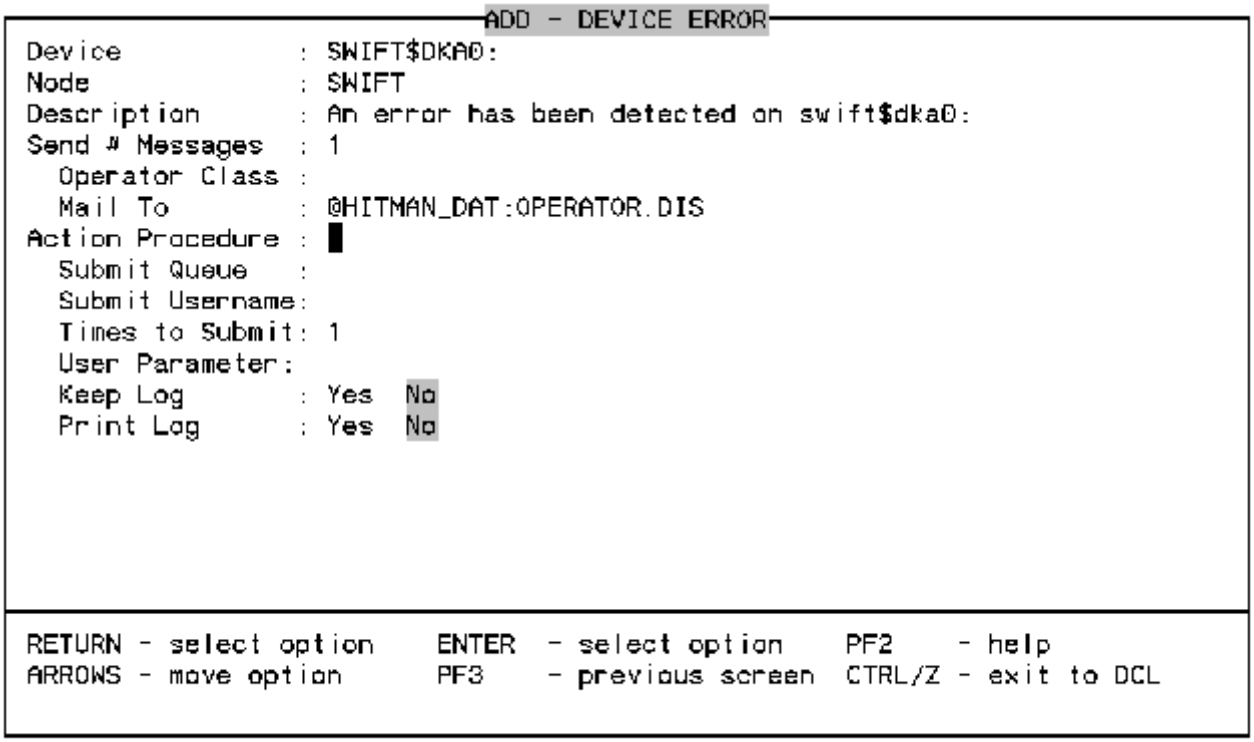
Checking for image execution

Use the IMAGE event to check for the execution of a specific image. You specify the image, HITMAN tells you when it runs and who is running it. This event is similar to the USER event except that it emphasizes images rather than users.

NOTE: These two events are essentially identical to the checking user status event on the previous page except that the menu will prompt for input in a more appropriate order.

Checking for device errors

Use the ERROR event to check for device errors. If the error count on the device you specify increases from that recorded at the previous data collection, HITMAN will take the action you specify. If you enter a logical device name, HITMAN will translate it to the equivalent physical device name. Enter “CPU:” or “MEMORY:” for the device if you wish to monitor these specific system resources for errors.



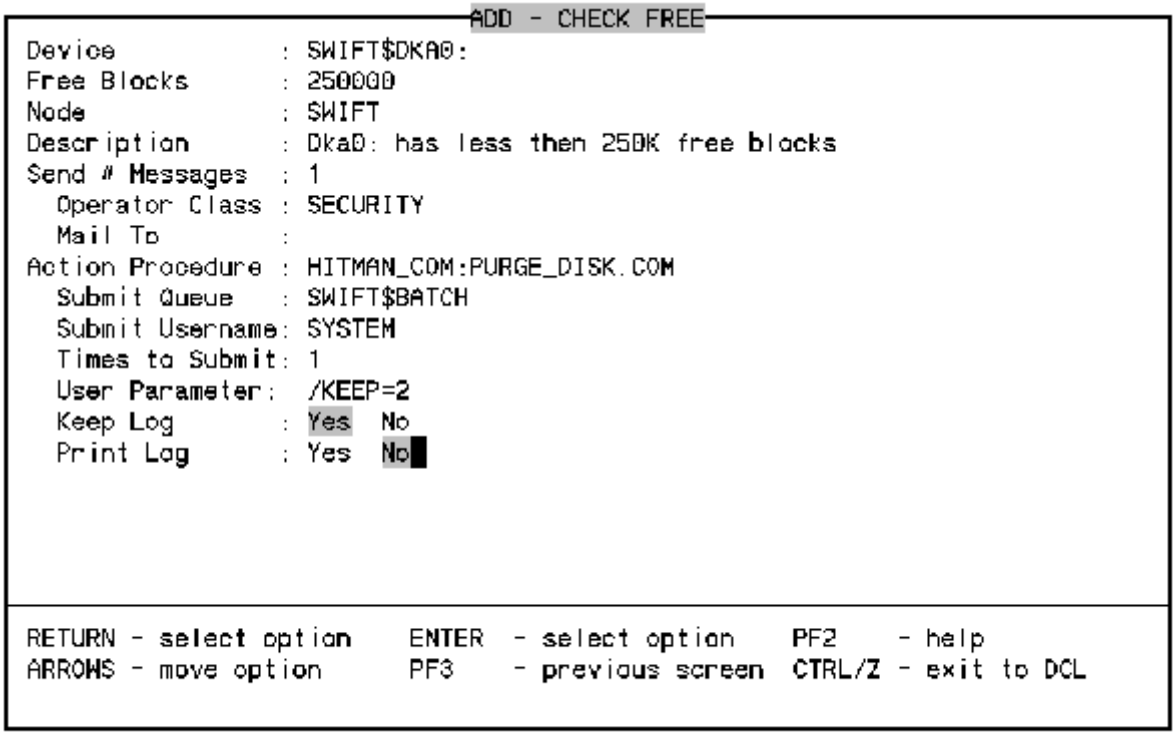
For example, to mail a message to the distribution list "OPERATORS" in HITMAN_DAT: when an error occurs on the disk drive SWIFT$DKA0:, type:

```
$ Hitman /event=(Type=error, device=swift$dka0:, -
_$ Mail=”@hitman_dat:operators.dis”, Node=swift, messages=1, -
_$ Description=”An error has been detected on swift$dka0:”)
```

In addition to the event TYPE, you must specify a DEVICE plus the actions you wish to take. You **cannot** use wildcards on the device. If you enter a logical device name, HITMAN will translate it to the equivalent physical device name. You can specify NODE if you wish a single node to check for and report the event, however, if the error occurs on a different node it will not get reported. You can include a DESCRIPTION to remind yourself what the event was for, it also gets included in the mail message.

Checking for low free space on disk

Use the CHECK_FREE event to find out when you are getting low on disk space. If the number of free blocks is less than or equal to the number you specify, HITMAN will take the action you specified (for example, purge the disk using a batch job).



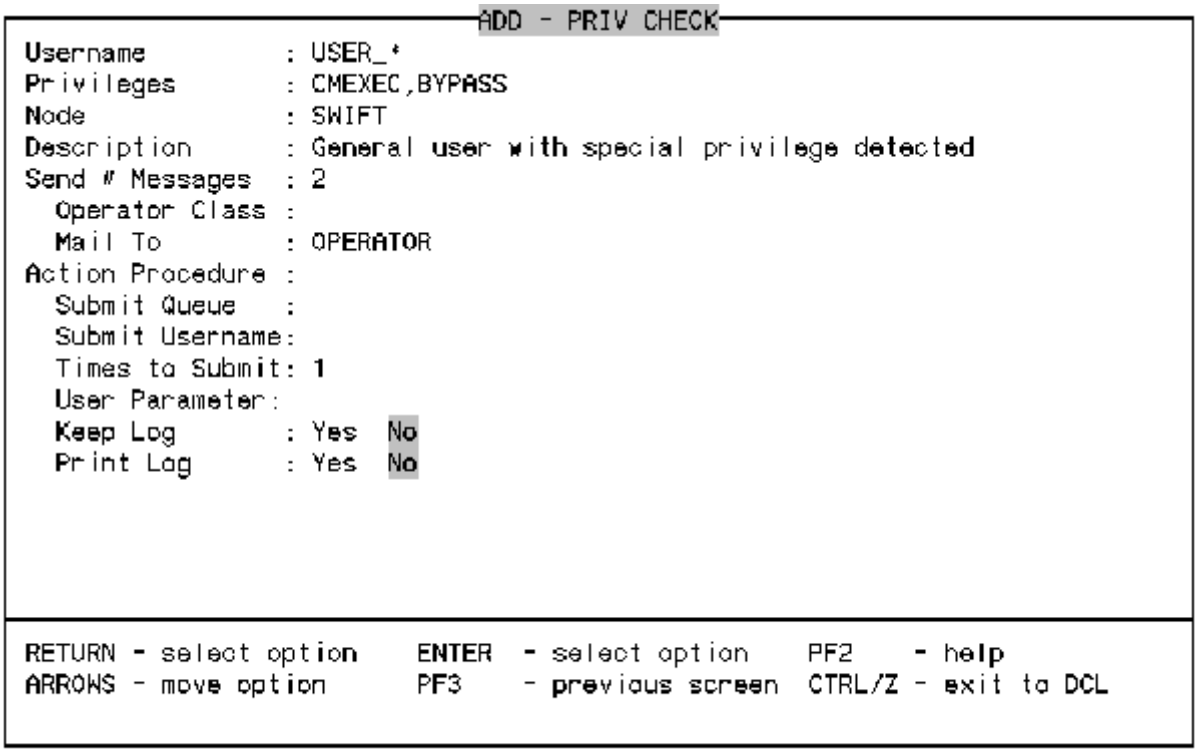
For example, to tell the operator and submit the batch job PURGE_DISK on the queue SWIFT$BATCH when SWIFT$DKA0: has less than 250,000 free blocks, type:

```
$ Hitman /event=(Type=check_free, Device=swift$dka0:, -
_$ Description="Dka0: has less then 250k free blocks", -
_$ Free=250000, Operator=security, Messages=1, Node=swift, -
_$ Submit=(Job=hitman_com:purge_disk.com, Keep, Noprint, -
_$ Queue=swift$batch, Times=1, P8="/keep=2"))
```

In addition to the event TYPE, you must specify a DEVICE and a FREE specification plus the actions you wish to take. You **cannot** use wildcards on the device. If you enter a logical device name, HITMAN will translate it to the equivalent physical device name. You can specify NODE if you wish a single node to check for and report the event. You can include a DESCRIPTION to remind yourself what the event was for, it also gets included in the mail message. If you enter a value between 1 and 100 for free blocks HITMAN will treat it as a percentage of disk's total space. In this case hitman_com:purge_disk.com can be as simple as:
$ purge/log 'p5'[*...]*.* 'p8' !p5 is the device and is supplied by Hitman

Checking for privileges

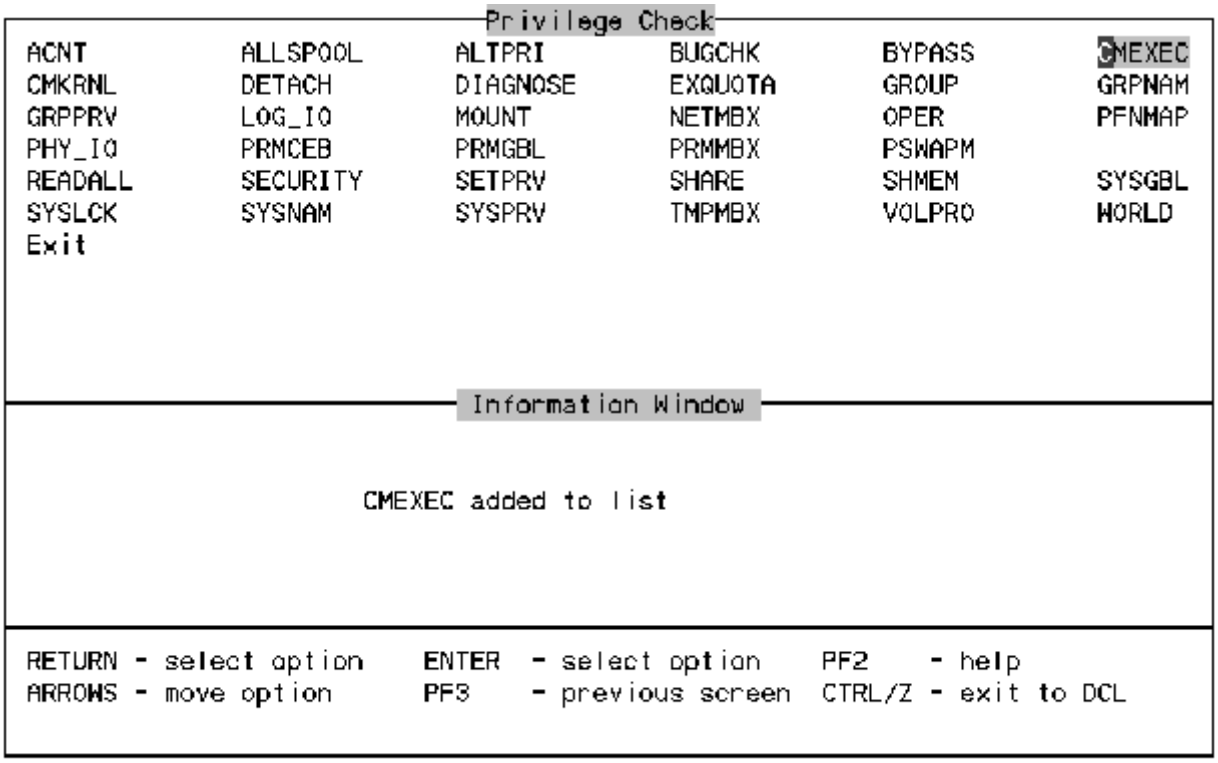
Use the PRIV_CHECK event to check for specific privileges for a specified user or for the whole system. You flag particular privileges and HITMAN lets you know when any process on your system has that privilege. The check privilege event actually has two windows, the main event window and a privilege window; examples of both follow.



This event will record in the log file when any of the general users, username begins user_, have either BYPASS or CMKRNL privileges. It will also send mail to the operator. When setting up this event through the menu a separate window will pop up allowing you to select each privilege you want in the check list, simply highlight a privilege using the cursor keys and then press the <space> or <enter> key; a message will be displayed that the privilege has been added to the list. Select exit when all the desired privileges have been selected and you can enter the rest of the event information. An example of this window is included at the top of the next page.

To enter this same event from the DCL level you could type:

```
$ HITMAN /EVENT=(TYPE=PRIV_CHECK, USER=USER_*, MESSAGES=2, -
_$ NODE=SWIFT, MAIL=OPERATOR, PRIVILEGES=(BYPASS, CMKRNL),-
_$ DESCRIPTION="General user with special privilege detected")
```

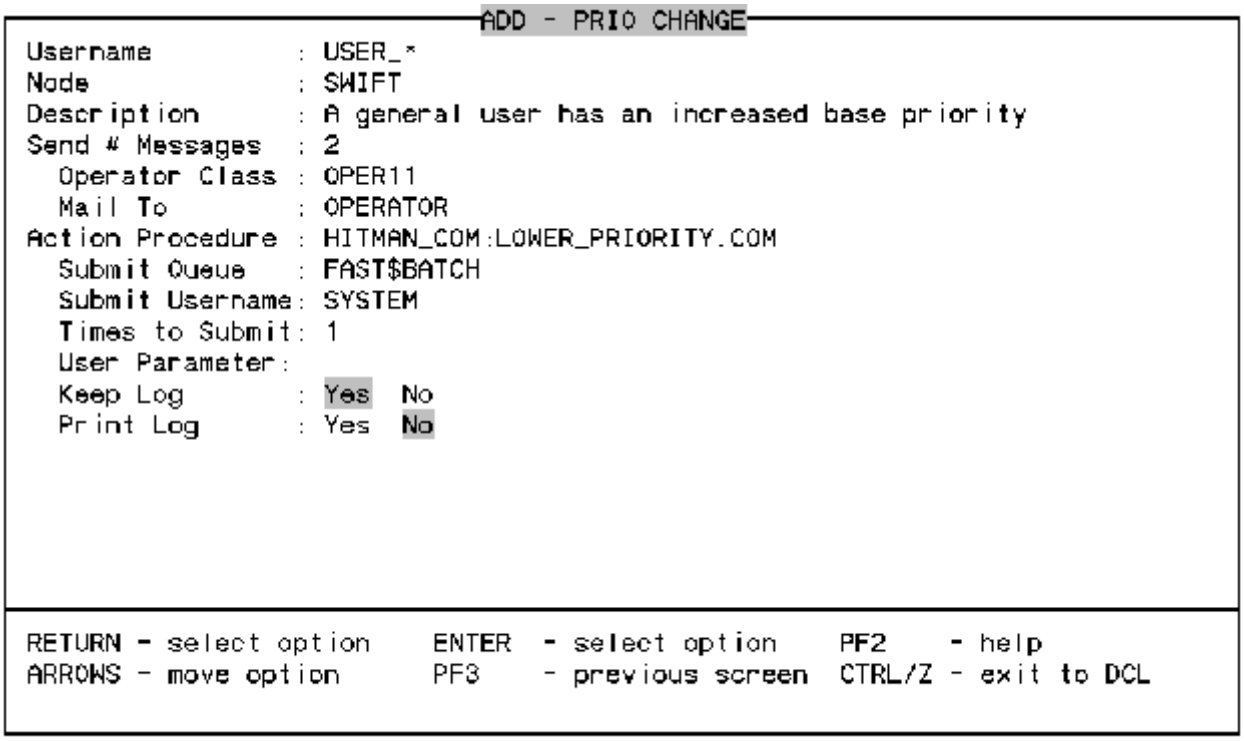


In addition to the event TYPE, you must specify a list of privileges plus the actions you wish to take. You can check for up to seven (7) privileges in each event. The names of the privileges are the same as OpenVMS uses. You can use wildcards in the username. You can specify NODE if you wish a single node to check for and report the event. You can include a DESCRIPTION to remind yourself what the event was for, it also gets included in the mail message sent.

NOTE: This event was intended to check for specific privileges. If you want to monitor and potentially terminate users with dangerous privileges (ALL category), use the /AUTHORIZE command. This event may not work if /NOSWAP is in effect since process privileges are part of the header that may be swapped out and be unavailable for HITMAN to check against.

Checking for priority changes

Use the PRIO_CHECK event to check for a user increasing their base priority using the "SET PROCESS/PRIORITY=" command. When anyone's priority increases, HITMAN tells you about it immediately. You can use a wildcard for the username.



For example, to send a message to operator class OPER11, mail the operator and submit a batch job to lower the users priority when any user with a username beginning user_ boosts their base priority, type:

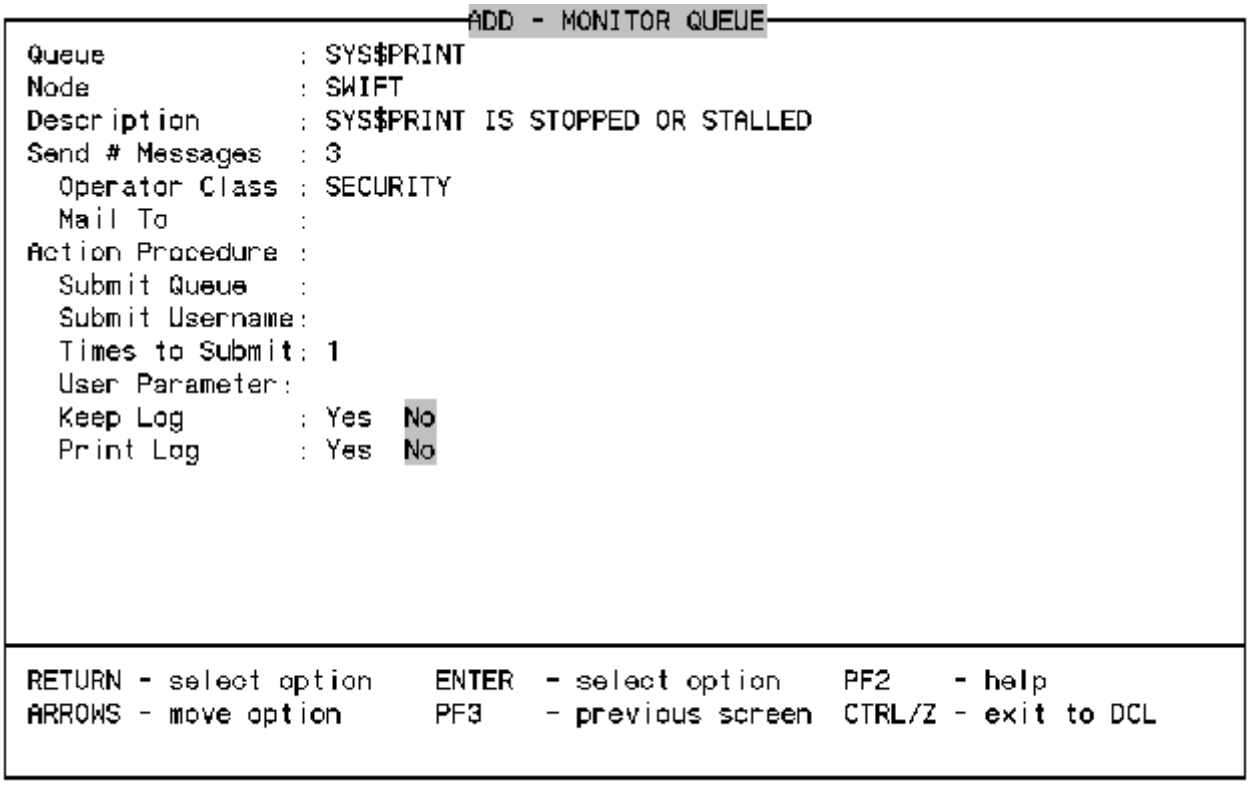
$ Hitman /Event=(Type=prio_change, User=user_*, Operator=oper11,-
_$ Node-swift, Messages=2, Mail=operator, Submit=(Username=system, Keep, -
_$ Noprint, Queue=fast$batch, Job=hitman_com:lower_priority.com),-
_$ Description="A general user has an increased base priority")

You must specify the event TYPE plus the actions you wish to take. You can use wildcards on username. You can specify NODE if you wish a single node to check for and report the event. You can include a DESCRIPTION to remind yourself what the event was for, it also gets included in the mail message. The PID of the detected process is passed to the batch job as P7 so that action can be taken; be sure to specify a batch queue that runs at a high priority.

Caution: If any priority change events are defined on your system the /NOSWAP qualifier will not work. Processes must be swapped in for HITMAN to determine priority.

Checking the status of queues

Use the MONITOR_QUEUE event to check if a queue is stopped or stalled. You can specify a wildcard for the queue name, however, choose a name that does not potentially represent a large number of queues since if more than 1 queue becomes stopped or stalled in a single data collection Hitman will only detect the 1st one.



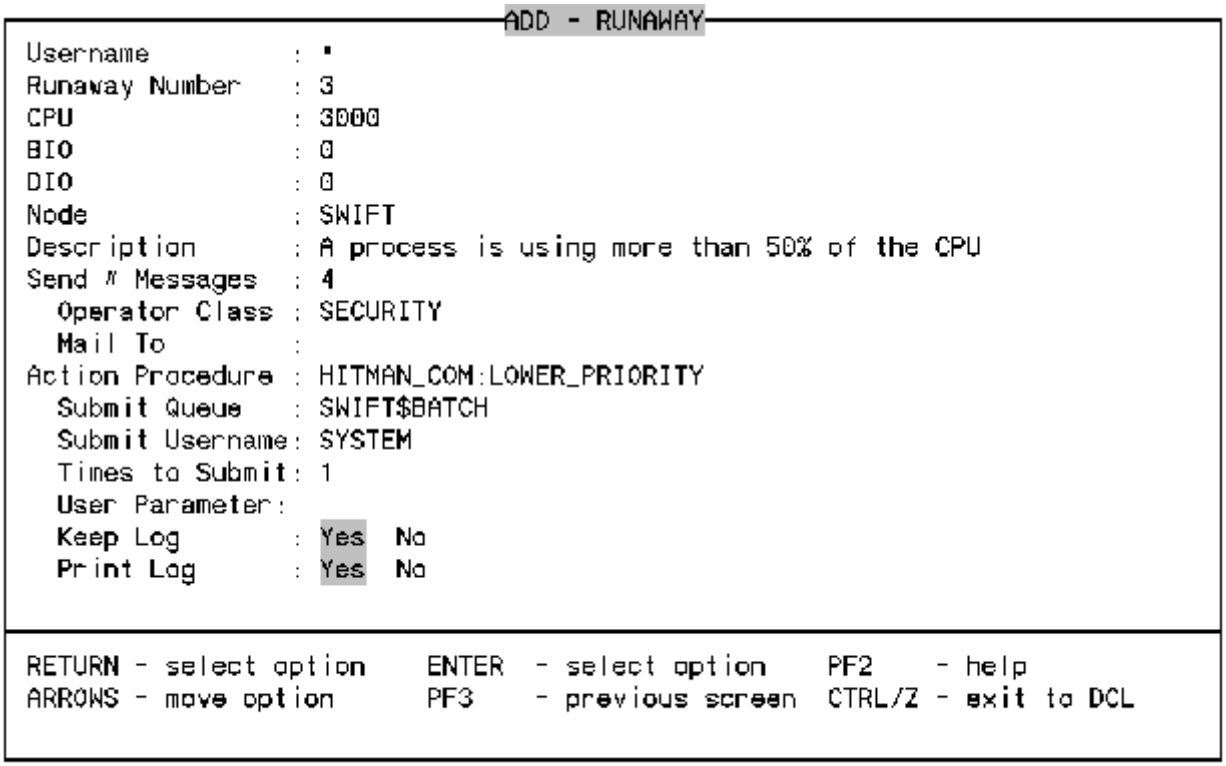
For example, to notify your security operator three (3) times when the queue SYS$PRINT is stopped or stalled, type:

```
$ HITMAN /EVENT=(TYPE=MONITOR_QUEUE, QUEUE=SYS$PRINT,-
_$ OPERATOR=security, MESSAGES=3)
```

In addition to the event TYPE, you must specify a QUEUE plus the actions you wish to take. You can specify NODE if you wish a single node to check for and report the event. You can include a DESCRIPTION to remind yourself what the event was for, it will also be included in the mail message if the event is configured to send mail. If you submit an action procedure for a MONITOR_QUEUE event the event type parameter (P1) will be set to either "QUEUE_STOPPED" or "QUEUE_STALLED".

Checking for runaway processes

Use the RUNAWAY event to check if a process is using excessive amounts of resources.



For example, to notify your operator four times when a process has used more than 50% of the total CPU resources for 3 data collection intervals. In this example we also submit a batch job, to a high priority queue, that lowers the priority to minimize the effect it is having on the system.

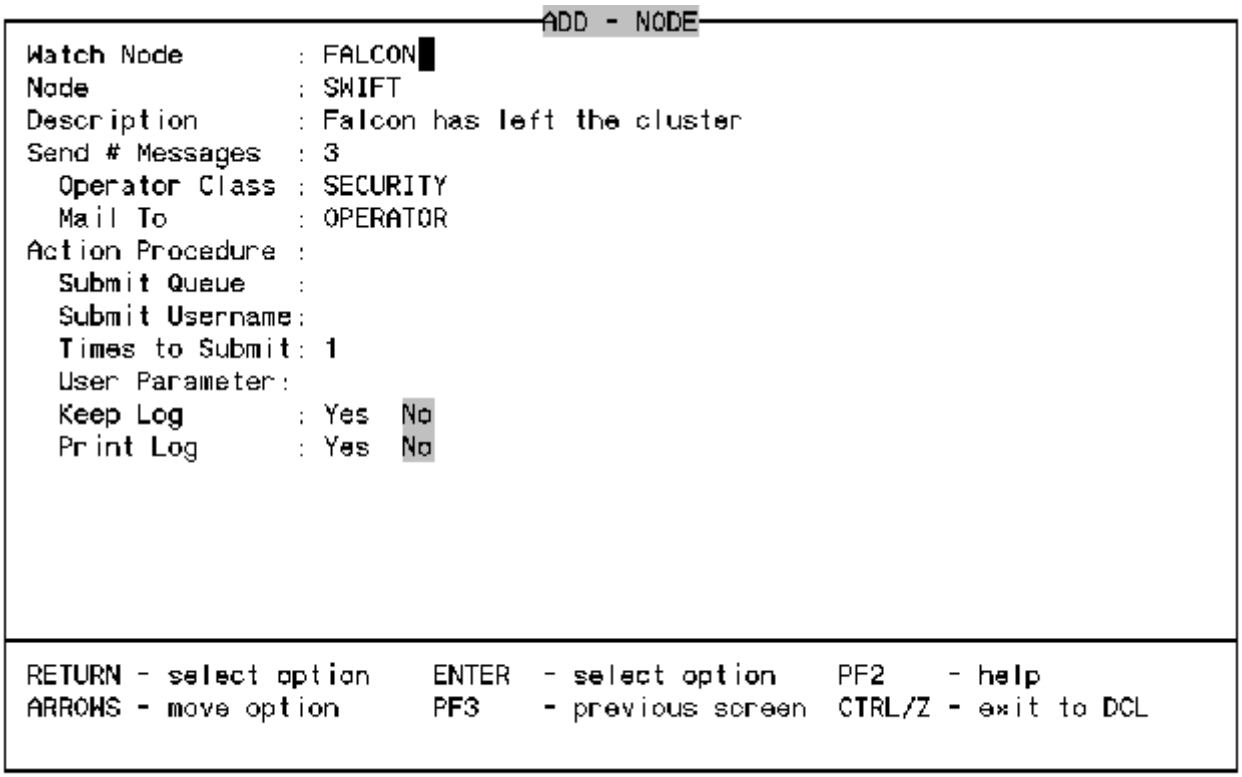
```
$ Hitman/event=(Type=runaway,runaway_number=3,cpu=3000,-
$_ Node=swift, Operator=security, Messages=4, Submit=(Keep,print, -
$_ Username=system, Queue=swift$batch,  Job=hitman_com:lower_priority))
```

In addition to the event TYPE, you must specify a runaway number and thresholds on CPU, BIO and/or DIO plus the actions you wish to take. You can specify NODE if you wish a single node to check for and report the event. You can include a DESCRIPTION to remind yourself what the event was for, it is also included in any mail messages sent. If you submit an action procedure for a RUNAWAY event the event type parameter P5 will be set to the PID.

Caution: If any runaway process events are defined on your system the /NOSWAP qualifier will not work. Processes must be swapped in for HITMAN to determine some of their resource usage characteristics. Hitman does NOT kill runaway processes, if you want this you must do it in the action procedure yourself.

Checking to see if a node has left the cluster - The Node Event

This event is used to monitor a node to take action anytime HITMAN detects that the node being monitored is no longer part of the cluster.



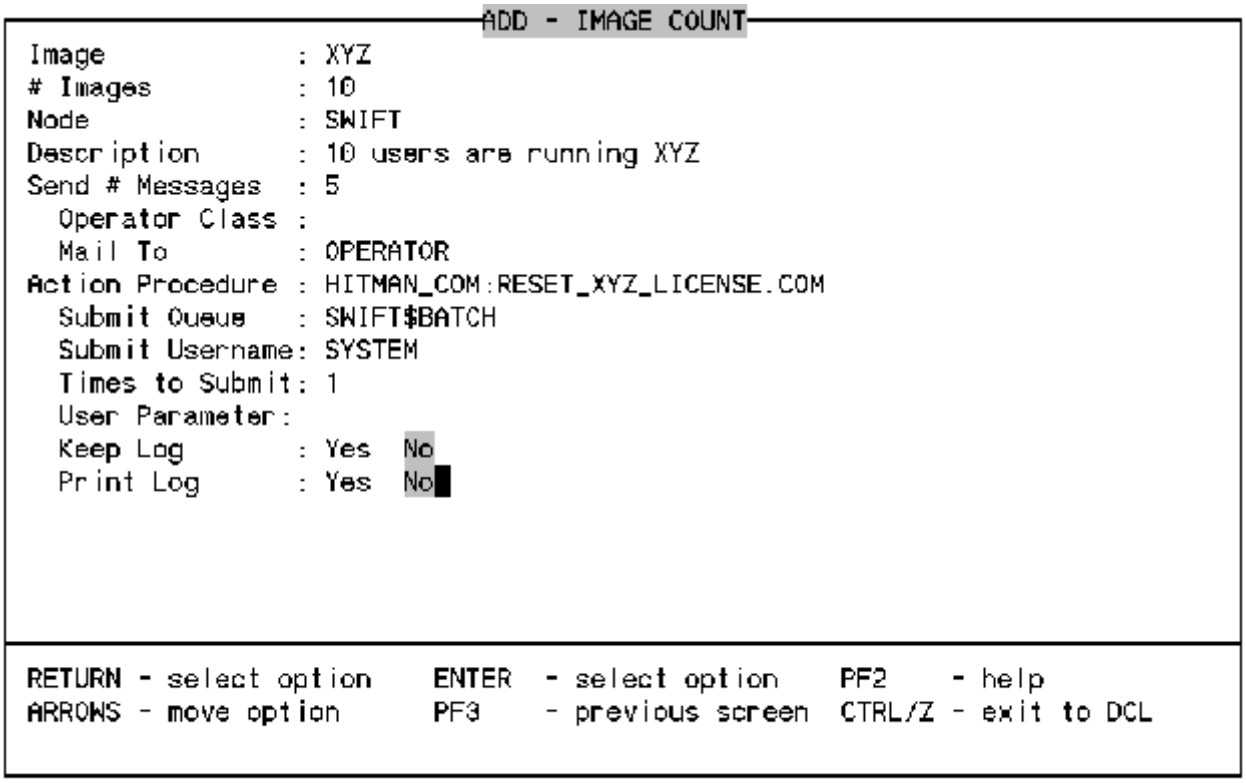
For example, to notify the operator class security and send mail to the system operator when node Falcon leaves the cluster, type:

$ Hitman /Event=(Type=node, Watch_node=falcon, Node=swift, -
$_ Operator=security, Mail=operator)

In addition to the event type you must specify a node to watch which cannot contain a wildcard. The node must be in the same cluster. To monitor nodes that are not part of the cluster refer to the absent and present process events. You can specify NODE if you wish a single node to check for and report the event. You can include a DESCRIPTION to remind yourself what the event was for, it is also included in any mail messages sent.

Checking for a number of users running the same image

Use the IMAGE COUNT event to take action whenever HITMAN sees a specified number of users, or more, running an image during a data collection pass.



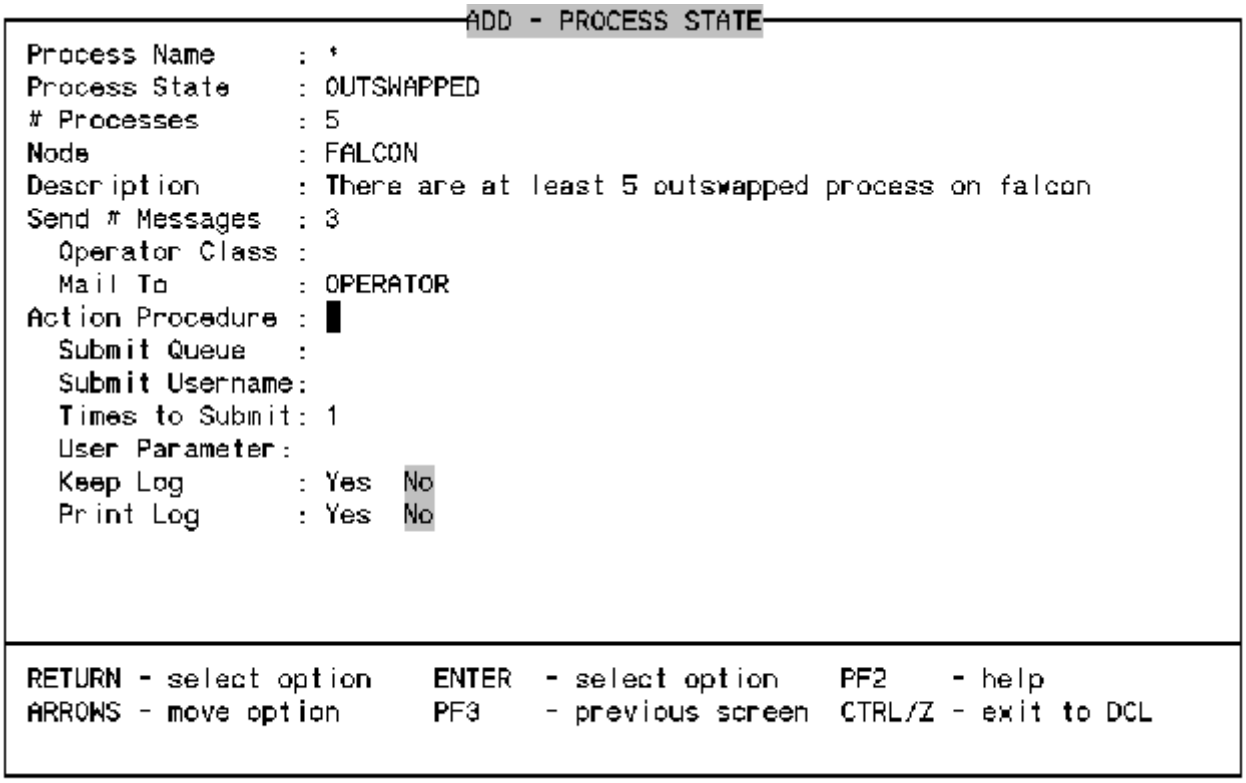
For example, program XYZ is installed on the system with a 15 user license. Anytime HITMAN detects 10 users, or more, running XYZ we want it to submit a batch job that resets the license counter to make sure it doesn't include any users that are no longer running the product; this can happen because it doesn't get reset automatically if users don't exit XYZ cleanly (for example if they use CTRL/Y to exit). A mail message is also sent to the system operator account. If the number of users in XYZ remains over 10 for more than 5 data collection intervals the event will stop sending mail messages, the reset license job is only run once.

```
$ Hitman/event=(Type=image_count, Node=swift, Messages=5, Limit=10, -
$_ Description="10 Users Are Running Xyz", Image=xyz, mail=operator, -
$_ submit=(job="hitman_com:reset_xyz_license.com", username=system,-
$_ queue=swift$batch))
```

In addition to the event type you must specify the image and the count (limit). You should specify NODE since most products are licensed for a specific node. You can include a DESCRIPTION to remind yourself what the event was for, it is also included in any mail messages sent.

Checking for processes in specific states

Use the process state event to a) take action anytime HITMAN detects a specific process in a specific state or b) take action anytime HITMAN detects a number of processes in the specified state.



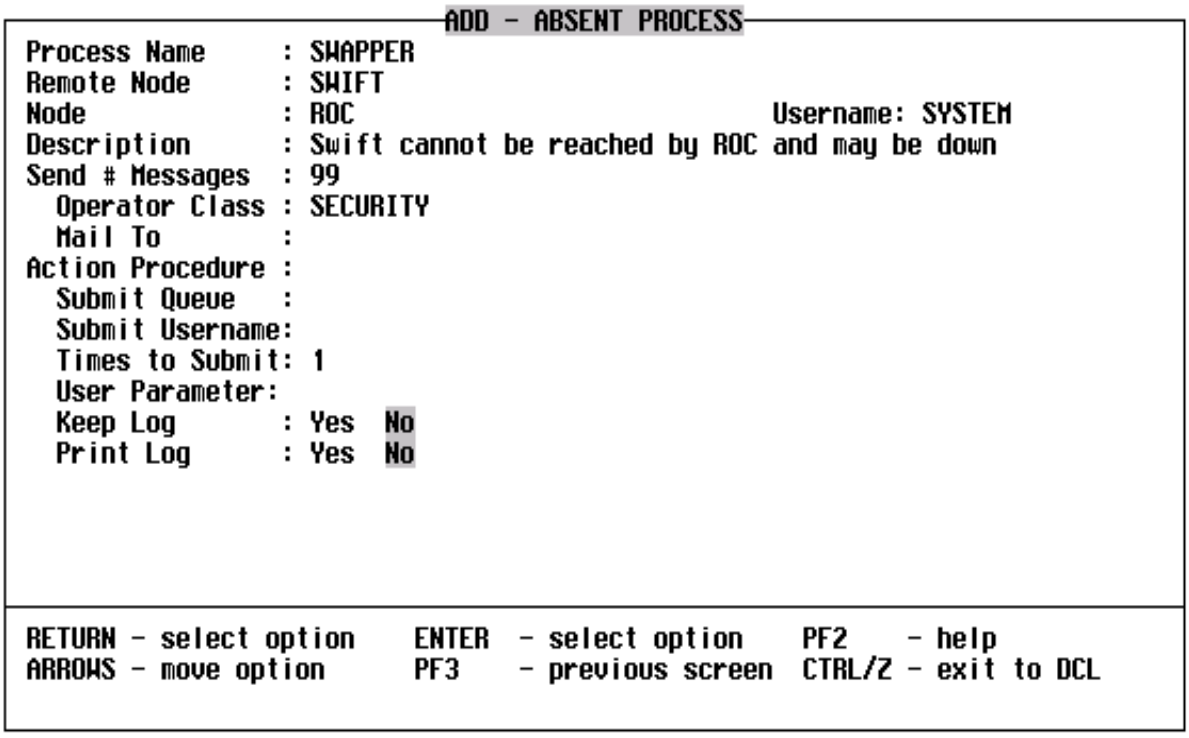
In this example mail is sent to the system operator when 5, or more, processes are in the outswapped process state. This could indicate a load problem. An event might also be set up to notify the operator if there are more than 2 or 3 processes computable at the same time. To set up this event at the DCL level:

```
$ Hitman /Event=(Type=process_state, Limit=5, Node=falcon,-
_$ Mail=operator, State=outswapped, Process=*,-
_$ Description”There are at least 5 outswapped process on falcon”)
```

In addition to the event type you must specify a process name, wildcards are acceptable, a process state and the process count. You can include a DESCRIPTION to remind yourself what the event was for, it is also included in any mail messages sent.

Checking for the absence of processes

Use the ABSENT_PROCESS event to check for a key process such are ERRFMT or OPCOM that should be on the system but is not. You can also use the absent process event to check for processes on other nodes that are not part of the cluster; if a process is chosen that is normally always present when the event is triggered it means that the remote node has gone down.
For example, to record in the log file when process SWAPPER is no longer on the remote node



SWIFT and to notify the operator class security every data collection until the process exists again, type:

```
$ Hitman/event=(Type=absent_process, Process="Swapper",-
$_ Description="Swift May Be Down", Remote_node=swift, Node=roc, -
$_ Messages=99, user=SYSTEM)
```

In addition to the event TYPE, you must specify a PROCESS plus the actions you wish to take. You can include a DESCRIPTION to remind yourself what the event was for, it is also included in any mail messages sent. It is recommended that you specify a NODE to check for and report the event; there is a small amount of overhead associated with logging in a checking for the process on the remote node, if multiple nodes are checking simultaneously this overhead could become significant. If you do not specify a remote node the process will be checked for on the current node and the event will behave exactly like the absent process event did in V8 and prior versions of

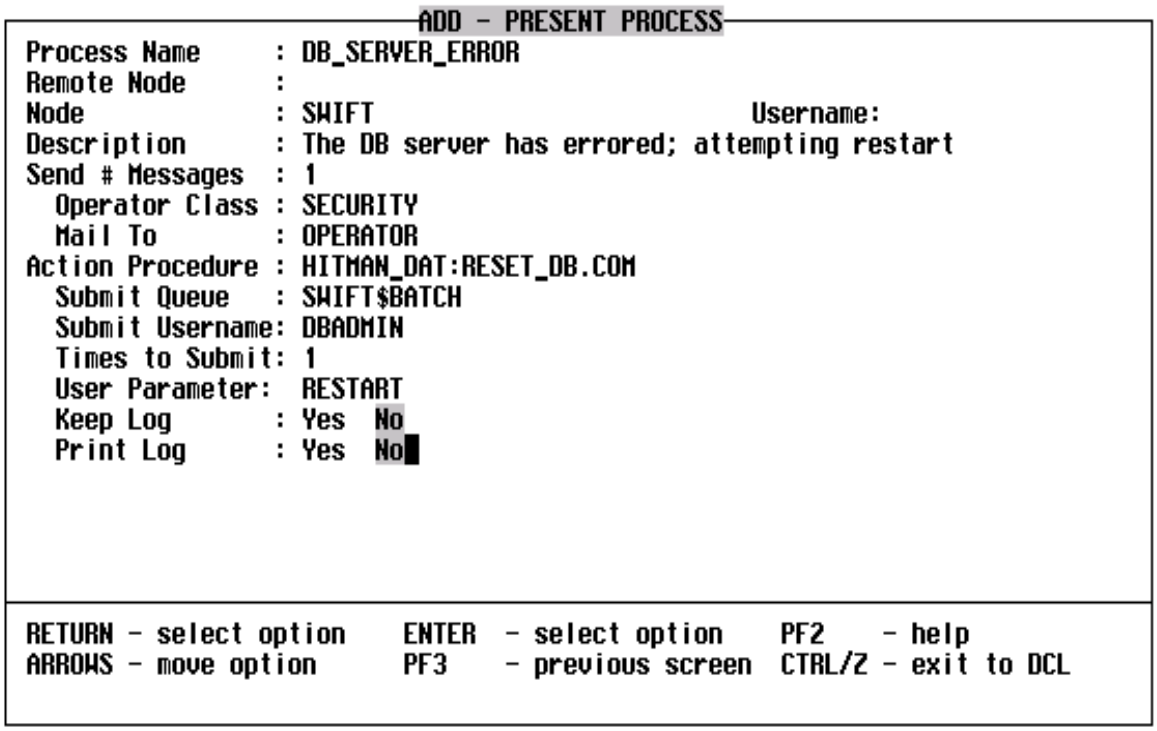
Hitman. Specifying 99 for the number of messages causes Hitman to signal this event every data collection interval until the process is once again seen on the remote node.

With Version 10 you may now also specify the username the process must be running under. If it is present but not running under that username the event will still be triggered. To have this event work as it did in prior versions of Hitman leave the username blank or enter an asterisk "*."

Caution: If any absent process events are defined on your system the /NOSWAP qualifier will not work. Processes must be swapped in for HITMAN to determine their process name. The remote node must be reachable via DECNET for this event to work on remote nodes. Some special setup is required on the remote node before this event will work correctly; refer to the section "Monitoring the status of remote nodes" on page 143.

Checking for the presence of processes

Use the PRESENT_PROCESS event to check for a process that should not be on the system but is. This event is the exact opposite of the ABSENT PROCESS event.
For example, to submit the batch procedure RDB_RESTART when process RDB_SERVER is no longer on the system, type:



In addition to specifying the event type you must specify the process name that, if seen on the system, Hitman should take action in response to. Since no remote node has been specified in this example Hitman will be checking for this process on the current node, if the current node is Falcon and the process is seen Hitman will send a console message to security, send mail to the system operator and submit the reset_db job. The user parameter has been set so that the reset_db job will be passed "RESTART" as the value for P8. The log file will be kept but not printed.

```
$ Hitman /Event=(Type=present_process, Process="DB_SERVER_ERROR",-
        _$ Messages=1, Node=SWIFT, Operator=security,
        _$ Mail=operator, Submit=(Job=hitman_dat:reset_db.com, -
        _$ Queue=swift$batch, Keep, Noprint, P8="Restart", Times=1),-
        _$ Description="The Db Server Has Errored, Attempting to Reset")
```

In addition to the event TYPE, you must specify a PROCESS plus the actions you wish to take. You can specify NODE if you wish a single node to check for and report the event. You can

include a DESCRIPTION to remind yourself what the event was for, it also gets included in any mail messages sent. If you specify a value greater than 1 for times to submit you tell HITMAN to submit the batch job for that many consecutive data collections IF the process being watched for still exists. Since no remote node was specified this event will only take action if the db_server_error process is seen on node Falcon.

With Version 10 you may now also specify the username the process must be running under before action is taken. If it is present but not running under that username the event will not be triggered. To have this event work as it did in prior versions of Hitman leave the username blank, as in this example, or enter an asterisk "*."

Caution: If any present process events are defined on your system the /NOSWAP qualifier will not work. Processes must be swapped in for HITMAN to determine their process name.

Using HITMAN events to automate routine operations

This section shows you how to use HITMAN events to automate your routine system operations.

In addition to the events described in the previous chapter, you can define events which are triggered by date and time as opposed to error conditions, you can specify events that are triggered every so many minutes and you can specify events that are triggered only when HITMAN starts up.

Use these "time", “startup” and “repeating” events to:

- automate your daily and weekly backups
- start new error and accounting logs on the first of each month
- locate file errors and lost files by running ANALYZE/DISK every night
- recover disk space by purging and deleting disk drives after backup has run
- regularily run license reset utilities

or any other system activity which occurs on specific dates or times, at system startup or needs to occur on a periodic basis throughout the day.

You can have procedures run:

- when HITMAN starts up
- every day
- on a specific date

- on the first day of every month	DAY=01
- on the last day of every month	DAY=END_OF_MONTH
- on one or more days of the week	DAY=(MONDAY, WEDNESDAY,-FRIDAY)
- on one or more days of the month	DAY=(1,07,15)
- on one or more months of the year	MONTH=(JANUARY, JUNE, -DECEMBER)

as well as specifying the hour and minute the procedure is to be submitted. Repeating events submit the procedure periodically (minutes=value); you tell HITMAN how many minutes between each run.

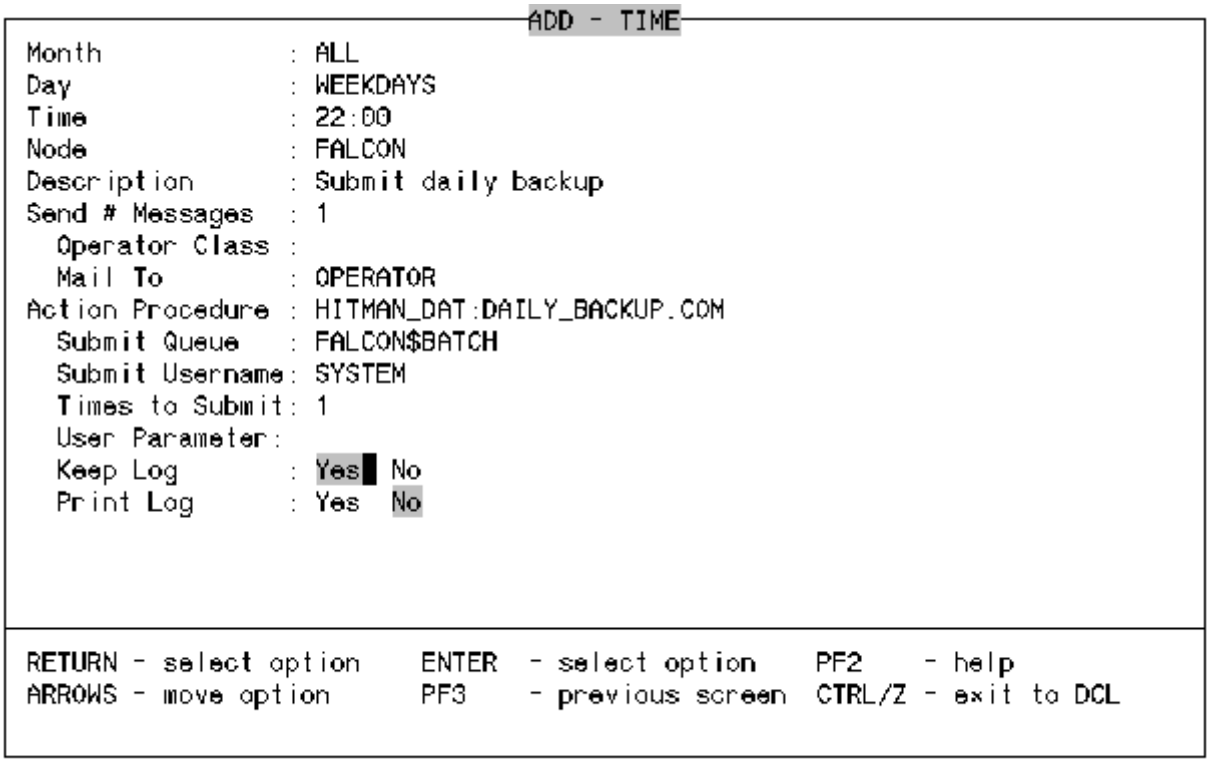
NOTE: If your system is down, HITMAN will **not** automatically resubmit jobs which should have been submitted while the system was down. Repeating events will be submitted when HITMAN starts and then again every specified number of minutes.

When you define a time event, ensure you put it in the file which will be active when the event occurs. An event in the prime file which occurs during non-prime time will not be submitted. If the job should be submitted during both times (for example, on weekdays and weekends), you

must define the event in **both** files.

By default, HITMAN will submit a time event every day just after midnight. You can alter this by specifying days, dates and times when you define the event. To have a job submitted more than once per day, you must define a separate event for each separate time the job is to be submitted or use the repeating event to have it submitted on a regular basis.

Menu interface for adding time events:

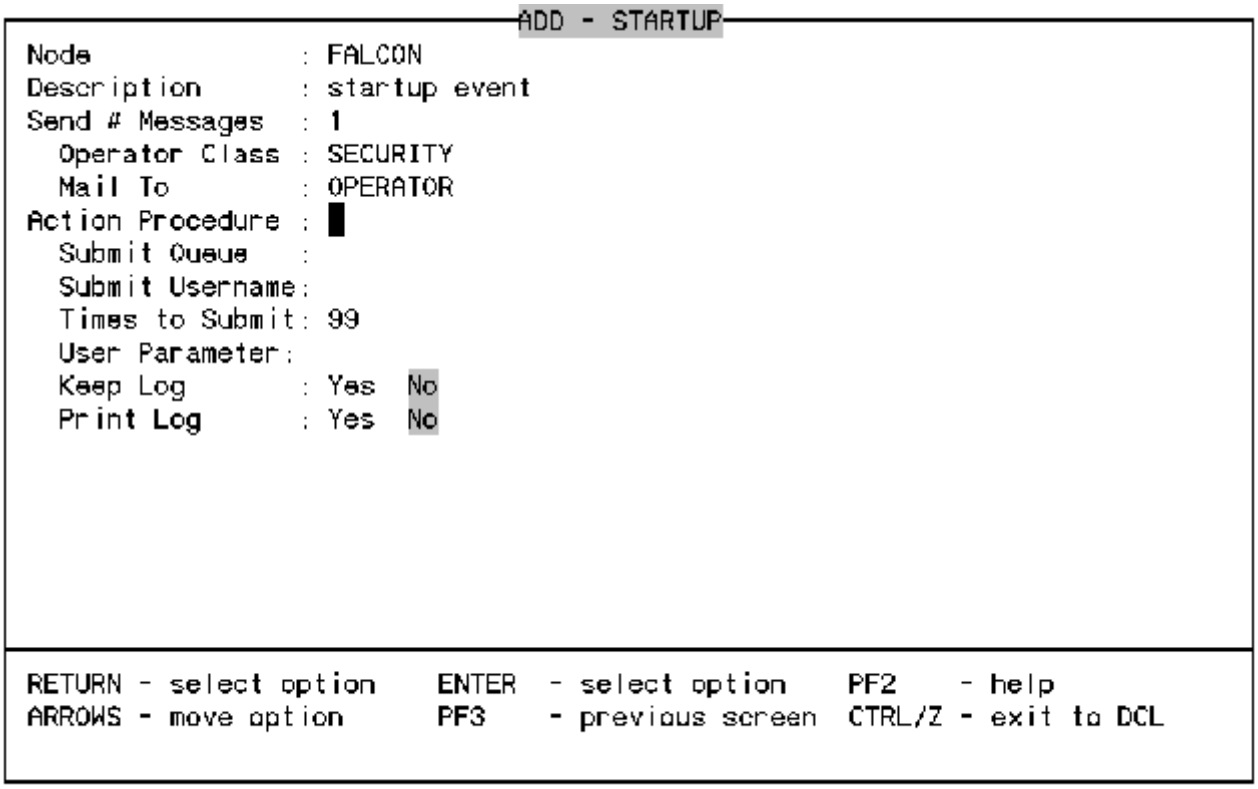


This event will submit the daily backup job, under username system, every weekday at 10:00 pm. The log is kept but not printed. A mail message is sent to the system operator.

```
$ hitman /event=(type=time, month=all, day=weekdays, time=22:00, node=falcon,-
_$ description="Submit daily backup", messages=1, mail=operator, -
_$ submit=(job=hitman_dat:daily_backup.com, queue=falcon$batch, times=1, -
_$ keep, noprint, username=system)) /file=nonprime
```

In addition to specify the event type you should specify the month, day and time. Since these events are normally used to submit action procedures be sure to specify the procedure that shoul be submitted. 22:00 is in our nonprime time so the qualifier /file=nonprime was used to guarantee that this event was added to that file; had it been added to our prime file the nightly backup would never get submitted.

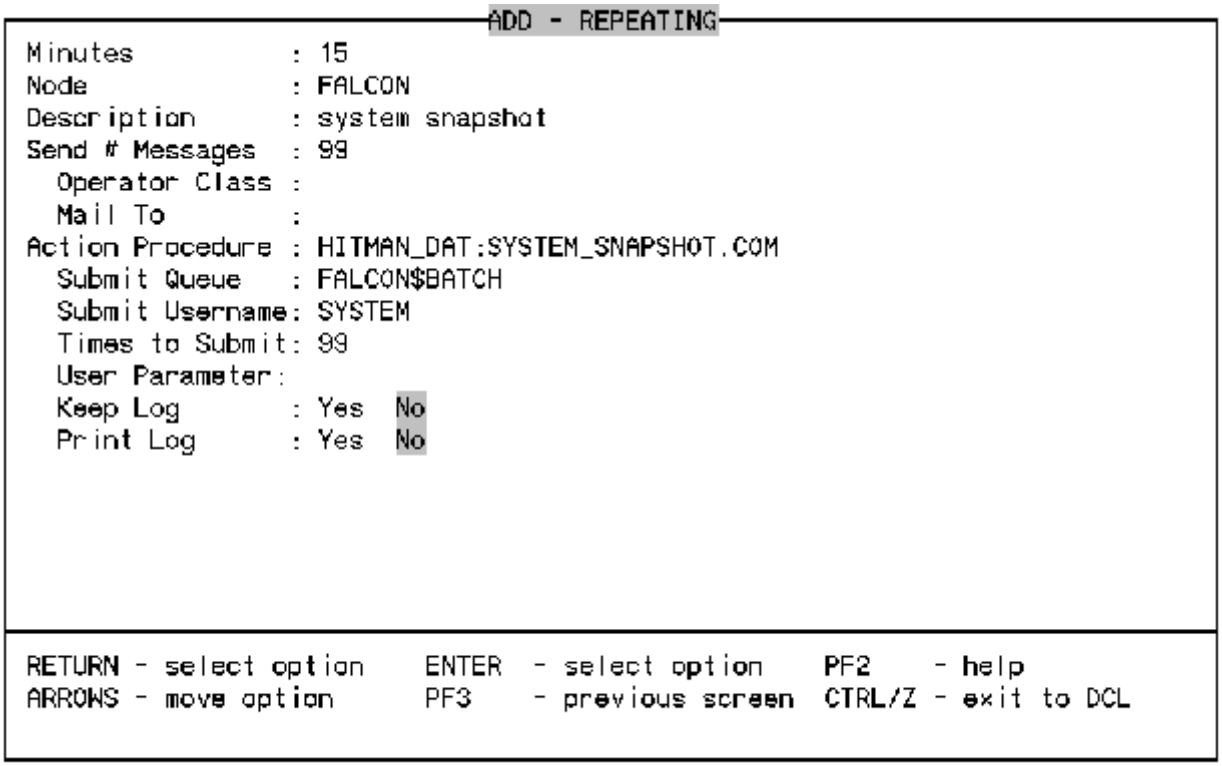
Menu Interface for adding startup events:
With Hitman's startup event you can submit batch jobs anytime Hitman is started. At many sites this will be only when the system is restarted and this event is effectively a system startup event.



In addition to specifying the event type you must specify whatever actions you wish to be taken. In this case everytime Hitman is started on node Falcon a console message is sent to operator class security and a message is mailed to the system operator; this mail message will include the events' text description.

```
$ Hitman /Event=(Type=startup, Messages=1, Node=falcon, Mail=operator,-
_$ Operator=security, Description="Startup Event")
```

Menu interface for adding repeating events
With HITMAN's repeating event you can have HITMAN submit a job every few minutes. This event was added with V8 and has proved to be a popular addition to Hitman.



For example, to run a command procedure that takes a snapshot of certain system resources every 15 minutes:

```
$ Hitman/event=(Type=repeating, Node=falcon, Messages=99, -
_$ Minutes=15, Description="System Snapshot", Submit=(Times=99,-
_$ Job="Hitman_dat:system_snapshot.com", Queue=falcon$batch,-
_$ Username=system, nokeep, noprint))
```

In addition to the event type you must specify an action procedure for the event to submit. Since this job will be run regularily, in this case every 15 minutes, it is recommended that log files not be kept after you are satisfied that the job is running correctly. It is also recommended that you do not send console or mail messages since these would be sent everytime the action procedure was submitted.

NOTE: If a value less than 99 (repeat continuously) is entered for messages or times to submit the repeating job will only be submitted that many times after Hitman is started.

Here are some additional examples of TIME and STARTUP events:

```
$ HITMAN/EVENT=(TYPE=TIME,-
_$ DAY=(WEEKDAYS),TIME=23:30, DESCRIPTION="BACKUP",-
_$ SUBMIT=(JOB=DAILY_BACKUP,QUEUE=FAST_BATCH))
```

submits the batch procedure DAILY_BACKUP on Monday - Friday at 11:30 p.m.

```
$ HITMAN/EVENT=(TYPE=TIME,-
_$ DAY=01,TIME=00:01,-
_$ DESCRIPTION="Reset the monthly OpenVMS accounting file",-
_$ SUBMIT=(JOB=RESET_ACCOUNTING,QUEUE=FAST_BATCH))
```

submits the batch procedure RESET_ACCOUNTING on the first day of every month at 1 minute after midnight.

```
$ HITMAN/EVENT=(TYPE=TIME,-
_$ DAY=(15,END_OF_MONTH),MONTH=(MARCH,JUNE,DECEMBER),-
_$ DESCRIPTION="Check for disk errors on VENUS",NODE=VENUS,-
_$ SUBMIT=(JOB=CHECK_DISK,QUEUE=FAST_BATCH))
```

submits the batch procedure CHECK_DISK on the fifteenth and last day of the three months listed at the default time of "00:00".

```
$ HITMAN/EVENT=(TYPE=TIME,-
_$ DESCRIPTION="Daily Security Audit JMD-09/06",-
_$ SUBMIT=(JOB=DAILY_AUDIT,QUEUE=FAST_BATCH))
```

submits the batch procedure DAILY_AUDIT every day at the default time of "00:00".

```
$ HITMAN/EVENT=(TYPE=STARTUP,-
_$ DESCRIPTION="HITMAN Startup",-
_$ SUBMIT=(JOB=HIT_START,QUEUE=FAST_BATCH))
```

submits the batch procedure HIT_START every time HITMAN starts up. The event is only triggered when HITMAN starts up. No other date, time or error can trigger it.

Specifying months

Use the MONTH qualifier to specify months. The syntax is:

MONTH=(list_of_months)

If you specify a list of months separated by commas, you must enclose the list in parentheses "()". If you specify a single month, you can omit the parentheses.

If you omit the MONTH qualifier, HITMAN will submit the procedure every month.

Specifying days
Use the DAY qualifier to specify months. The syntax is:

DAY=(list_of_days)

If you specify a list of days separated by commas, you must enclose the list in parentheses "()". If you specify a single day, you can omit the parentheses.

You can specify individual days of the week. You can use the keyword "WEEKEND" to indicate Saturday and Sunday and the keyword "WEEKDAY" to indicate Monday through Friday.

You can use the numbers 1 - 31 to indicate a particular day of the month and the keyword "END_OF_MONTH" to indicate the last day of the month. You can enter a leading zero ("01" as opposed to "1") on the days of the month which only have a single digit.

If you omit the DAY qualifier, HITMAN will submit the procedure every day.

Specifying times
Use the TIME qualifier to specify times. The syntax is:

TIME=time

If you omit the time qualifier, HITMAN uses "00:00"

Additional actions
Although the examples have emphasized submitting batch jobs, you can specify any event actions such as sending operator messages or mailing a message to a list of users.

Additional qualifiers
Although the examples have emphasized TIME specific qualifiers such as DAY, MONTH and TIME, you can specify other qualifiers such as NODE, MAIL, etc.. TIME and STARTUP events are unusual in the way they are triggered but they are still events just like CHECK_FREE and ABSENT_BATCH.

Maintaining HITMAN Events (DCL level)

The HITMAN /EVENT qualifier is used to add, modify or delete events from the DCL level. The following pages give a complete reference for this qualifier.

DCL Command Format

To **add** events, use the syntax:

$ HITMAN/EVENT=(TYPE=event_type,condition_qualifiers,action_qualifiers)

To **modify** a field in a specified event, use the syntax:

$ HITMAN/EVENT=(ID=event_id,field=value)

To **modify** a field in one or more events, use the syntax:

$ HITMAN/EVENT=(FIELD=field_name,OLD=value,NEW=value)

To **delete** an event, use the syntax:

$ HITMAN/NOEVENT=(ID=event_id)

Use the /EVENT qualifier to check your system for device errors, low disk free space, missing batch jobs and priority changes. HITMAN can monitor any number of events. When you add an event, HITMAN assigns it a unique number called an event_id. See the chapter "Using HITMAN events" for more details on this facility. Events can be added, deleted or modified from either the command line or the menu/form interface. You can change a field in several events with a single command.

Here are a few examples of events:

/EVENT=(TYPE=ERROR,DEVICE=DUA0:,OPERATOR)

Send a message to the operator when an error occurs on disk DUA0:

/EVENT=(TYPE=USER,USER=FRED,IN,MAIL=SYSTEM)

Mail a message to user SYSTEM when user FRED logs on to the system

/EVENT=(TYPE=PRIV_CHECK,USER=*,NODE=ATHENA,PRIV=(BYPASS))

Write a record in the log file when any process on node ATHENA has the privilege BYPASS.

/EVENT=(TYPE=CHECK_FREE,FREE=500,DEVICE=DUA0:,SUBMIT=(JOB=PRG))

Submit the batch procedure PRG when drive DUA0 has less than 500 blocks free.

/EVENT=(TYPE=ABSENT_BATCH,QUEUE=OPS$BAT,JOB=BCKP,OPERATOR)

Send a message to the operator when the daily backup job is not in the queue.

NOTE: HITMAN also fills in any necessary values which you don't specify with the values in the DEFAULT event.

Event qualifiers are divided into three classes:

- event_type
- condition_qualifiers
- action_qualifiers

The EVENT=(/TYPE=event_type) qualifier specifies what type of event you are defining. It must always be present when you define an event. Event types are defined on the following page.

The condition qualifiers specify additional conditions beyond the type of event. Each qualifier is optional or mandatory depending on the type of event. Here is a list of the condition qualifiers:

BIO	define buffered I/O threshold for runaway processes
CPU	define the CPU usage threshold for runaway processes
DAY	define TYPE=TIME events
DESCRIPTION	record a descriptive information about the event
DEVICE	lists devices to check for error and low space
DIO	define buffered I/O threshold for runaway processes
FIELD	specify a field to be modified
FREE	define free space thresholds
IMAGE	list images to check for
IN	specify that the event triggers when the user is present
JOB	define job names to monitor
LIMIT	number of processes that must be in a particular state or image
MESSAGES	number of times the event should be signalled, controls how many console messages and mail messages are sent
MINUTES	sets how often repeating events should be processed
MONTH	define TYPE=TIME events
NEW	define a new value for a field in a mass change
OLD	define a existing value for a field in a mass change
OUT	specify that the event triggers when the user is not present
PRIVILEGE	list privileges to check for
PROCESS	list process names to check for
REMOTE_NODE	specifies the remote node that should be checked for the absent or present process event
RUNAWAY_NUMBER	define the number of data collections
QUEUE	list queues to monitor
STATE	lists process state to check for
TERMINAL	list terminals to monitor
TIME	define TYPE=TIME events
USER	list users to monitor

	Version 10 adds this as an additional check for absent/present process/batch events
WATCH_NODE	the node that is being watch for cluster membership

The action qualifiers specify what is to be done when the event is true and which node is to take the action. Any action can be specified for any event. If no action is specified, HITMAN will simply write a record in the log file. Here is a list of the action qualifiers

MAIL	specify the user that mail should be sent to
MESSAGES	specify the number of times to notify people
NODE	specify which node is to watch for and report the event
OPERATOR	specify that an operator message is to be sent
SUBMIT	specifies the details of the action procedure to be submitted

TYPE=event_type

Use the TYPE qualifier to specify the type of event to monitor. This qualifier must be present to add an event. Valid event types are:

ABSENT_BATCH	Check if batch job is absent
ABSENT_PROCESS	Check for missing process
CHECK_FREE	Check for minimum number of free blocks
ERROR	Check for device errors
IMAGE	Check image activities
IMAGE_COUNT	Check for many processes running 1 image
MONITOR_QUEUE	Check status of a queue
NODE	Check the status of a different node
PRESENT_BATCH	Check if batch job is present
PRESENT_PROCESS	Check for process on system
PRIO_CHANGE	Check for priority boost
PRIV_CHECK	Check for specific privileges
REPEATING	Take action every x minutes
PROCESS STATE	Check for processes in a specific state
RUNAWAY	Check for runaway processes
STARTUP	Trigger on HITMAN startup
TERMINAL	Check terminal activities
TIME	Trigger on date or time
USER	Check user activities

The following qualifiers are used depending on the event:

BIO=threshold

Use the BIO qualifier in RUNAWAY events to specify the number of buffered I/Os in a data collection interval, above which a process is considered to be a "runaway".

CPU=threshold

Use the CPU qualifier in RUNAWAY events to specify the number of CPU ticks in a data collection interval, above which a process is considered to be a "runaway". This value is not a percentage. There are 6,000 ticks in one minute. To find processes using 25% of your CPU, use the formula:

.25 x 6000 x data_collection_interval

DAY=(list_of_days)

Use the DAY qualifier to specify days of the week or days of the month in TIME events. This qualifier can only be used with TIME events. Use one of the following entries:

DAY=ALL (Default)
DAY=MONDAY
DAY=TUESDAY
DAY=WEDNESDAY
DAY=THURSDAY
DAY=FRIDAY
DAY=SATURDAY
DAY=SUNDAY
DAY=WEEKDAYS
DAY=WEEKEND
DAY=END_OF_MONTH (the last day of any month)
or
DAY = 01 - 31 to designate a day of the month

DESCRIPTION="text"

Use the DESCRIPTION qualifier to record information to allow you to remember why you created the event or to pass information to a batch procedure triggered by the event occurrence. This qualifier can be used with any event type. It is displayed in the modify/delete events menu. HITMAN does not make any use of the field, it is there solely for your reference. The maximum field size is 50 characters.

DEVICE=device_name

Use the DEVICE qualifier to specify a device in CHECK_FREE and ERROR events. Wildcards are **not** allowed. Version 10 of Hitman adds two special keywords for error events, CPU: and MEMORY:, to watch for errors on these system devices.

DIO=threshold

Use the DIO qualifier in RUNAWAY events to specify the number of direct I/Os in a data collection interval, above which a process is considered to be a "runaway".

field=value
Use this qualifier in conjunction with the ID qualifier to modify a specific field in an existing event. Fields have the same names as the qualifiers in this list.

FIELD=field_name
Use the FIELD qualifier in conjunction with the NEW and OLD qualifiers to change a field in one or more existing events.

FREE=n
Use the FREE qualifier to specify the minimum number of free blocks in CHECK_FREE events. The minimum allowed is 1. There is no maximum. If you specify 100 or more, HITMAN assumes the number is expressed in blocks. If you enter 99 or less, HITMAN assumes the number is a percentage of the total blocks available on the disk drive.

IMAGE=image_name
Use the IMAGE qualifier to specify an image name in USER, TERMINAL, IMAGE and IMAGE COUNT events. Wildcards are allowed.

IN
Use the IN qualifier in USER events to specify that you want to be notified when the user is present on the system.

JOB=job_name
Use the JOB qualifier to specify batch job names in ABSENT_BATCH and PRESENT_BATCH events. The job name is case sensitive and must match the case of the actual job..

LIMIT=process_count
Use LIMIT help control how process state events are signalled. If LIMIT is set to one then each process that HITMAN sees in the specified state will be signalled, mail sent and an action procedure submitted. If LIMIT is set greater than one then the event is only triggered once any time HITMAN sees that many, or more, processes in the specified state.

MAIL=username
Use the MAIL qualifier to specify that HITMAN is to mail a message when the event occurs. This qualifier can be used with any event type. Use it to specify the username that is to receive mail. If you specify the MAIL qualifier, you must enter a username. Wildcards are not allowed. You can specify a mail distribution list using the syntax MAIL="@distribution_list_name". Version 10 adds support for mail addresses that are not OpenVMS usernames for systems running extra mail packages such as SMTP mail. To enter a mail address that does not conform to the OpenVMS username status precede the address with a colon (:) and it will be treated as literal and no error checking done on that particular e-mail address.

MESSAGES=n
Use the MESSAGES qualifier to specify how many times you are to be notified when the event occurs. You can tell HITMAN to signal the event from 1 to 98 times. If you specify 99 notifications, HITMAN will signal the event until it is no longer true. The default number of notifications is 3. This qualifier can be used with any event type.

MINUTES=number_of_minutes
Use MINUTES to specify how often a REPEATING event should be performed. A repeating event will send a console message, send mail and/or submit an action procedure every number_of_minutes minutes.

MONTH=(list_of_months)
Use the MONTH qualifier to specify one or more months in TIME events. This qualifier can only be used with TIME events. Specify months as text, using the full name or ALL if you want the event to run every month.

NODE=node_name
Use the NODE qualifier to specify which node in a cluster is to check for and report the event. This qualifier can be used with any event type. By default, every node will check for and report the event. Wildcards are not allowed.

NEW=value
Use the NEW qualifier in conjunction with the FIELD and OLD qualifiers to specify the new value for a field in one or more existing events.

OLD=value
Use the OLD qualifier in conjunction with the FIELD qualifier to select one or more existing events whose values are to be updated using the NEW qualifier.

OPERATOR=operator_class
Use the OPERATOR qualifier to specify that HITMAN is to send a message to the specified operator class. Wildcards are not allowed. This qualifier can be used with any event type. If you specify an operator class, the message will be sent to that class. Otherwise operator messages will be sent to the class specified by HITMAN's /OPERATOR_CLASS qualifier.

OUT
Use the OUT qualifier in USER events to specify that you want to be notified when the user is not on the system.

PRIVILEGE=(p,...,p)
Use the PRIVILEGE qualifier to specify privileges to check for. You can specify a maximum of seven (7) privileges. Wildcards are not allowed. Specify privileges using the standard OpenVMS text abbreviations for the privilege (i.e. CMKRNL, SYSPRV, BYPASS, etc.)

PROCESS="process_name"

Use the PROCESS qualifier to specify a process name in ABSENT_PROCESS and PRESENT_PROCESS events. Wildcards are allowed. No case conversion is done on process names so be sure to specify them exactly as they appear on the system.

QUEUE=queue_name

Use the QUEUE qualifier to specify a queue in MONITOR_QUEUE, ABSENT_BATCH and PRESENT_BATCH events. Wildcards are allowed. We recommend, however, grouping queues into small groups with wildcards because once one queue in the group is stopped or stalled, no other queue will be detected until HITMAN has seen all queues that match the group in a normal state.

REMOTE_NODE=node

Use the REMOTE_NODE qualifier to specify a remote node that should be checked for the absent or present process condition. The specified node does not need to be part of the current cluster but must be reachable by DECNET. If the remote_node is not specified the absent or present process event will check the current node.

RUNAWAY_NUMBER=number_of_data_collections

Use the RUNAWAY_NUMBER qualifier in RUNAWAY events to specify the number of data collection intervals over which the BIO, CPU and DIO thresholds apply.

STATE=process_state

Use the STATE qualifier in STATE events to specify which process state is to be watched for. Valid processor states are:

CEF
COM
COMO
CUR
COLPG
FPG
HIB
HIBO
LEF
LEFO
MWAIT
OUTSWAPPED
PFW
SUSP
SUSPO

SUBMIT=(JOB=j,QUEUE=q,USER=u,KEEP,PRINT)

Use the SUBMIT qualifier to specify that HITMAN is to submit a procedure when the event

occurs. This qualifier can be used with any event type.

JOB=job_name
Use the JOB qualifier to specify the procedure HITMAN is to submit. If you specify the SUBMIT qualifier, you must enter this qualifier. Wildcards are not allowed. HITMAN passes information to the procedure via parameters.

QUEUE=queue_name
Use the QUEUE qualifier to specify the queue on which HITMAN is to submit the procedure. If you do not specify this qualifier, HITMAN will use SYS$BATCH. Wildcards are not allowed.

USER=username
Use the USER qualifier to specify the account under which HITMAN is to submit the procedure. If you do not specify this qualifier, HITMAN will use the account it was started under. Wildcards are not allowed.

KEEP
Use the KEEP qualifier to specify that the procedure log is to be kept after the job runs. If you do not specify this qualifier, the batch queue will delete the log.

P8=user_parameter
Use P8 to pass a site specific parameter into the action procedure. Whatever text is specified will be available to the action procedure as P8. If the text includes spaces or special characters be sure to enclose it in double quotes (“).

PRINT
Use the PRINT qualifier to specify that the procedure log is to be printed after the job runs. If you do not specify this qualifier, the log will not be printed.

Specifying the qualifiers NOKEEP,NOPRINT is equivalent to specifying the SUBMIT/NOLOG.

TIMES=number
Use TIMES to specify how many data collections in a row an action procedure should be submitted if the event remains true. The default is 1. Specify 99 if you want an action procedure submitted every data collection interval until the event is no longer true. Use this parameter carefully since submitting multiple copies of a batch job can have performance implications if one copy is not finished when the next one starts.

TERMINAL=terminal

Use the TERMINAL qualifier to specify a terminal in USER and TERMINAL events. Wildcards are allowed.

TIME=time

Use the TIME qualifier to specify an hour and minute to submit procedures in TIME events. This qualifier can only be used with TIME events.

USER=username

Use the USER qualifier to specify a username in USER events. Wildcards are allowed. For example, USER=AB* checks all usernames beginning with the characters 'AB'. This qualifier is required in the USER event and optional in the IMAGE, TERMINAL, PRIO_CHANGE and PRIV_CHECK events.

With Version 10 of Hitman username can also be used to enhance the absent/present batch and absent/present process events. If a username is specified with these events an additional check will be made to guarantee that the event is triggered if the job/process is not found under the specified username or, in the case of present events, is found under that specific username.

WATCH_NODE=nodename

Use the WATCH_NODE qualifier to specify the node that should be monitored by a NODE event. This qualifier cannot be a wildcard and can only be used with NODE events.

Using Hitman to monitor remote nodes

Hitman has the ability to use the absent and present process events to check remote nodes (ie: DECNET nodes not in the current cluster). To monitor remote nodes a special account must be created on the remote node. The create_remote_account.com procedure in hitman_com: can be copied to the remote node and run to create it; before running it you must decide on an available UIC to use. The UIC can be any available but we recommend you pick a UIC group that is higher than the value of the SYSGEN parameter MAXSYSGROUP since this account does not require any special privileges beyond WORLD and all accounts in groups 1 to MAXSYSGROUP have SYSPRV automatically.

The create_remote_account will:

11. Prompt for the UIC to add the account under
12. Check to ensure that the UIC is not in use
13. Add the user SAIGA_REMOTE to the SYSUAF file
14. Modify the SAIGA_REMOTE account to allow only network connections; this account cannot be used for interactive logins or to run batch jobs
15. Create the sys$sysdevice:[saiga_remote] directory
16. Translate the logical name Hitman_exe. If it exists it will automatically copy the check_process.exe and check_process.com files to the saiga_remote directory.
17. If the hitman_exe logical does not exist it will write out a description of the files needed to complete setting up the account.

The [saiga_remote] directory needs the following two files before it can be used:

check_remote.com from hitman_com on a node that Hitman is installed on

check_remote.exe from hitman_exe on a node that Hitman is installed on. In order for the executable to work the node must be the same hardware family (VAX or AXP) and the same major version of OpenVMS (ie: 6 or 7)

You should periodically purge the [saiga_remote] directory since netserver.log files will be created by the remote access process and can consume a significant amount of disk space over time.

Allowing users to lock their own terminals (HITLOCK)

This chapter shows you how users can use HITMAN's HITLOCK program to "lock" their terminals when they must leave them unattended momentarily.

To lock their terminal, users enter the command:

```
$ HITLOCK
```

which disconnects their process. The terminal-locking program, HITLOCK, requires them to enter their password(s) when they return. And, it deletes the process after a user-specified number of minutes if it does not receive a valid password. If users enter their password correctly, they are connected to their existing process with any privileges, logicals and symbols they have defined in the current session.

NOTE: The program HITLOCK must have the protection W:E so that other users can execute it. You should define a global symbol which executes HITLOCK with the qualifiers you want at your site. HITLOCK has its own command definition file (HITMAN_CDU:HITLOCK.CLD) which you should install in the system table (DCLTABLES) so that users will be able to use HITLOCK easily. An example of installing a command in DCLTABLES is given in Appendix C, "Moving Command Definitions".

Terminal locking will not work over the network. "RT" and "NT" terminals cannot use HITLOCK.

By default, HITLOCK will delete itself after 120 minutes (2 hours) if the user does not reconnect. HITMAN will delete the process at the normal time because it will appear idle, even if this time is less than the /TIMEOUT you specified; to prevent this add HITLOCK to the image protection list. Use the /TIMEOUT qualifier to alter the number of minutes before process deletion. The minimum value is 1 minute. The maximum value is 999 minutes. If /NOTIMEOUT is specified, a process running HITLOCK will remain on the system indefinitely.

By default, HITLOCK will limit the number of login failures to the system limit of SYI$_LGI_BRK_LIM. Use the /LIMIT qualifier to alter the number of login failures allowed before process deletion. The minimum value is 1. The maximum value is 99. If /NOLIMIT is specified, HITLOCK will allow unlimited password attempts which prevents anyone from logging you out by causing repeated login failures.

By default, HITLOCK will display the username and require users to enter only their password(s). Use the /NOUSERNAME qualifier to require users to enter both their username and password.

By default, HITLOCK will allow users to type "LOGOUT" at the password prompt. This allows

anyone to free up a terminal which has been locked by its owner. Use the /NOLOGOUT qualifier to prevent users from entering "LOGOUT" at the password prompt and logging the terminal off.

Killing processes regardless of activity (MAD_DOG)

This chapter shows you how to use HITMAN's MAD_DOG feature to kill all interactive and optionally batch, network and non-system detached processes whether they are idle or active.

Use MAD_DOG when you want to get everyone off your system regardless of activity. For example, you may want to kill processes on your system before your daily backup.

Overview

To setup and use MAD_DOG you must:

- create the MAD_DOG parameter file
- modify the MAD_DOG parameter file, if necessary, to customize it for your site, including adding any required user and image protections
- submit MAD_DOG.COM to a batch queue using /AFTER=HH:MM to instruct the system when to run MAD_DOG if necessary

Creating a new MAD_DOG parameter file

To create a new MAD_DOG parameter file containing default values, type:

```
$ @HITMAN_COM:CREATE_MAD_DOG_FILE
```

Modifying the MAD_DOG parameter file

You can modify the MAD_DOG parameter file by entering the command:

```
$ @HITMAN_COM:MODIFY_MAD_DOG_FILE
```

This redefines the HITMAN logicals to allow you to make changes to the MAD_DOG parameter file just as you would to your prime and non-prime files.

Using MAD_DOG

To invoke MAD_DOG, type:

```
$ SUBMIT/QUEUE=queue HITMAN_COM:MAD_DOG
```

HITMAN loads a special parameter file containing the commands:

```
$ HITMAN/INTERVAL=1 /WAIT=5 /FIRST_WARN=1 /SECOND_WARN=2 -
    /MULTIPLE /FIRST_MESSAGE="?U?, Please log out" -
    /SECOND_MESSAGE="?U?, Please log out" -
    /NOCPU_THRESHOLD /NOBIO_THRESHOLD /NODIO_THRESHOLD -
    /HIT=HIT /ON /DISABLE_PRIME
```

All interactive users **regardless of activity** will be warned every minute until they log out or are terminated after five minutes. After five minutes, HITMAN will revert to using whatever parameter file is current for that date, day and time.

The /MAD_DOG command **MUST** be submitted from a batch job. If it is submitted from an interactive job, HITMAN will reject it. This restriction prevents someone from triggering it accidentally. The user who submits the batch job is automatically protected.

Users that decide to run MAD_DOG on a regular basis before backups, or for some other reason, can use a HITMAN time event to submit MAD_DOG automatically at the appropriate time. Refer to the "Using HITMAN events to automate routine operations" section of this manual.

***CAUTION*:** If you delete the batch job that is running MAD_DOG before it finishes, HITMAN will be stuck in MAD_DOG mode. Always let the batch job complete normally. If HITMAN is ever stuck in MAD_DOG mode you can return to normal operations by deassigning the system logicals for PERM_DATA_PRIME and PERM_DATA_NONPRIME. If you normally use these logicals you should then redefine them with the correct values.

Setting up your own idle process procedures

This chapter shows you how to set up your own method(s) for dealing idle processes using the HITMAN USER_EXIT facility.

You can use the HITMAN USER_EXIT to invoke site-specific actions instead of, or in addition to, the standard HITMAN warn and kill procedures. Commonly, this is used to run-down databases which do not respond to $FORCEX properly but essentially any action can be taken that can be handled within a batch job. Since many packages, like databases, include DCL interfaces for management functions it is often possible to write your own exit procedures which are invoked by Hitman when a process is idle by submitting a batch job. The submitted batch is passed several parameters including the process ID (PID) of the process which Hitman has determined to be idle.

The previous versions of Hitman supported two types of user exits; the first was linking in a compiled Fortran module which took the actions desired for handling idle processes - with this method specific actions could be taken before the main process termination module was invoked. The second type of user exit was called the default user exit and was configured using a set of system logicals so that Hitman would submit a batch job whenever it saw a user idle in a particular image (the image name could be wildcarded). This facility worked very well but involved some overhead (translating logical names on each data collection) and was limited to only one special exit procedure. At the request of many users Version 10 of Hitman now allows as many special exit procedures as desired and eliminates the need for special logical names by storing the special exit details in its permanent data file. The default user exit used an image name for checking against the Hitman process table; for greater control this has been updated to use a complete path for the special image (version number is not included). Hitman submits the appropriate batch job for an idle process based on the image path that the process is executing. Processes for which no special exit procedure is found that meet all other idle process criteria will continue to be terminated normally. Users taking advantage of the old method of linking in a special user_exit fortran function will need to contact Technical Support at Saiga Systems to discuss the upgrade to Version 10.

HITMAN checks the USER_EXIT function for special exit procedures at the <u>end</u> of each data collection <u>before</u> any action has been taken. At this point, HITMAN has current information on every process on the system but it has not warned or killed anyone.

The /USER_EXIT qualifier is used to enable for disable this special exit checking. When the /NOUSER_EXIT qualifier is in effect Hitman will not perform the special exit routine check, regardless of whether or not special exits are defined. This allows you to dynamically include/exclude the USER_EXIT processing as you wish. The default setting is /NOUSER_EXIT.

The default user exit procedure added in Version 8 of Hitman that utilized the logical names hitman$userexit_image, hitman$userexit_queue, hitman$userexit_command_procedure,

hitman$userexit_username and hitman$userexit_debug has been replaced with this new functionality. During data file conversion from V9 to V10 Hitman will check for these logical names and if they are found it will add a special_exit record for them; since the old function used only image name and not path this newly added special_exit will need to be updated with the complete image path before it will function correctly.

Adding a special exit procedure

Adding a special exit procedure is fairly straightforward. It must be done from the DCL level; no menu interface has been added for this functionality yet. To begin you need the following pieces of information (the keyword used to add these items to a special_exit clause is shown in ***bold italics*** at the end of each item:

1. The path to the image for which a special exit is desired. This path will include device, directory, filename and type but not version number. Since there are many ways to enter a path in OpenVMS for matching purposes it is essential that it be specified the same way that it is shown by the show process /continuous command. In some cases this may mean configuring more than 1 special exit for the same image since the path to the image may be different depending on how it is invoked by various applications. ***Path=***

2. A DCL command procedure which takes the special actions required for users running the image specified by the path in step 1. ***Job=***

3. The name of a queue that these jobs should be submitted to. ***Queue=***

4. The username these jobs should be submitted to run under. ***Username=***

5. The action desired with the log file from the batch job; it may be kept (true) or automatically deleted when the job completes (false). ***Debug=***

6. The node(s) on which this special exit procedure should be used instead of Hitman's default procedure. ***Node=*** This qualifier was added since the logical names used in previous versions of Hitman for this functionality were node specific and could be different on each node.

An Example Special Exit

At Site A it has been determined that users are remaining logged in within the database application when they should not be. Since deleting these users processes may have a negative impact on the database it was decided that all idle users in the database would be forced out of the database with a special database administrator command for this and then their process would be suspended until they contacted operations to ask for their process to be unlocked. This was considered the best course of action since it would quickly educate the user community on how essential it is, for security reasons, to log out whenever a session is completed. By suspending their process instead

of terminating it management felt remaining logged in unnecessarily would become far more inconvenient than simply logging out and logging back in for the next database session.

Step 1: The Image Path

The database is invoked from a menu. By getting a user into the database and having them sit idle we can use the show process/id/continuous command to determine the exact path for the database image.
Our path clause will be:

```
                        Process SYSTEM                      11:30:17

   State                LEF                 Working set             275
   Cur/base priority    9/4                 Virtual pages         11202
   Current PC           8011D184            CPU time       0 00:00:01.17
   Current PSL          0000001B            Direct I/O              165
   Current user SP      7AEA74E0            Buffered I/O           1484
   PID                  000000AC            Page faults            3714
   UIC                  [SYSTEM]            Event flags        C1000001
                                                               C0000000

   SWIFT$DKA0:[GILBERTS.][CXX]PROGRAM5-2.EXE;2
```

path=swift$dka0:[gilberts.][cxx]program5-2.exe

Step 2: The DCL command procedure to be submitted
In this case the DCL command procedure will be stored in the Hitman_dat directory and is called suspend_idle_db_user.com. A simple example:

```
$ DBOPER FORCEX ‘p5'          ! P5 is the PID of the idle DB process
$ Wait 0:0:15                 ! Wait 15 seconds to allow FORCEX to complete
$ set process/suspend/id=’p5'  ! Suspend the idle process
$ username = f$getjpi (""p5"","USERNAME")
$ mail nl: SYSOP,’username’ /subject=”"username': Your process "p5' was suspended as idle”
$ exit
```

Our job clause will be:

job=hitman_dat:suspend_idle_db_user.com

Step 3: Controlling batch qualifiers
The special exit DCL command procedure should be submitted to the DB$OPER queue under username SYSTEM. The logs for these jobs should be kept and not automatically deleted.

Our batch related clauses will be:

queue=DB$OPER, username=SYSTEM, debug=TRUE

Step 4: Decide which node(s) this action should be taken on
Since we do not care about idle processes on our development node and the development and production nodes share a Hitman data file we can use the node clause to limit the special exit to one node. In a cluster with multiple nodes where a node, or nodes, are being excluded you must set up a special exit procedure for each node that you wish to special exit procedure to be in effect for. Our node clause is:

node=swift

Step 5: Add the special exit procedure and turn on user_exit checking
We can now add our special exit to the Hitman permanent data file and enable checking. We'll add it to both the prime and nonprime files since we want this action taken at anytime an idle database user is detected :

```
$ hitman /special_exit=(job=hitman_dat:suspend_idle_db_user.com, -
queue=db$oper, userame=system, debug=true, node=swift, -
path=swift$dka0:[gilberts.][cxx]program5-2.exe) /file=prime
$ hitman /special_exit=(job=hitman_dat:suspend_idle_db_user.com, -
queue=db$oper, userame=system, debug=true, node=swift, -
path=swift$dka0:[gilberts.][cxx]program5-2.exe) /file=nonprime
$ hitman /user_exit /file=both
```

The special exit is now in effect, Hitman will start checking for it with its next data collection.

HITMAN DCL Commands

This chapter shows you the purpose of each HITMAN command and lists the qualifiers that are valid to use with it. It then lists all the qualifiers in alphabetical order and describes each qualifier in detail.

It provides a complete command reference "manual" for a HITMAN user.

Command syntax

HITMAN commands look and act like DCL commands. The commands conform to normal OpenVMS DCL command syntax.

HITMAN commands have the format:

```
$ HITMAN[/qualifier...]
```

HITMAN is the name of the command and /qualifier is the name of a command qualifier. There are no parameters. Lowercase and uppercase characters in command and qualifier names are equivalent. As in DCL, '/', serves as a delimiter and command lines can be continued on to another line by terminating the line with a hyphen, pressing <RETURN> and typing the rest of the command on the next line.

Abbreviating commands

As with DCL, many HITMAN commands can be truncated to a unique string, for example, "HITMAN" can be truncated to "HI". This provides compact command entry for experienced users. In command procedures, it is best to avoid abbreviating command names and qualifiers. Please refer to the OpenVMS DCL manual for more details.

NOTE: Not all commands such as /AUTHORIZED and /AUTHORIZE_MESSAGE can be truncated to 2 or 3 characters. We recommend that you type in the full qualifier name. If you are going to abbreviate a command, please check that it is unique.

Editing command lines

As with DCL, HITMAN commands and qualifiers can be edited using terminal keys. Command-line editing is most useful for modifying long command lines.

HITMAN/START

This command starts the HITMAN running. (ie, it creates the detached process which monitors the system). If HITMAN is already running, the command will return the error message:

%HITMAN-E-HITSTART, HITMAN is already running

HITMAN/STOP

This command stops the HITMAN detached process. If HITMAN is not running, the command will return the informational message:

%HITMAN-I-NOTRUN, HITMAN is not running

HITMAN/STATUS

This command shows you the status of the HITMAN detached process including whether the day is prime or non-prime, what types of processes are eligible to be terminated and how often data is being collected. The first line indicates whether HITMAN is running.

HITMAN/DUMP/USER=username

This command shows you all the information HITMAN has on the specified user. If HITMAN is not running, the command will return the informational message:

%HITMAN-I-NOTRUN, HITMAN is not running

HITMAN/LIST

This command shows you all HITMAN's parameter settings. If HITMAN is not running, the command will return the informational message:

%HITMAN-I-NOTRUN, HITMAN is not running

In the case of the list command, this is only a warning and can be ignored.

NOTE: If you enter "HITMAN" on a line by itself without any qualifiers, no action will be taken, no messages will be generated and the "$" prompt will re-appear.

HELP

The HELP command provides access to HITMAN's on-line HELP text. HITMAN's HELP text can be installed into the system help library or into a user help library. To access HITMAN's help enter:

$ HELP HITMAN

HITMAN qualifier list

This next few pages contain a complete list of all HITMAN qualifiers. Subsequent pages contain detailed descriptions of each qualifier.

/ALARM	Send message to system console.
/ALLOW/USER=(u,...,u)	Define users who can set their own idle time.
/ASCII	Generate a comma-delimited log file.
/AUTHORIZED/USER=(u,..,u)	Define users who are allowed to have privilege.
/AUTHORIZE_MESSAGE=" "	Modify message text sent to unauthorized privileged users
/AUTOHIT/IMAGE=(i,..,i)	Define images to hit regardless of activity.
/AUTOHIT_MESSAGE=" "	Modify autohit message text.
/BELL	Ring terminal bell for message.
/BIO_THRESHOLD=n	Set BIO "idle" limit.
/BROADCAST=class	Specify broadcast class for user messages.
/CLEAR	Clear screen before terminating process.
/COMMAND	Generate a DCL command procedure that could be used to recreate the current parameter file
/COMMON=common_type	Link processes through a common field.
/CONNECT	Terminate processes after a maximum amount of connect time.
/CONSOLE=console_type	Send messages to the designated operator class.
/CPU_THRESHOLD=n	Set CPU "idle" time limit.
/DAY=day_of_week	Specify day for prime/non-prime processing.
/DECWINDOWS[=batch_queue]	Specify handling queue for DECwindows
/DIALUP	Set wait and warning or connect times for dialup processes.
/DIO_THRESHOLD=n	Set DIO "idle" limit.
/DISABLE_PRIME	Disable prime/non-prime processing.
/DISCONNECT	Specify disconnect rather than termination.
/DUMP/USER=u	Dump user information to output file.
/ENABLE_PRIME	Enable prime/non-prime processing.
/EVENT=(event_type.,.,.)	Add monitor event.
/EXCLUDE=protect_list	Reverse the function of a protection list.
/EXIT_CODE	Specify the exit code for terminated images
/FILE=file_type	Select permanent data file type.
/FINAL_MESSAGE=" "	Define termination message.
/FIRST_MESSAGE=" "	Define first warning message.
/FIRST_WARN=n	Set time to receive first warning.
/FORCE_WAIT=n	Define wait for force exit.
/HIT=hit_mode	Set mode in which HITMAN runs.
/HOLIDAY=(d,...,d)	Specify date as non-prime day.
/IDENTIFIER=(i,..,i)	Specify identifier for wait, warn, connect or protect.
/IMAGE=(i,..,i)	Specify image for wait, warn, connect or protect.
/IMAGE_NAME	Enable hitting of processes executing images.

/INTERVAL=n	Set interval between data collections.
/KILL	Terminate process upon receiving message.
/LIST[=filename]	List HITMAN parameters on the screen or output file.
/LOG	Close existing log file and start a new one.
/MAIL	Send mail messages to terminated users.
/MAX_IDLE=n	Set maximum limit for user-specified idle time.
/MENU	Start HITMAN menu-driven screen interface.
/MIN_IDLE=n	Set minimum limit for user-specified idle time.
/MINIMUM_PROCESS=n	Set minimum number of processes that must be on the system before HITMAN will start hitting idle users.
/MULTIPLE	Send multiple messages to users.
/NAME=image name	Change HITMAN process name.
/OFF	Turn data collection off.
/ON	Turn data collection on.
/OPERATOR_CLASS=class	Set operator class to send messages to.
/OUTPUT[=file_name]	Specify output destination.
/PATH=path	Specify a path for protecting or special time limits
/PRIME_END=time	Set end of prime time.
/PRIME_START=time	Set start of prime time.
/PRIORITY=n	Set base priority of detached process.
/PROCESS=(p,..,p)	Protect process from being hit.
/PROTECT	Protect user, image, etc. from being hit.
/REPORT=[filename]	Generate complete log file report.
/RESEQUENCE_EVENTS	Resequence all the hitman events
/SECOND_MESSAGE	Define second warning message.
/SECOND_WARN=n	Set time to receive second warning.
/SERVER=(s,..,s)	Specify server id for wait, warn, connect or protect.
/SPECIAL_EXIT=(path,...)	Setup a special exit routine to be invoked when a process is running a specific program instead of terminating it
/START	Start the detached process.
/STATUS	Show HITMAN current status.
/STOP	Stop the detached process.
/SUMMARY	Generate a summary report of the Hitman log file
/SWAP	Swap processes in to get information.
/SYS_PROTECT	Protect system processes.
/TABLE_SIZE=value	Specify the size of Hitmans process monitoring table
/TERMINAL=(t,...,t)	Specify terminal for wait, warn, connect or protect.
/TERMINATION=method	Define termination procedure.
/TIMEOUT=n	Set timeout for disconnected process.
/TYPE=(t,...,t)	Specify process types to be terminated.
/UIC=(uic,..,uic)	Specify UIC for wait, warn, connect or protect.
/UPDATE_INTERVAL=n	Set interval between log file updates.
/USER=(u,..,u)	Specify user for wait, warn, connect or protect.

/USER_EXIT	Invoke user exit routine.
/VERSION	Display HITMAN version.
/WAIT=n	Set number of minutes of idle time.
/WS_DEFAULT	Set the working set default value to be used when Hitman creates the detached process
/WS_EXTENT	Set the working set extent value to be used when Hitman creates the detached process
/WS_QUOTA	Set the working set quota value to be used when Hitman creates the detached process

Negatable qualifiers

/NOALARM	Do not send message to system console.
/NOBELL	Do not ring terminal bell.
/NOBIO_THRESHOLD	Turn off BIO threshold checking.
/NOCLEAR	Disable terminal screen clear.
/NOCOMMON	Disable common field checking.
/NOCONSOLE	Stop logging hits to the console.
/NOCPU_THRESHOLD	Set CPU "idle" to zero.
/NODAY	Designate a day as non-prime.
/NODIALUP	Eliminate dialup wait and warning times.
/NODIO_THRESHOLD	Turn off DIO threshold checking.
/NOEVENT=(ID=n)	Remove event from event list.
/NOEXCLUDE	Disable protection list exclusion.
/NOFINAL_MESSAGE	Eliminate final hit message.
/NOFIRST_MESSAGE	Eliminate first warning message.
/NOFIRST_WARN	Disable first warning.
/NOHOLIDAY	Remove date from holiday list.
/NOIDENTIFIER=(i,..,i)	Remove identifier from wait, warn or protect list.
/NOIMAGE=(i,..,i)	Remove image from wait, warn, protect, autohit list.
/NOIMAGE_NAME	Protect processes executing an image.
/NOKILL	Do not terminate authorize or autohit processes.
/NOLOG	Close log file and DO NOT open a new one.
/NOMAIL	Suppress termination mail messages to users.
/NOMINIMUM_PROCESS	Disable minimum process checking.
/NOMULTIPLE	Disable multiple messages.
/NOPATH	Remove path from list
/NOPROCESS=(p,...,p)	Remove image from protect list.
/NOSECOND_MESSAGE	Eliminate second warning message.
/NOSECOND_WARN	Disable second warning messages.
/NOSERVER=(s,..,s)	Remove server from wait, warn or protect list.
/NOSPECIAL_EXIT=(path)	Remove the special exit handler set up for this path.
/NOSWAP	Do not swap processes in to get information.
/NOSYS_PROTECT	Do not protect system processes.

/NOTERMINAL=(t,..,t)	Remove terminal from wait, warn or protect list.
/NOUIC=(uic,..,uic)	Remove UIC from wait, warn or protect list.
/NOUSER=(u,..,u)	Remove user from wait, warn, protect, authorized list.
/NOUSER_EXIT	Disable user exit routine.

HITMAN DCL Qualifiers Reference

/ALARM

Modify Parameters *Prime/Nonprime* *Modify Message Text*	**MEN**

/ALARM (Default)
/NOALARM

Use the /ALARM qualifier to send a message to the operator class specified by /OPERATOR_CLASS when a message is sent to a user. This qualifier works in conjunction with all of the message text qualifiers such as /AUTHORIZE_MESSAGE and /FIRST_MESSAGE. You do not have to respecify the message text to change this qualifier.

Example

$ HITMAN/SECOND_MESSAGE/ALARM

Send a message to the system console when a user receives a second warning message.

$ HITMAN/FINAL_MESSAGE/NOALARM

Do not send a message to the system console for final messages.

/ALLOW

Modify Parameters *Prime/Nonprime* *Modify Idle & Warn Times* *Modify Set Idle List*	**MENU**

/ALLOW/USER=(u,...,u)
/ALLOW/NOUSER=(u,...,u)

Use the /ALLOW qualifier to specify users who are allowed to set their own idle times

Example

$ HITMAN/USER=LARRY/ALLOW

Allow LARRY to set his own idle time.

/ASCII

There is no menu access for this function	**MENU**

/REPORT=logfile/OUTPUT=reportname/ASCII

Use the /ASCII qualifier in conjunction with the /REPORT and /OUTPUT qualifiers to generate the log in comma delimited format, with string values in double quotes, so the data can be easily imported by

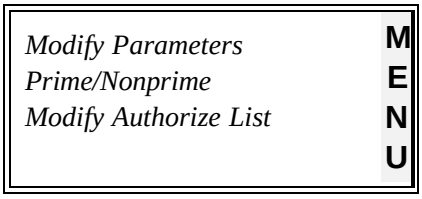
database, report writing or graphics tools. The default delimiter is the comma character (","); you can change the delimiter by defining HITMAN's HITMAN$DELIMITER logical.

Example

$ HITMAN /REPORT=HITMAN_LOG_FILE.NODE1 /OUTPUT=LOG_REP.N1-
/ASCII

Read HITMAN_LOG_FILE.NODE1 and create a report called LOG_REP.N1 in comma-delimited format. Both of these files will be in the HITMAN_DAT directory.

AUTHORIZED/USER=(u,...,u)



/AUTHORIZED/USER=(u,...,u)
/AUTHORIZED/NOUSER=(u,...,u) (Default)

Use the /AUTHORIZED qualifier to specify users who are allowed to have privileges in the ALL category. Wildcards are allowed. If a user has privilege and is not on the list, HITMAN will stop the process immediately. To remove a username from the authorized list, use /AUTHORIZED/NOUSER=(username). The default is not to check privileges. Use /AUTHORIZED/USER=ALL to disable privilege checking. There is no maximum number of users you can add to the authorized list.

Example

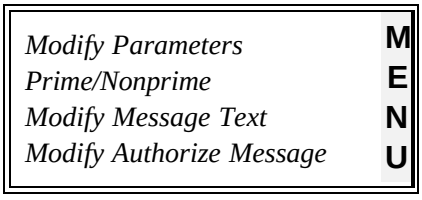
$ HITMAN/AUTHORIZED/USER=LARRY

Authorize LARRY to have ALL category privileges.

$ HITMAN/AUTHORIZED/USER=(LARRY,MOE,CURLY)

Remove LARRY, MOE and CURLY from the list of authorized users.

/AUTHORIZE_MESSAGE="text"



/AUTHORIZE_MESSAGE="text"
(Default = "** PROCESS IN UNAUTHORIZED FOR PRIVILEGE ** ?L?")
/NOAUTHORIZE_MESSAGE

Use the /AUTHORIZE_MESSAGE qualifier to modify the message that is sent when a user is unauthorized. This qualifier works in conjunction with the /AUTHORIZE function. You can also use the /BELL, /ALARM, and /KILL qualifiers with this.

Example

$ HITMAN/AUTHORIZE_MESSAGE="?U?, You are unauthorized"

Change the authorize message that is sent to the user.

/AUTOHIT/IMAGE=(i,...,i)

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Autohit List	**N**
	U

/AUTOHIT/IMAGE=(i,...,i)
/AUTOHIT/NOIMAGE=(i,...,i) (Default)

Use the /AUTOHIT/IMAGE qualifiers to specify images which are to be hit as soon as HITMAN sees them on the system. Wildcards are allowed. When a process runs an image that is on the autohit list, it will be immediately stopped, UNLESS the process is protected because of a protect list. To remove an image from the autohit list, use /AUTOHIT/NOIMAGE=(image_name). If no /AUTOHIT is specified, all images are allowed to run unless they are idle. There is no maximum number of users you can add to this list.

Example

$ HITMAN/AUTOHIT/IMAGE=OTHELLO

Terminate any process running OTHELLO.

$ HITMAN/AUTOHIT/NOIMAGE=(ADVENTURE,OTHELLO)

Remove ADVENTURE and OTHELLO from autohit list.

/AUTOHIT_MESSAGE="text"

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Message Text	**N**
Modify Autohit Message	**U**

/AUTOHIT_MESSAGE="text"
(Default = "** PROCESS IS RUNNING A RESTRICTED IMAGE ** ?L?")
/NOAUTOHIT_MESSAGE

Use the /AUTOHIT_MESSAGE qualifier to modify the message that is sent to a user if they are found running an autohit image. This qualifier works in conjunction with the /AUTOHIT qualifier. You can also use the /BELL, /ALARM, and /KILL qualifiers with this.

Example

$ HITMAN/AUTOHIT_MESSAGE="You are running an AUTOHIT image"

Change the message for AUTOHIT images.

/BELL

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Message Text	**N**
Any message	**U**

/BELL (Default)
/NOBELL

Use the /BELL qualifier to ring the bell on the user's terminal when a message is sent. This qualifier works in conjunction with all of the message qualifiers. You do not have to respecify the text of the message to change this qualifier.

Example

$ HITMAN/FIRST_MESSAGE/BELL

Ring the user's terminal bell when the first message is sent.

$ HITMAN/AUTHORIZE_MESSAGE/NOBELL

Do not ring the user's terminal bell when the authorize message is sent.

/BIO_THRESHOLD=n

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Threshold Lists	**N**
System OR Image	**U**

/BIO_THRESHOLD=n (Default = 1)
/NOBIO_THRESHOLD
/BIO_THRESHOLD=n /IMAGE=image

Use the /BIO_THRESHOLD qualifier to define the minimum buffered I/O's that a process must use to be considered active. The BIO threshold is the number of I/O's per collection interval that must be used. If a process does not use this amount during a collection interval or exceed /CPU_THRESHOLD or /DIO_THRESHOLD, it is considered idle for that interval. This is mainly used for processes which use some small amount of I/O even though they are truly idle. The amount to use for BIO threshold depends on your Alpha AXP and VAX model and the image that is running. Please be careful when you raise the /BIO_THRESHOLD since you may start killing active processes. Use the /HIT=WARN mode with /NOBIO_THRESHOLD, and see if it is missing processes that are truly idle, then set a BIO threshold until it is getting all the processes that you want it to get. The minimum value allowed is 0. The maximum value allowed is 999. This qualifier may also be used in conjunction with the /IMAGE qualifier to set a BIO threshold for a specific program higher or lower than the system default value. If the image that a user is running is found in the image threshold list the thresholds for that image will be used instead of the default values.

Example

$ HITMAN /BIO_THRESHOLD=3

Any process with less than 3 buffered I/O's per collection interval will be considered idle for that interval.

$ HITMAN /NOBIO_THRESHOLD

Turn off BIO threshold checking. HITMAN will not use BIO's when deciding if a process is active.

$ HITMAN /BIO_THRESHOLD=10 /IMAGE=CAD

Processes running CAD.EXE are considered idle if they are using less than 10 BIOs in each data collection interval.

/BROADCAST=broadcast_class

Modify Parameters
Prime/Nonprime
Modify Flags & Values
Broadcast
MENU

/BROADCAST=USER1 - USER16

Use the /BROADCAST qualifier to specify the broadcast type at which HITMAN messages are sent. This allows the user to receive only HITMAN messages using $ SET BROADCAST=USERx where x is the same number as specified in /BROADCAST. All messages are sent to the general category as well as the category specified by the user. This value must be between USER1 and USER16. For more information on $ SET BROADCAST, type $ HELP SET BROADCAST.

Example

$ HITMAN/BROADCAST=USER11

Send messages to users who have GENERAL or USER11 categories enabled.

/CLEAR

Modify Parameters
Prime/Nonprime
Modify Flags & Values
Clear
MENU

/CLEAR
/NOCLEAR (Default)

Use the /CLEAR qualifier to clear the user's terminal screen before the final message is sent and the process is deleted. Use this qualifier if terminal screens might contain sensitive information which should be erased.

Example

$ HITMAN/CLEAR

Clear user screen when process is terminated.

/COMAND

This item is not available through the menu

/COMMAND
/NOCOMMAND (Default)

Use the /COMMAND qualifier to create a DCL command procedure that contains all the settings necessary to recreate your Hitman parameter file at the DCL level. This option provides a simple text backup of your current parameters. Use the /file qualifier to control which parameter file is written out in DCL format; use the /output qualifier to specify a

file name for the created DCL command procedure. If you omit the /output qualifier the output will be directed to your terminal. After the file is created any text editor can be used to customize it; this makes multiple changes to your parameter file possible using a text editor.

Example

$ Hitman /Command /File=nonprime /Output=nonprime.com

This command creates a DCL command procedure called nonprime.com. Executing this procedure would recreate your current nonprime hitman parameters file.

/COMMON=common_field

/COMMON=SERVER
/COMMON=UIC (Default for /COMMON)
/COMMON=USERNAME
/NOCOMMON (Default)

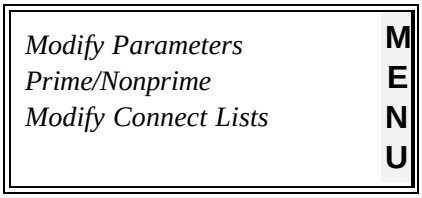
Use the /COMMON qualifier to specify a common field to link two detached processes which HITMAN would otherwise consider unrelated. Based on the specified common field, HITMAN will look at processes and if one is active, it will consider both active. This is useful for multiple sessions on a server port and client/server database operations. If you specify $ HITMAN/COMMON=SERVER, HITMAN will terminate the sessions on that port only if all of them are idle. If one of them is active, all are considered active.

Example

$ HITMAN/COMMON=UIC

Link processes through the common field UIC.

/CONNECT



/CONNECT

Use the /CONNECT qualifier to log users off after a specified amount of time regardless of activity. The /CONNECT qualifier must be used in conjunction with one of the /DIALUP, /IDENTIFIER, /IMAGE, /PATH, /PROCESS, /SERVER, /TERMINAL, /UIC or /USER qualifiers, it can not be used alone to specify system wide maximum connect times. It is used in conjunction with the /WAIT qualifier. For more information on associated qualifiers, refer to the /WAIT qualifier documentation.

When you decrease a connect time, there may be processes on the system which have already accumulated the reduced amount of connect time. HITMAN will immediately warn them and then kill them on the next data collection. To avoid problems, we recommend you stop HITMAN, decrease the connect times and restart HITMAN. This may also occur when HITMAN changes permanent datafiles if the wait and warning times are different; HITMAN doesn't reset it's internal process table where it keeps a record of idle time when it changes files.

Example

$ HITMAN /CONNECT /USER=FRED -
/FIRST_WARN=10 /SECOND_WARN=25 /WAIT=30 /CONNECT

Set the maximum connect time for user FRED to 30 minutes.

$ HITMAN /UIC="[20,*]" /WAIT=10 /CONNECT

Set the maximum connect time for processes in the UIC group 20 ([20,*]) to 10 minutes.

$ HITMAN /CONNECT /USER=*-
/FIRST_WARN=20/SECOND_WARN=25/WAIT=30/CONNECT

Set a maximum connect time of 30 minutes for every user on the system that isn't protected.

/CONSOLE=console_class

Modify Parameters
Prime/Nonprime
Modify Flags & Values
Console
MENU

/CONSOLE
/CONSOLE=(ERROR, EVENT, LOG_FILE, PROCESS, -
TIME) (Default)
/NOCONSOLE

Use the /CONSOLE qualifier to control which messages are logged to the console as they occur. The default action is to log all messages to the console. To stop messages from being logged, use /NOCONSOLE. Specify ERROR to log error messages to the console. Specify EVENT to log event messages to the console. Specify LOG_FILE to log when a new log file is started. Specify PROCESS to log process warnings and terminations. Specify TIME to log hourly time stamp messages to the console. All operator messages are sent to the operator class specified by /OPERATOR_CLASS. Messages are logged to all operators enabled as the current operator class. Remember, the main console, OPA0:, cannot be disabled as a security operator.

Example

$ HITMAN/CONSOLE=(NOEVENT,NOTIME)

Don't send event or time messages to the operator's console.

$ HITMAN/NOCONSOLE

Don't send messages to the operator's console.

/CPU_THRESHOLD=n

MENU
Modify Parameters
Prime/Nonprime
Modify Threshold Lists
System OR Image

/CPU_THRESHOLD=n (Default = 3)
/NOCPU_THRESHOLD
/CPU_THRESHOLD=n /IMAGE=image

Use the /CPU_THRESHOLD qualifier to define the minimum CPU usage that a process must use to be considered active. The CPU threshold is the number of 10 millisecond units the process must use per collection interval to be considered active. If a process does not use this amount during a collection interval or exceed /BIO_THRESHOLD or /DIO_THRESHOLD, it is considered idle for that interval. This is mainly used for processes which use a small amount of CPU time even though they are truly idle. The setting for the CPU threshold depends on your Alpha AXP and VAX model, and the image that is running. Please be careful when you raise the /CPU_THRESHOLD since you may start killing active processes. Use the /HIT=WARN mode with /NOCPU_THRESHOLD, and see if it is missing processes that are truly idle, then set a CPU threshold until it is getting all the processes that you want it to get. The minimum value allowed is 0, the maximum is 999. This qualifier may also be used in conjunction with the /IMAGE qualifier to set a CPU threshold for a specific program higher or lower than the system default value. If the image that a user is running is found in the image threshold list the thresholds for that image will be used instead of the default values.

Example

$ HITMAN/CPU_THRESHOLD=5

Any process with less than 50 (5x10) milliseconds of CPU time per collection interval will be considered idle for that interval.

$ HITMAN/CPU_THRESHOLD=0

Any CPU usage in the collection interval will make the process active for that interval.

$ HITMAN /CPU_THRESHOLD=30 /IMAGE=MENU

Any process running MENU.EXE is considered idle if it uses less than 300 (30x10) milliseconds of CPU time per collection interval. This is necessary because MENU has some background CPU usage because it updates a time field on the screen even when users are idle.

/DAY=day_of_week /PRIME_START=time - /PRIME_END=time

MENU
Modify Parameters
Prime/Nonprime
Modify Prime Time

/DAY=ALL (Default)
/DAY=MONDAY
/DAY=TUESDAY
/DAY=WEDNESDAY

/DAY=THURSDAY
/DAY=FRIDAY
/DAY=SATURDAY
/DAY=SUNDAY
/DAY=WEEKDAYS
/DAY=WEEKEND
/NODAY=(day_of_week)

Use the /DAY qualifier to define individual days or groups of days as prime or non-prime, and specify when prime time starts and ends. The keyword WEEKDAYS refers to Monday through Friday. The keyword WEEKEND refers to Saturday or Sunday. The keyword ALL refers to all days of the week Monday through Sunday. By default, HITMAN considers WEEKDAYS to be prime and WEEKENDS to be non-prime. By default, HITMAN considers prime time to start at 8:00 a.m. and end at 5:00 p.m. For prime days, start and end qualifiers must be used together for this feature to work. When you enter these qualifiers, both the prime and non-prime permanent data files are updated regardless of the /FILE qualifier setting.

Example

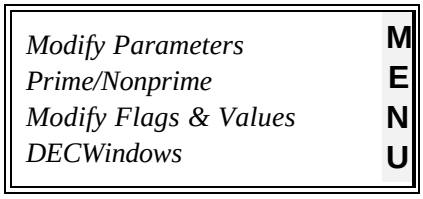
$ HITMAN/DAY=WEEKDAY/PRIME_START=7:30/PRIME_END=17:00

Start prime time at 7:30 AM and end it at 5:00 PM Monday through Friday.

$ HITMAN/NODAY=WEEKEND

Set Saturday and Sunday to non-prime days.

/DECWINDOWS[=queue_name]



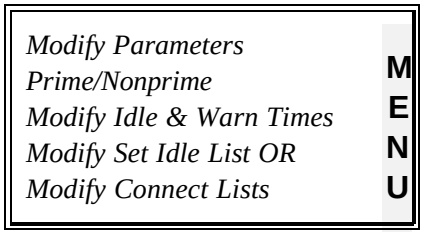
/DECWINDOWS=queue_name
(Default /DECWINDOWS=SYS$BATCH)

Use the /DECWINDOWS qualifier to specify the batch queue to be used to shutdown DECwindows processes. By default, HITMAN will use SYS$BATCH to do the shutdown. You can specify another queue if you wish. Please note, all this qualifier does is specify the queue to be used. It does not enable or disable DECwindows shutdowns.

Example

$ HITMAN/DECWINDOWS=FAST$BATCH

Use FAST$BATCH to run the DECwindows shutdown procedure.



/DIALUP/WAIT=n

/DIALUP /WAIT=n
/DIALUP /WAIT=n /CONNECT
/NODIALUP (Default)

Use the /DIALUP qualifier to set wait and warning times or connect time limits for dialup processes. It is used in conjunction with the /WAIT, /FIRST_WARN and /SECOND_WARN qualifiers, and is invalid if used with any other qualifiers; you must specify at least one of these three qualifiers. Use the /DIALUP qualifier to give dialup users different wait and warning times. You may also add /CONNECT to enforce connect time limits on dialup processes.

See the chapter "Warning idle users" for more information on the order of precedence of wait and warning times.

Example

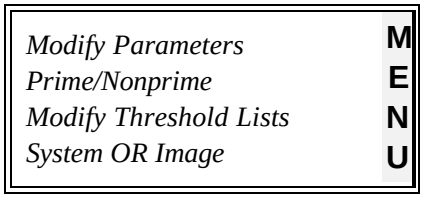
$ HITMAN/DIALUP/WAIT=10/FIRST_WARN=5/NOSECOND_WARN

Set the first warning time to 5 minutes and wait time to 10 minutes for all dialup processes.

$ HITMAN/NODIALUP

Disable specific wait and warning times for dialup users.

/DIO_THRESHOLD=n



/DIO_THRESHOLD=n
/DIO_THRESHOLD=n /IMAGE=image_name
/NODIO_THRESHOLD (Default=0)

Use the /DIO_THRESHOLD qualifier to define the minimum direct I/O's that a process must use to be considered active. The DIO threshold is the number of I/O's per collection interval that must be used. If a process does not use this amount during a collection interval or exceed /CPU_THRESHOLD or /BIO_THRESHOLD, it is considered idle for that interval. This is mainly used for processes which use some small amount of I/O even though they are truly idle. The amount to use for DIO threshold depends on the image that is running. Please be careful when you raise the /DIO_THRESHOLD since you may start killing active processes. Use the /HIT=WARN mode with /NODIO_THRESHOLD, and see if it is missing processes that are truly idle, then set a DIO threshold until it is getting all the processes that you want it to get. The minimum value allowed is 0. The maximum value allowed is 999. This qualifier may also be used in conjunction with the /IMAGE qualifier to set a DIO threshold for a specific program higher or lower than the system default value. If the image that a user is running is found in the image threshold list the thresholds for that image will be used instead of the default values.

Example

$ HITMAN/DIO_THRESHOLD=2

Any process with less than 2 direct I/O's per collection interval will be considered idle for that interval.

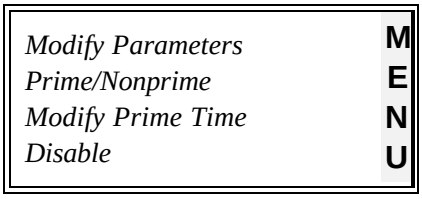
$ HITMAN /NODIO_THRESHOLD

Turn off DIO threshold checking. HITMAN will not use DIO's when deciding if a process is active.

$ HITMAN /DIO_THRESHOLD=5 /IMAGE=WORDPROC

Processes running WORDPROC.EXE are considered idle if they use less than 5 direct I/Os in a data collection interval. This threshold was increased because WORDPROC uses some resources when users are idle because of regular timed backups of the currently open documents.

/DISABLE_PRIME



/DISABLE_PRIME

Use the /DISABLE_PRIME qualifier to turn off prime/non-prime processing regardless of the day of the week or the time of day. HITMAN will use the prime permanent data file continuously. The qualifier reverses the effect of /ENABLE_PRIME.

Example

$ HITMAN/DISABLE_PRIME

Use the prime permanent data file until prime/non-prime checking is enabled.

/DISCONNECT

/DISCONNECT

Use the /DISCONNECT qualifier to specify users or images which are to be disconnected from their terminal rather than terminated.

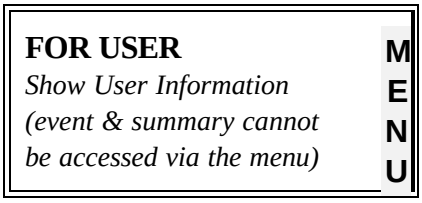
Example

$ HITMAN/USER=LARRY/DISCONNECT

Disconnect LARRY rather than terminating his process.

$ HITMAN/IMAGE=WPX*/DISCONNECT

Disconnect processes running any image starting with "WPX".



/DUMP

```
/DUMP/USER=username        ! Full details for one username
/DUMP=EVENT                    ! Summary of information for all events
/DUMP=SUMMARY                  ! Summary of system wide information
/DUMP=LIST                     ! One line for each process on system
```

Use the /DUMP qualifier to show all the information HITMAN has on a specific user, a summary table of all the processes on this node with information about whether or not HITMAN considers them eligible to be terminated or a summary of all the events in your HITMAN parameters with information about whether or not HITMAN has seen these events as true. HITMAN writes user information to HITMAN_DAT:DUMP.DMP, summary information to HITMAN_DAT:DUMP_SUMMARY.DMP event information to HITMAN_DAT:DUMP_EVENT.DMP or list information to HITMAN_DAT:DUMP_LIST.DMP. If the dump file exists, HITMAN appends the additional information to it. Otherwise, it creates a new file. For the default user dump you must specify a username with the /user= qualifier; wildcards are not allowed. You can only dump one user at a time. This qualifier is used for analyzing HITMAN's behaviour such as determining why a process is not being killed or why an event doesn't seem to be being reported. A sample of the different /DUMP output reports with a complete explanation of the various report items is included in the section "Getting information on what HITMAN is monitoring" (page 211).

Example

$ HITMAN/DUMP/USER=LARRY/OUTPUT=SAIGA.LIS

Dump all HITMAN information about LARRY to HITMAN_DAT:SAIGA.LIS.

$ HITMAN /DUMP=EVENT

Dump all HITMAN information about events that are being monitored to HITMAN_DAT:DUMP_EVENT.DMP.

$ HITMAN /DUMP=SUMMARY

Dump summary information about the processes currently on the system with some statistics for how many processes have been warned and terminated since Hitman was last started. Output is written to HITMAN_DAT:DUMP_SUMMARY.DMP.

$ HITMAN /DUMP=LIST

Dump summary information about every process on the system that Hitman is monitoring to hitman_dat:dump_list.dmp.

/ENABLE_PRIME

Modify Parameters
Prime/Nonprime
Modify Prime Time
Enable

MENU

/ENABLE_PRIME

Use the /ENABLE_PRIME qualifier to turn prime/non-prime processing back on after it has been disabled. This qualifier reverses the effect of /DISABLE_PRIME.

Example

$ HITMAN/ENABLE_PRIME

Check prime/non-prime settings and use the appropriate file.

/EVENT

The /EVENT qualifier is explained in detail in the section on maintaining events at the DCL level on page 134.

/EXCLUDE=protection_list

MENU
Modify Parameters
Prime/Nonprime
Modify Flags & Values
Exclude

/EXCLUDE=IDENTIFIER
/EXCLUDE=IMAGE
/EXCLUDE=PATH
/EXCLUDE=PROCESS
/EXCLUDE=SERVER
/EXCLUDE=TERMINAL
/EXCLUDE=UIC
/EXCLUDE=USER
/NOEXCLUDE (Default)

Use the /EXCLUDE qualifier to reverse the function of a protection list. When you specify /EXCLUDE for a protection list, every process on the system is protected except for the entries in that protection list. All other protection lists are not checked when a list is EXCLUDED. You can use /EXCLUDE to test HITMAN on a limited set of users.

Example

$ HITMAN/PROTECT/USER=SAIGA
$ HITMAN/EXCLUDE=USER

Protect all users on the system except SAIGA.

/EXIT_CODE=value

MENU
Modify Parameters
Prime/Nonprime
Modify Exit Code List

/EXIT_CODE=n

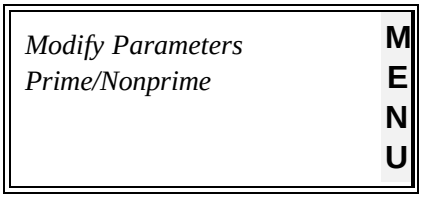
Use the /EXIT_CODE qualifier in conjunction with the /IMAGE qualifier to specify the exit code for an image which HITMAN sends a $FORCEX command to. This qualifier allows you to customize HITMAN to databases which require a specific exit value.

Example

$ HITMAN/EXIT_CODE=1/IMAGE=DSUPD

Set the exit code of the image DSUPD to 1 when it is forced to exit.

/FILE=file_type



/FILE=PRIME
/FILE=NONPRIME
/FILE=BOTH

(Default is **current** file based on prime start and end times)

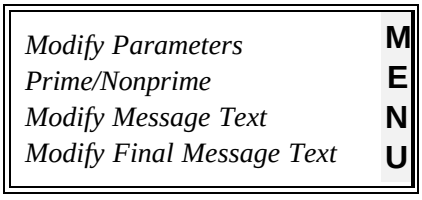
Use the /FILE qualifier to specify which permanent data file you want to use. If you do not use this qualifier, HITMAN uses the current permanent data file. If you are not using the prime data file feature (ie. /DISABLE_PRIME is in effect), HITMAN will always use the PRIME data file. By default, prime/non-prime days, start times and end times are updated in both files simultaneously. For all other parameters, you have to specify the file you want to update. For more information on PRIME and NONPRIME data files, see the chapter "Using different data files based on time or date".

Example

$ HITMAN/FILE=NONPRIME/LIST

List the non-prime data file.

/FINAL_MESSAGE="text"



/FINAL_MESSAGE="text"
/NOFINAL_MESSAGE
(Default = "** HITMAN Termination, You have been ?M? for ?I? minutes **?L?")

Use the /FINAL_MESSAGE qualifier to specify the message sent to the terminal when a user is terminated due to lack of activity. Like normal OpenVMS DCL commands, the text string must be enclosed in double quotes. If no text is provided then the default hit message is used. The final message text can be up to 200 characters long AFTER variable substitution.

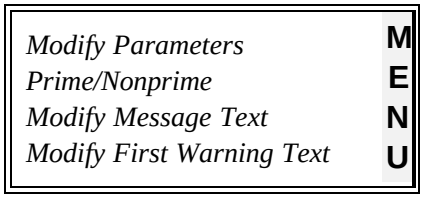
Example

$ HITMAN/FINAL_MESSAGE="LACK OF ACTIVITY TERMINATION!!"
Change the text of the final message.

$ HITMAN/FINAL_MESSAGE
Use the default final message text.

$ HITMAN/NOFINAL_MESSAGE
Do not send the final message.

/FIRST_MESSAGE="text"



/FIRST_MESSAGE="text"
/NOFIRST_MESSAGE
(Default ="** HITMAN Warning, You have been ?M? for ?I? minutes **?L?")

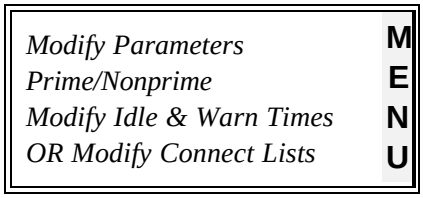
Use the /FIRST_MESSAGE qualifier to specify the first warning message. Like normal OpenVMS DCL commands, the text string must be enclosed in double quotes. If no text is provided, the default warning message is used. The first message text may be up to 200 characters long AFTER variable substitution. We recommend always ending the first message text with ?L?, which inserts a carriage return and line feed, to prevent subsequent messages from being written overtop of the first message.

Example

$ HITMAN/FIRST_MESSAGE="?U?, you have been idle for ?I? minutes."
Use the default first warning message.

$ HITMAN/NOFIRST_MESSAGE
Do not send the first warning message.

/FIRST_WARN=n



/FIRST_WARN=n (Default = 20)
/NOFIRST_WARN

Use the /FIRST_WARN qualifier to specify the time to send the first warning message. After users have accumulated the specified amount of idle time (or connect time if applicable), they will receive a warning message. The minimum allowed is twice the data collection interval. The maximum allowed is either the /SECOND_WARN or /WAIT time. This qualifier is used in conjunction with the /SECOND_WARN and /WAIT qualifiers. This qualifier cannot be abbreviated.

Example

$ HITMAN/FIRST_WARN=25/SECOND_WARN=35/WAIT=50

Send the first warning time after 25 minutes of idle time.

/FORCE_WAIT=n

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Flags & Values	**N**
Force Wait	**U**

/FORCE_WAIT=n (Default = 5)

Use the /FORCE_WAIT qualifier to specify the maximum amount of time allowable for an image run-down. The amount of time that an image takes to run-down depends on how heavily loaded your system is. If your site is handling many ALL-IN-1 users, you may want to increase the /FORCE_WAIT time to 10 seconds. The minimum allowed is 5 seconds. The maximum allowed is 60 seconds. This qualifier is used when you specify /TERMINATION=BOTH. If the image has not rundown within the time allowed, HITMAN deletes the process regardless and writes a log record indicating this.

NOTE: HITMAN takes only as much time as it needs to do an image run-down no matter how long the /FORCE_WAIT time is. HITMAN now sends a second forced exit by default if the image has not run down; this handles complex applications such as All-In-1 more cleanly.

Example

$ HITMAN/FORCE_WAIT=10

Set the maximum amount of time allowable for an image run-down to 10 seconds.

/HIT=hit_mode

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Flags & Values	**N**
Hit	**U**

/HIT=HIT
/HIT=WARN
/HIT=WATCH (Default)

Use the /HIT qualifier to enable/disable process warning and termination.

If you specify /HIT=WATCH, HITMAN will write messages to the log file but will not warn or kill idle users. Using this test mode, you can verify that the behaviour is what you want, before turning HITMAN loose. This is HITMAN's default operating mode.

If you specify /HIT=WARN, HITMAN will warn but not kill idle users. This allows you to "nudge" users into activity. HITMAN will send warning and termination messages, but will not actually kill the process.

If you specify /HIT=HIT, HITMAN will warn and kill idle users. After you have satisfied yourself that HITMAN is performing properly, switch HITMAN into HIT mode.

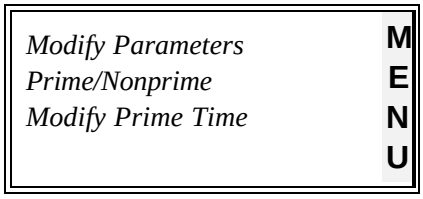
NOTE: If you are using prime and non-prime data files, remember that unless you specify the /FILE qualifier, HITMAN will update whichever permanent data file is currently in use.

Example

$ HITMAN/HIT=WARN

Record log messages and warn idle users but do not kill them.

/HOLIDAY=(d,...,d)



/HOLIDAY=(d,...,d)
/NOHOLIDAY=(d,...,d) (Default)

Use the /HOLIDAY qualifier to specify a date which is to be treated as non-prime regardless of the time of day or day of week. To remove a date from the holiday list, use /NOHOLIDAY=(date). You must specify the entire date including year, month and day. Any number of holidays may be defined.

Example

$ HITMAN/HOLIDAY=25-DEC-2003

Designate December 25 as a holiday.

$ HITMAN/NOHOLIDAY=(01-JUN-2004,11-NOV-2003)

Remove June 1 and November 11 of <u>last</u> year from holiday list.

/IDENTIFIER=(i,...,i)

/IDENTIFIER=(i,...,i) /FIRST_WARN=n -
/SECOND_WARN=n /WAIT=n
/IDENTIFIER=(i,...,i) /FIRST_WARN=n -
/SECOND_WARN=n /WAIT=n /CONNECT
/IDENTIFIER=(i,...,i) /PROTECT
/NOIDENTIFIER=(i,...,i)

Use the /IDENTIFIER qualifier to set wait and warning times for idle processes, connect times for users with specific identifiers or to protect users with specific identifiers. It is used in conjunction with the /WAIT, /FIRST_WARN, /SECOND_WARN, /CONNECT and /PROTECT qualifiers, and cannot be used with any other qualifiers. If you do not specify wait and warning times, the system default will be used. There is no maximum number of identifiers you can add to the idle and warn or connect lists.

Example

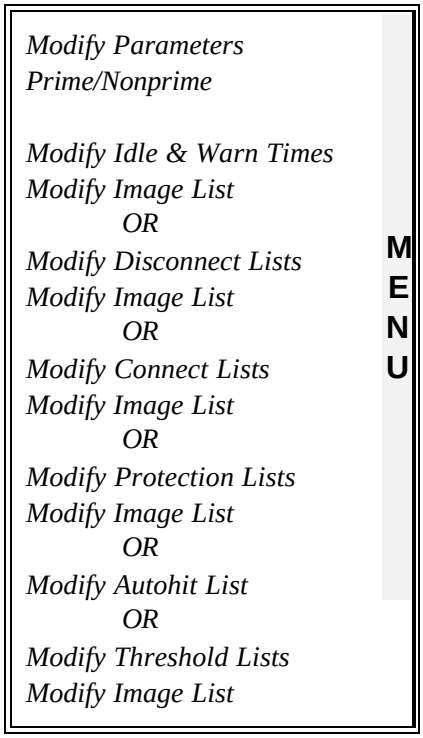
$ HITMAN /IDENTIFIER=ENGR /FIRST_WARN=55 /SECOND_WARN=60 /WAIT=90

Modify the wait and warning times for any user with the identifier ENGR.

$ HITMAN /IDENTIFIER=ENGR /PROTECT

Protect any user with the identifier ENGR.

/IMAGE=(i,...,i)



/IMAGE=(i,...,i) /PROTECT (Default = RTPAD, LTPAD)
/NOIMAGE=(i,...,i) /PROTECT

/IMAGE=(i,...,i) /DISCONNECT
/NOIMAGE=(i,...,i) /DISCONNECT (Default)

/IMAGE=(i,...,i) /FIRST_WARN=n -
/SECOND_WARN=n /WAIT=n
/NOIMAGE=(i,...,i)

/IMAGE=(i,...,i) /CONNECT /FIRST_WARN=n -
/SECOND_WARN=n /WAIT=n
/NOIMAGE=(i,...,i) /CONNECT

/IMAGE=(i,...,i) /AUTOHIT
/NOIMAGE=(i,...,i)/AUTOHIT (Default)

/IMAGE=(i,...,i) /BIO_THRESHOLD=n -
/CPU_THRESHOLD=n /DIO_THRESHOLD=n
/NOIMAGE=(i,...,i) /BIO_THRESHOLD (Default)

Use the /IMAGE qualifier to set wait and warning times or protect processes running a specific image. It is used in conjunction with the /WAIT, /FIRST_WARN, /SECOND_WARN, /CONNECT, /DISCONNECT and /PROTECT qualifiers, and is invalid if used with any other qualifiers. If you do not specify wait and warning times, the system default will be used. You can specify a wildcard to select a group of images. There is no maximum number of images you can add to any of these lists.

Example

$ HITMAN/IMAGE=MY_PROG/FIRST_WARN=55/SECOND_WARN=60/WAIT=90

Modify the wait and warning times for any user running the image MY_PROG.

$ HITMAN/IMAGE=MY_PROG/PROTECT

Protect any user running the image MY_PROG.

$ HITMAN /IMAGE=STARTREK /AUTOHIT /FILE=PRIME

Terminate any processes seen running STARTREK during prime time.

Use the /PROTECT/IMAGE qualifiers to specify images which are not to be hit regardless of inactivity. Processes running protected images will not be killed, no matter how much idle time they have. You can specify a wildcard to protect a group of images. HITMAN will automatically strip off the extension, for example, .EXE, and everything before the ']' (file_dev:[HITMAN.EXE]) on the entered image name. To remove an image from the protected list, use the /PROTECT/NOIMAGE qualifiers. By default, the DECNET images RTPAD and LTPAD are protected. There is no maximum number of images that can be protected.

NOTE: /PROTECT/IMAGE overrides the operations of /AUTHORIZED and /AUTOHIT. If you protect an image, HITMAN will not touch processes running that image regardless of its activities.

Example

$ HITMAN/PROTECT/IMAGE=EDT

Protect all users while they are executing EDT.EXE

$ HITMAN/PROTECT/NOIMAGE=LTPAD

Remove image LTPAD from image protect list.

With Version 8 of HITMAN you may now specify idle threshold values for specific images. Use this feature when you have determined that a particular image needs higher or lower thresholds than the system defaults for HITMAN to see it as idle, or active. When HITMAN sees a user running an image with image specific thresholds it will use those thresholds instead of the system defaults.

$ HITMAN /IMAGE=MENU /BIO_THRESHOLD=10 /CPU_THRESHOLD=4 -
/DIO_THRESHOLD=0

When a user is running the image MENU.EXE they can use up to 10 buffered I/Os and 40 (4 x 10) milliseconds of CPU and still be considered idle. MENU has some background usage because it displays a clock on the screen that is continually updated.

/IMAGE_NAME

Modify Parameters
Prime/Nonprime
Modify Flags & Values
Image Name
MENU

/IMAGE_NAME (Default)
/NOIMAGE_NAME

Use the /IMAGE_NAME qualifier to specify that all

interactive processes, running images or not, are eligible to be hit. If you specify /NOIMAGE_NAME, any process that is running an image, will be protected and only processes at the DCL level are considered eligible to be hit.

Example

$ HITMAN/NOIMAGE_NAME

Protect any process that is running an image.

/INTERVAL=n

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Flags & Values	**N**
Data Interval	**U**

/INTERVAL=n (Default = 1)

Use the /INTERVAL qualifier to set the number of minutes between data collections. The optimum interval for your system will depend on the number of users and how heavily loaded your system is. The minimum allowed is 1 minute. The maximum is 60 minutes. Normally, the difference in resource usage between 1 minute and 5 minutes is not significant. However, if you have many events defined or long wait or warning times, you may want to experiment with a longer data collection interval.

NOTE: When you increase the interval, HITMAN will automatically change wait and warning times to be multiples of it. You also may need to increase your threshold(s).

Example

$ HITMAN/INTERVAL=3

Set the data collection interval to 3 minutes.

/KILL

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Message Text	**N**
Authorize OR Autohit	**U**

/KILL (Default)
/NOKILL

Use the /KILL qualifier to terminate the process after the user receives this message. This qualifier works in conjunction with the /AUTHORIZE_MESSAGE and /AUTOHIT_MESSAGE qualifiers. Do not use it with /FIRST_MESSAGE, /SECOND_MESSAGE or /FINAL_MESSAGE. If you specify /NOKILL, HITMAN warns the user but does not kill them. You do not have to respecify the text to change this parameter.

Example

$ HITMAN/AUTHORIZE_MESSAGE/KILL

Terminate unauthorized users.

$ HITMAN/AUTOHIT_MESSAGE/NOKILL

Warn rather than kill users running an autohit image.

/LIST[=type_of_list/OUTPUT=filename]

/LIST[=type_of_list/OUTPUT=filename]
(Default
/LIST=ALL/OUTPUT=SYS$OUTPUT)

/LIST=ALL (Default)
/LIST=EVENTS
/LIST=FLAGS
/LIST=LISTS
/LIST=MESSAGES

Use the /LIST qualifier to list the parameters in the HITMAN database. This allows you to verify that the commands you have typed are being used, and find out what parameters HITMAN is actually using. If you omit the filename, the list is sent to the screen. If you have both prime and non-prime files, you must specify the /FILE qualifier to indicate which file you want to list; by default HITMAN will list the file currently in use.

Example

$ HITMAN/LIST=EVENTS/OUTPUT=E.LIS/FILE=NONPRIME

List the HITMAN non-prime events to the file E.LIS.

/LOG

/LOG (Default)
/NOLOG

Use the /LOG qualifier to close the old log file and open a new one. This allows you to produce reports from the log file. By default, the log file is called HITMAN_DAT:HITMAN_LOG_FILE.(nodename). The node name is added as the extension to the log file to avoid file access conflict errors at Alpha AXP and VAX cluster sites. To change the log file name, define the system logical name HITMAN_LOG_FILE as the file name you wish the log file to have. For example, the command:

$ DEFINE/SYSTEM HITMAN_LOG_FILE dua1:[fred]hit_log.dat

will cause the log file to be dua1:[fred]hit_log.dat.

To regularily open a new file you can create a batch procedure that does the HITMAN/LOG command and setup a HITMAN time event to submit it automatically every day, week or month.

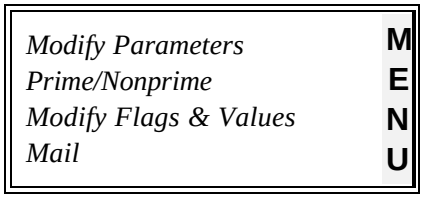
The /NOLOG qualifier causes HITMAN to close the current log file and NOT open a new one. To start a new log, type $ HITMAN/LOG. Normally, HITMAN creates a new log file each time it starts up. If you wish to use the same log file, define the logical HITMAN$LOG in the SYSTEM table. This will cause HITMAN to use an existing log file if there is one or create a new file if there is not one.

Example

$ HITMAN/LOG

Close the existing log file and start a new one.

/MAIL



/MAIL
/NOMAIL (Default)

Use the /MAIL qualifier to send mail messages to users, notifying them of their termination. If you do not want to send mail messages to users when they are terminated, specify HITMAN/NOMAIL.

Example

$ HITMAN/MAIL

Send mail messages to users

$ HITMAN/NOMAIL

Do not send mail messages when users are killed.

/MAX_IDLE=n

/MAX_IDLE=n (Default 999)

Use the /MAX_IDLE qualifier to specify the maximum number of minutes a user can specify for their own idle time. The qualifiers /MAX_IDLE and /MIN_IDLE set limits on the range of values users can specify for their own idle times. The minimum allowed is /MIN_IDLE. The maximum allowed is 999.

Example

$ HITMAN/MAX_IDLE=40

Set the maximum user-specified idle time to 40 minutes.

/MENU

/MENU

Use the /MENU qualifier to enter the HITMAN menu-driven screen interface.

Example

$ HITMAN/MENU

Enter the screen interface.

/MIN_IDLE=n

/MIN_IDLE=n (Default 2 x interval)

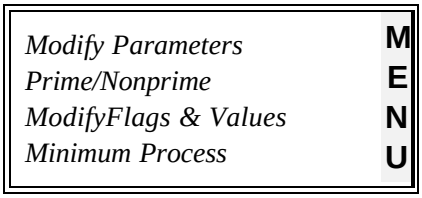
Use the /MIN_IDLE qualifier to specify the minimum number of minutes a user can specify for their own idle time. The qualifiers /MAX_IDLE and /MIN_IDLE set limits on the range of values users can specify for their own idle times. The minimum allowed is twice the data collection interval. The maximum allowed is /MAX_IDLE.

Example

$ HITMAN/MIN_IDLE=10

Set the minimum user-specified idle time to 10 minutes.

/MINIMUM_PROCESS=n



/MINIMUM_PROCESS=n
/NOMINIMUM_PROCESS (Default)

Use the /MINIMUM_PROCESS qualifier to specify the minimum number of nonsuspended processes that must be on the system before HITMAN will start terminating any processes. HITMAN will still remain on the system and collect data, but will check if there are the minimum number of processes specified before warning or killing any idle users. The minimum allowed is 0. The maximum allowed is 999. The maximum value would essentially turn HITMAN off except on extremely large systems. /NOMINIMUM_PROCESS turns off this feature.

Example

$ HITMAN/MINIMUM_PROCESS=30

Start data collection if there are 30 or more processes on the system.

/MULTIPLE

Modify Parameters	MENU
Prime/Nonprime	
Modify Flags & Values	
Multiple	

/MULTIPLE
/NOMULTIPLE (Default)

Use the /MULTIPLE qualifier to send multiple messages to the user before being terminated. If this feature is enabled, HITMAN will send a warning message every data collection interval (/INTERVAL) after the second warning has been sent. HITMAN uses the second message text as the multiple message text.

Example

$ HITMAN/MULTIPLE

Send multiple messages to idle users.

/NAME=image_name

Name cannot be modified through the menu.	MENU

/NAME=image_name
/NAME=HITMAN (Default)

Use the /NAME qualifier to change the process name of HITMAN. This hides HITMAN so that users cannot tell that it is running. Wildcards are not allowed. HITMAN does not have to be running and will start up with whatever name is in the permanent data file. The new process name can be up to 15 characters long and must start with an alphanumeric character. If you have changed the names of the permanent data or log files by reassigning the logicals, you cannot use this option. When you change the name, both prime and non-prime files are updated regardless of the /FILE qualifier.

Example

$ HITMAN/NAME=NEW_IMAGE

Set new process name to 'NEW_IMAGE'.

/OFF

Modify Parameters	MENU
Prime/Nonprime	
Modify Flags & Values	
Data Collection	

/OFF

Use the /ON and /OFF qualifiers to turn data collection on or off. When a user turns data collection off, the detached process remains on the system but does nothing. When it wakes up, it checks if data collection is turned on, if not, it hibernates again.

Example

$ HITMAN/OFF

Turn data collection off.

/ON

MENU
Modify Parameters
Prime/Nonprime
Modify Flags & Values
Data Collection

/ON (Default)

Use the /ON and /OFF qualifiers to turn data collection on or off. When a user turns data collection off, the detached process remains on the system but does nothing. When it wakes up, it checks if data collection is turned on, if not, it hibernates again.

Example

$ HITMAN/ON

Turn data collection on.

/OPERATOR_CLASS=operator_class

MENU
Modify Parameters
Prime/Nonprime
Modify Flags & Values
Operator Class

/OPERATOR_CLASS=SECURITY (Default)
/OPERATOR_CLASS=CENTRAL
/OPERATOR_CLASS=OPER1 - OPER12

Use the /OPERATOR_CLASS qualifier to specify the class of operator that you wish to send error and other messages to. To specify an operator class, you must have OPER privilege. The available operator classes are : SECURITY, CENTRAL, and OPER1 through OPER12. To receive the error messages at your terminal, type: REPLY/ENABLE = operator_class.

NOTE: Messages will only be sent to the operator class if /CONSOLE is specified.

Example

$ HITMAN/OPERATOR_CLASS = OPER9

Enable OPER9 as the operator class to which HITMAN will send all its operator messages.

/OUTPUT=file_name

MENU
Output is normally directed to the screen when in the menu.

/OUTPUT=file_name (Default = SYS$OUTPUT)

Use the /OUTPUT qualifier to specify where output is to be sent.

Example

$ HITMAN/REPORT=HITMAN_LOG_FILE.NODE1/OUTPUT=LOG_REP.N1

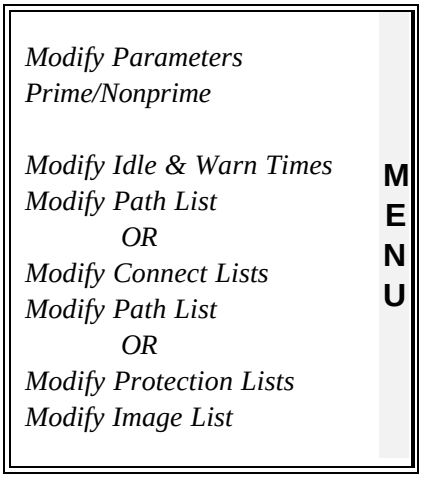
Read HITMAN_LOG_FILE.NODE1 and create a report called LOG_REP.N1. Both of these files will be in the HITMAN_DAT directory.

/PATH=path

/PATH=(p,...,p) /PROTECT
/NOPATH=(p,...p) /PROTECT (Default)

/PATH=(s,...,s) /FIRST_WARN=n -
/SECOND_WARN=n /WAIT=n
/NOPATH=(s,...,s) (Default)

/PATH=(s,...,s) /CONNECT /FIRST_WARN=n -
/SECOND_WARN=n /WAIT=n
/NOPATH=(s,...,s) /CONNECT (Default)



Use the /PATH qualifier to set wait and warning time, connect wait and warning times or to protect processes running a specific copy of a specific image. This qualifier is new with Version 10 of Hitman and was added in response to customer requests for a more specific way of protecting images; with V9 only the name portion was used and all copies of the same image would be affected. Acommon use of this qualifier would be during testing to protect a production copy of an image without protecting the development copy. It is used in conjunction with the /WAIT, /FIRST_WARN, /SECOND_WARN, /CONNECT and /PROTECT qualifiers and is invalid if used with any other qualifiers. If you do not specify wait and warning times, the system default will be used. There is no maximum number of paths that can be protected or have special times defined.

A path is a complete file specification without the version number. When using paths it is important to know the path as it appears to the system from within an application since all files on OpenVMS systems may have more than 1 valid path. For example each of the following is a valid path for the EDT editor; as a result it could be necessary to add more than 1 path entry to guarantee that users in EDT are protected.

DKA0:[SYS0.SYSEXE]EDT.EXE
SWIFT$DKA0:[SYS0.SYSEXE]EDT.EXE
SYS$SYSTEM:EDT.EXE
1DKA0:[SYS0.SYSEXE]EDT.EXE

To determine which path entry is likely to work properly we suggest having someone enter the image using your normal method (in this case by entering EDIT/EDT file). Use the show

process/continuous/id=pid command to see the process information for that user; the path will be displayed at the bottom of the screen. Using this path, minus the version number, should work. If users have more than one way of entering a particular image be sure to verify that the path name is the same each way.

Example

$ hitman /path=node$dka0:[sys0.sysexe]backup.exe /protect

This command will protect users running backup without protecting users in other applications that might have an executable named backup.exe

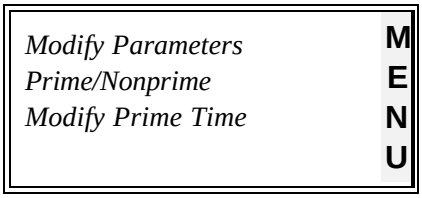
$ hitman /path=node$dka0:[ourapps.exe]ourapp.exe /protect
$ hitman /path=node$dka1:[dev.ourapps.exe]ourapp.exe -
/first_warn=5 /second_warn=7 /wait=10

This combination of commands protects users in the production copy of ourapp.exe while giving users running the development copy significantly lower idle time limits. Configuring an application this way would allow a site to test a new version of their software with Hitman to make sure process termination is working properly WITHOUT affecting your product users.

$ hitman /nopath=node$dka0:[ourapps.exe]ourapp.exe /protect

Removes this particular path from the protection list after testing has been completed.

/PRIME_END=time



/PRIME_END=time
/NOPRIME_END (Default)

Use the /PRIME_END qualifier to specify the time at which to start using the nonprime data file (PERM_DATA_NONPRIME.DAT). The prime end time must be greater than the prime start time. This time is based on a 24-hour clock. For example, you enter 11:00 p.m. as 23:00. When HITMAN sees that the current system time is equal to or greater than the non-prime start time, it reads the non-prime permanent data file and uses its qualifiers until the prime start time the next day. Prime day, start and end qualifiers must be used together for this feature to work. When you enter these qualifiers, both the prime and non-prime permanent data files are updated regardless of the /FILE qualifier setting.

Example

$ HITMAN/DAY=WEEKDAYS/PRIME_START=09:00/PRIME_END=16:00

On weekdays (Monday - Friday), prime time starts at 9:00 a.m. and ends at 4:00 p.m.

/PRIME_START=time

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Prime Time	**N**
	U

/PRIME_START=time
/NOPRIME_START (Default)

Use the /PRIME_START qualifier to specify the time at which to start using the prime permanent data file (PERM_DATA_PRIME.DAT). The prime start time must be less than the prime end time. The time is based on a 24-hour clock. For example, you enter 5:00 p.m. as 17:00. When HITMAN sees that the current system time is equal to or greater than the prime start time, it reads the prime permanent data file and uses its qualifiers until the prime end time. The period outside the prime interval is considered non-prime and HITMAN uses the nonprime permanent data file. Prime day, start and end qualifiers must be used together for this feature to work. When you enter these qualifiers, both the prime and non-prime permanent data files are updated regardless of the /FILE qualifier setting.

Example

$ HITMAN/DAY=TUESDAY/PRIME_START=8:00/PRIME_END=17:00

On Tuesdays, prime time starts at 8:00 a.m. and ends at 5:00 p.m.

/PRIORITY=n

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Flags & Values	**N**
Priority	**U**

/PRIORITY=n (Default = 6)

Use the /PRIORITY qualifier to specify the base priority at which the detached process starts up. A higher priority ensures more accurate data collection. The minimum allowed is 4. The maximum allowed is 10. In order to set a base priority higher than your own, you must have ALTPRI privilege. This qualifier must be set to the desired priority before HITMAN is started.

Example

$ HITMAN/PRIORITY=8

Set base priority to 8.

/PROCESS=(p,...,p)/PROTECT

Modify Parameters	
Prime/Nonprime	
Modify Protection Lists	
Modify Process List	**M**
OR	**E**
Modify Idle & Warn Times	**N**
Modify Process List	**U**
OR	
Modify Connect Times	
Modify Process List	

/PROCESS=(p,...,p)
/NOPROCESS=(p,...,p)

/PROCESS=(s,...,s) /FIRST_WARN=n -
/SECOND_WARN=n /WAIT=n
/NOPROCESS=(s,...,s) (Default)

/PROCESS=(s,...,s) /CONNECT /FIRST_WARN=n -
/SECOND_WARN=n /WAIT=n
/NOPROCESS=(s,...,s) /CONNECT (Default)

Use the /PROTECT/PROCESS qualifiers to specify process names which are not to be hit regardless of inactivity. Protected processes will not be killed, no matter how much idle time they have. You must enclose the process name in double quotes. Remember, process names can contain both upper and lower case characters; the process name must match the case of the appropriate process exactly. You can specify a wildcard to protect a group of processes. To remove a process name from the protected list, use the /PROTECT/NOPROCESS qualifiers. There are no defaults for process name protection. All interactive processes are eligible to be killed. There is no maximum number of process names that can be protected.

Use /PROCESS with /First_warn, /Second_warn and /Wait to set special idle times by process name or with /CONNECT to set special connect time limits for processes by name.

NOTE: /PROTECT/PROCESS overrides the operations of /AUTHORIZED and /AUTOHIT. If you protect a process, HITMAN will not touch it regardless of its activities.

Example

$ HITMAN/PROTECT/PROCESS="MAN*"

Protect any processes with a name starting with "MAN".

$ hitman /process="My Process" /first_warn=15 /nosecond_warn /wait=18

Any process with the process name "My Process" will be monitored with special time limits and not given a second warning if it is detected as idle.

/PROTECT

Modify Parameters
Prime/Nonprime
Modify Protection Lists
MENU

/PROTECT
/PROTECT/NOkeyword=(i,...,i)

Use the /PROTECT qualifier in conjunction with /USER, /UIC, /IMAGE, /PROCESS, /SERVER, /TERMINAL, /IDENTIFIER and /PATH qualifiers to specify processes which are not to be hit regardless of inactivity. Protected processes will not be killed, no matter how much idle time they have. Excluding identifiers, you can specify a wildcard to protect a group of processes. To remove a entry from the protected list, use /PROTECT/NO... whatever type of entry you are trying to remove (NOUSER, NOIMAGE, etc.). All of the currently checked processes specified in /TYPE are eligible to be killed unless they are protected. There is no maximum number of protection entries in any of these tables.

NOTE: /PROTECT overrides the operations of /AUTHORIZED and /AUTOHIT. If you

protect a process, HITMAN will not touch it regardless of its activities.

Example

$ HITMAN/PROTECT/PROCESS="MAN*"

Protect any processes with a name starting with "MAN".

$ HITMAN/PROTECT/USER=(FRED,LARRY,CURLY,MOE)

Protect any user on the specified list.

/REPORT[=logfile]/OUTPUT=reportname

/REPORT[=filename]

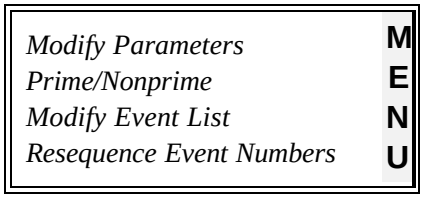
Use the /REPORT qualifier to generate a log file report from DCL. This command **does not** provide selection criteria to generate the report. It generates a report based on the entire log file. If you wish to generate a report with selection criteria, use the ANALYZE LOG FILE option in the menu interface. If you do not specify the log file name, HITMAN uses the default name LOG_REPORT.LIS. If you do not specify the report file name, HITMAN uses the default name HITMAN_LOG_FILE.{node_name}. Using the /ASCII qualifier will generate a delimited ASCII text file which may be imported into many popular spreadsheets, databases and graphics packages.

Example

$ HITMAN /REPORT=HITMAN_LOG_FILE.NODE1 -
/OUTPUT=LOG_REP.NODE1

Read the HITMAN_LOG_FILE.NODE1 log file and create a report called LOG_REP.NODE1. Both of these files will be in the HITMAN_DAT directory.

/RESEQUENCE_EVENTS



/RESEQUENCE_EVENTS

Use the resequence events qualifier to reset event IDs in sequence. The new event IDs will start at 2 and increment by 1 for as many events as are currently defined. Anytime events are deleted their id numbers become unused since adding events always adds them to the end of the list unless the ID has been specified; this command resequences the Ids and fills in any gaps.

Example

$ HITMAN /RESEQUENCE_EVENTS /FILE=BOTH

Resequence the events in both the prime and nonprime files.

/SECOND_MESSAGE="text"

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Message Text	**N**
Modify Second Warning Text	**U**

/SECOND_MESSAGE="text"
/NOSECOND_MESSAGE
(Default= "** HITMAN Warning, You have been ?M? for ?I? minutes **?L?")

Use the /SECOND_MESSAGE qualifier to change the text of the second warning message. Like normal OpenVMS DCL commands, the text string must be enclosed in double quotes. If no text is provided, the default warning message is used. Used in conjunction with the /MULTIPLE qualifier the second warning message will be sent when the user has been idle (or connected if applicable) /SECOND_WARN minutes; it will then be sent every data collection interval after that until they resume activity or they are hit. Messages may be up to 200 characters long AFTER variable substitution. We recommend that second warning messages always end with ?L? which causes a carriage return and line feed to prevent the final message being written overtop of the second message.

Example

$ HITMAN/SECOND_MESSAGE="LACK OF ACTIVITY WARNING!!?L?"

Define site-specific text for the second warning message, end with a carriage return and line feed.

$ HITMAN/SECOND_MESSAGE

Use the default second warning message.

$ HITMAN/NOSECOND_MESSAGE

Do not send the second warning message.

/SECOND_WARN=n

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Idle & Warn Times	**N**
OR Modify Connect Lists	**U**

/SECOND_WARN=n (Default = 25)
/NOSECOND_WARN

Use the /SECOND_WARN qualifier to specify the time to send the second warning message to idle or connected users. The minimum allowed is /FIRST_WARN. The maximum allowed is /WAIT. This qualifier is used with the /FIRST_WARN and /WAIT qualifiers. It must be greater than the /FIRST_WARN time and less than the /WAIT time. It is possible to configure Hitman for no second warning message; in this case use /FIRST_WARN=x /NOSECOND_WARN /WAIT=x with the wait time being greater

than the first_warn time.

Example

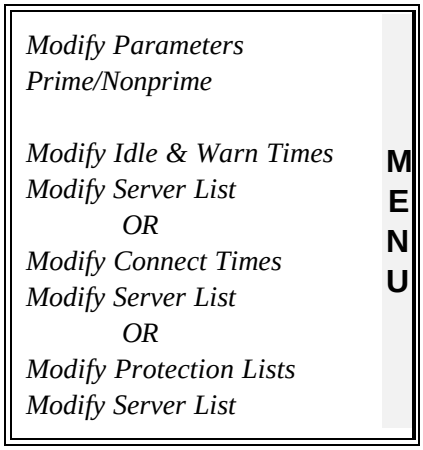
$ HITMAN/SECOND_WARN=35

Send the second warning message after 35 minutes of idle time.

$ HITMAN /SECOND_WARN=35 /WAIT=45 /TERMINAL=TX* /CONNECT

Send the second warning message after 35 minutes of connect time to anyone who is connected through a TX terminal port.

/SERVER=(s,...,s)



/SERVER=(s,...,s) /PROTECT
/NOSERVER=(s,...,s) /PROTECT (Default)

/SERVER=(s,...,s) /FIRST_WARN=n -
/SECOND_WARN=n /WAIT=n
/NOSERVER=(s,...,s) (Default)

/SERVER=(s,...,s) /CONNECT /FIRST_WARN=n -
/SECOND_WARN=n /WAIT=n
/NOSERVER=(s,...,s) /CONNECT (Default)

Use the /SERVER qualifier to set wait or warning times (or both) for a specific server ID. It is used in conjunction with the /WAIT, /FIRST_WARN, /SECOND_WARN, /CONNECT and /PROTECT qualifiers. If you do not specify values for wait or warning times, the default for the system is used. You can specify a wildcard to set wait, warning or connect times for a group of server IDs. There is no maximum number of servers you can define wait and warning times or connect times for. The server string frequently consists of two parts separated by the "/" character; the server name and the port name.

See the chapter "Warning Idle Users" for the order of precedence of wait and warning time.

Example

$ HITMAN /SERVER="DSV200 /PORT-*" /FIRST_WARN=10 /WAIT=25

Give a warning to all users logging in through the server named "DVS200".

$ HITMAN /NOSERVER="DSV200/PORT-*" /CONNECT

Remove server DSV200 (all ports) special connect time limits.

Use the /PROTECT/SERVER qualifiers to specify server IDs which are not to be hit regardless of inactivity. Protected server IDs will not be killed, no matter how much idle time they have. You

must enclose the server ID in double quotes. You can get your server ID by typing $ SHOW USERS/FULL. Your server ID is shown beside your terminal ID in brackets "()". You can specify a wildcard to protect a group of servers or ports. If you specify a wildcard for the server, all ports on that server will be protected. To remove a server ID from the protected list, use the /PROTECT/NOSERVER qualifiers. There are no defaults for server protection. All interactive processes using a terminal server are eligible to be killed. There is no maximum number of server entries you can define.

NOTE: /PROTECT/SERVER overrides the operations of /AUTHORIZED and /AUTOHIT. If you protect a server ID, HITMAN will not touch it regardless of its activities.

Example

$ HITMAN /PROTECT /SERVER="LAT_12345678/PORT_4"

Protect any processes logged in to this port from being terminated.

/SPECIAL_EXIT

Not definable through the menu	**MENU**

/SPECIAL_EXIT=(path=image_path, job=job_name, -
[queue=batch_queue,-] *Default=sys$batch*
[username=submit_username,-]
Default=current username Hitman is running under
[node=check_node,] *Default=all nodes*
[debug=true or false]) *Default=False*

/NOSPECIAL_EXIT=(path=image_path)

With Version 10 of Hitman it is no possible to define any number of special exit handlers which get processed instead of Hitman's default process termination module. Each special exit handler must include the path of the image which, when an idle user is found in that image, will have a special exit procedure submitted to batch and a job, the DCL command procedure which will be submitted. Optionally the queue for the job to be submitted can be specified; if not included the default is SYS$BATCH. A username to submit the job under can be specified; if not included the default is whatever username the current Hitman detached process is running under. A node on which this check should be made can be specified; the default is any node that this special exit is defined on. Finally the disposition of the log file for the batch job can be controlled; by default the log is not kept, however, setting the debug flag to true will change this action.

When a process is found that is idle in the image specified by the path clause the special exit procedure will be submitted to batch IF the /user_exit qualifier has been used to turn on special exit checking. At this point the process will be flagged to be ignored by Hitman and no subsequent action will be taken by Hitman. If it is appropriate to kill the idle process the special exit batch job must include the stop process/id command to terminate the idle process.

The special exit procedure will be submitted with the following parameters:

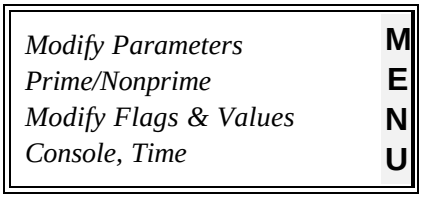
Parameter	Value passed in this parameter
P1	The hexidecimal process ID of idle process (PID) as an 8-byte string.
P2	Number of minutes idle
P3	Explanation for termination; normally “Exceeded Allowed Idle Time”
P4	Username of idle process
P5	Process name of idle process
P6	Path of image being run by idle process
P7	not used
P8	not used

Using these parameters it is normally possible to write a DCL procedure that can do almost any required action. Note: while process IDs (PID) are normally 8 hexadecimal digits on single node systems they maybe left padded with blanks; in this situation we recommend using the lexical function f$edit with the “COLLAPSE” edit function to remove these spaces before trying to use the P1 (PID) parameter.

A more detailed explanation of how to use special exit procedures can be found in the section on special user exit procedures on page 148.

If /nouser_exit is in effect the process would be terminated by Hitman’s normal procedures.

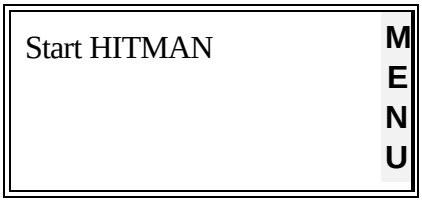
/STAMP (Obsolete)



/STAMP

This qualifier has been replaced by the enhanced /CONSOLE command.

/START



/START

Use the /START qualifier to start HITMAN's detached

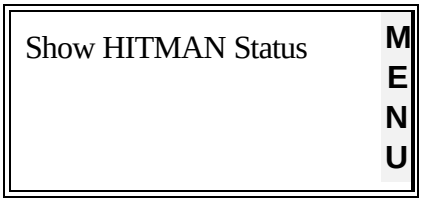
process. Use this qualifier on a separate line without any other qualifiers.

Example

$ HITMAN/START

Start the detached process.

/STATUS



/STATUS

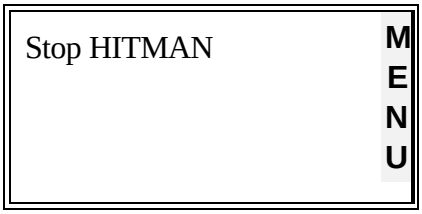
Use the /STATUS qualifier to see a brief display of what HITMAN is currently doing. Use this qualifier to check that HITMAN is running, what permanent data file it is using, what version of HITMAN you have, what mode HITMAN is in, etc.

Example

$ HITMAN/STATUS

Show the current HITMAN status.

/STOP



/STOP

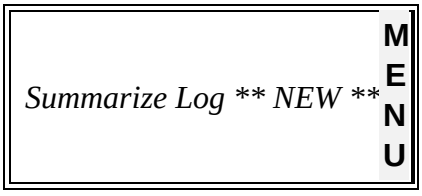
Use the /STOP qualifier to stop HITMAN's detached process. Use this qualifier on a separate line without any other qualifiers.

Example

$ HITMAN/STOP

Stop the HITMAN detached process.

/SUMMARY



/SUMMARY[=logfile] [/output=reportfile]
/NOSUMMARY (Default)

Use the /SUMMARY qualifier to generate a summary report of a Hitman log file. If no log file is specified the current log file will be summarized. The default reportfile name is hitman_dat:log_summary.lis. The summary report is broken down by the

individual log file record types such as 1st warning for idle time exceeded, 2nd warning for idle time exceeded or process terminated because idle time was exceeded. Under each type all the users that were acted on that way are listed with the number of times. For example:

```
     Idle First Warning Messages
-------------------------------------------------------------
          User1                                6
          User4                                 2
-------------------------------------------------------------
     Idle Process Terminations
-------------------------------------------------------------
          User1                                 4
```

You can tell from this summary that anytime User4 received a first warning they either logged out because they were finished working or resumed using the system; they were warned but never killed for being idle.

/SWAP

MENU
Modify Parameters
Prime/Nonprime
Modify Flags & Values
Swap

/SWAP (Default)
/NOSWAP

Use the /SWAP qualifier to tell HITMAN to swap in processes to get information. If your system runs with many outswapped processes, using the /NOSWAP qualifier may save system resources by preventing HITMAN from swapping processes in. However, since HITMAN does not have process information, it ignores the process.

Example

$ HITMAN/NOSWAP

Do not swap in outswapped processes to collect process information.

/SYS_PROTECT

MENU
Modify Parameters
Prime/Nonprime
Modify Flags & Values
System Protect

/SYS_PROTECT
/NOSYS_PROTECT (Default)

Use the /SYS_PROTECT qualifier to protect system processes from being terminated. All group UIC's that are less than or equal to MAXSYSGROUP (usually 8) will be protected when this qualifier is specified. Please note that OpenVMS system processes are automatically and unconditionally protected. The setting of this qualifier does not affect that. **NOTE:** If you decide to include more than interactive processes using the /TYPE feature of HITMAN, we **STRONGLY** recommend you use /SYS_PROTECT.

Example

$ HITMAN/SYS_PROTECT

Protect all system processes.

/TABLE_SIZE=value

/TABLE_SIZE=value (Nodefault)

Use the /TABLE_SIZE qualifier to control how many processes the Hitman detached process can monitor. ***This qualifier is not dynamic; if the value is changed you must stop and restart Hitman for the change to take effect.***

Values specified can be in the following ranges:

Range	Hitman's detached process will monitor
0	Up to the number of processes specified by the SYSGEN parameter MAXPROCESSCNT.
From 1 to 99	Up to the specified percentage of the value of the SYSGEN parameter MAXPROCESSCNT.
100 or greater	The exact number of processes specified.

NOTES: In clusters where the data file is shared by multiple nodes it is strongly recommended that /table_size be set to 0 since each node will probably have a different value for the MAXPROCESSCNT sysgen parameter.

Care must be taken when using a value between 1 and 99 to set the table size to a percentage of MAXPROCESSCNT you should keep in mind that if the total number of processes exceeds the calculated percentage Hitman will generate a console message and stop running. A percentage value can be useful on systems, like workstations, where the maximum number of processes is significantly less than the MAXPROCESSCNT value. Setting the /table_size to a specific value can result in the same error.

Modify Parameters
Prime/Nonprime

Modify Idle & Warn Times
Modify Terminal List
OR
Modify Connect Times
Modify Terminal List
OR
Modify Protect Lists
Modify Terminal List

MENU

/TERMINAL=(t,...,t)

/TERMINAL=(t,...,t) /PROTECT (Default = OPA0:)
/NOTERMINAL=(t,...,t) /PROTECT

/TERMINAL=t /FIRST_WARN=n /SECOND_WARN=n -
/WAIT=n
/NOTERMINAL=t (Default)

/TERMINAL=t /CONNECT /FIRST_WARN=n -
/SECOND_WARN=n /WAIT=n
/NOTERMINAL=t /CONNECT (Default)

Use the /TERMINAL qualifier to set wait or warning times (or both) for a specific terminal. It is used in conjunction with the /WAIT, /FIRST_WARN, /SECOND_WARN, /CONNECT and /PROTECT qualifiers. If you do not specify values for wait or warning times the default for the system is used. You can specify a wildcard to set wait or warning times for a group of terminals. There is no maximum number of terminals you can define wait and warn or connect time limits for.

See the chapter "Warning Idle Users" for the order of precedence of wait and warning time.

/TERMINAL is not useful if you are using virtual terminals since the terminal number is not related to the location of the terminal. Virtual terminals show a device of "VTA####" in the $ SHOW USERS/FULL command.

Example

$ HITMAN /TERMINAL=TTA* /FIRST_WARN=10 /WAIT=25

Give one warning to all "TTA*" terminals and kill them after 25 minutes of inactivity.

$ HITMAN /NOTERMINAL="TTA1: /CONNECT"

Remove terminal TTA1 special connect time limits.

Use the /PROTECT/TERMINAL qualifier to specify terminals which are not to be hit regardless of inactivity. Protected terminals will not be killed even if they're idle or exceed the allowable connect time. You can specify a wildcard to protect a group of terminals. To remove a terminal from the protected list, use the /PROTECT/NOTERMINAL qualifiers. By default, the operator console OPA0: is protected. There is no maximum number of terminals you can protect.

/PROTECT/TERMINAL is not useful if you are using virtual terminals since the terminal number is not related to the location of the terminal.

NOTE: /PROTECT/TERMINAL overrides the operations of /AUTHORIZED and /AUTOHIT. If you protect a terminal, HITMAN will not touch it regardless of its activities.

Example

$ HITMAN /PROTECT /TERMINAL=TTA1:

Protect terminal "TTA1:" from being killed.

$ HITMAN /PROTECT /NOTERMINAL=OPA0:

Remove terminal "OPA0:" from the terminal protect list.

/TERMINATION=termination_type

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Flags & Values	**N**
Termination	**U**

/TERMINATION=BOTH (Default)
/TERMINATION=FORCE_EXIT
/TERMINATION=DELETE_PROCESS

Use the /TERMINATION qualifier to specify the procedure HITMAN will use to kill a process. The default is BOTH, which has HITMAN perform a force exit ($FORCEX) to cause an image rundown (if the process is running an image), followed by process deletion ($DELPRC). If you specify FORCE_EXIT, the force exit and image rundown will be executed but the process will not be deleted. For current OpenVMS releases, the default of BOTH is the best choice; it allows HITMAN to perform multiple forced exits if necessary.

The force exit termination type may be useful for systems with captive accounts.

Example

$ HITMAN/TERMINATION=FORCE_EXIT

Do a force exit and image rundown when HITMAN terminates a process but do not delete the process.

/TIMEOUT=n

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Disconnect List	**N**
User OR Image Disconnect	**U**

/TIMEOUT=n
(Default=30)

Use the /TIMEOUT qualifier to specify the number of minutes a disconnected process is to remain on the system before being deleted. The minimum allowed is 1. The maximum allowed is 999. Use this qualifier in conjunction with the /DISCONNECT, /IMAGE and /USER qualifiers.

Example

$ HITMAN/USER=LARRY/DISCONNECT/TIMEOUT=12

Allow LARRY's process to remain on the system for 12 minutes after it is disconnected.

/TYPE

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Flags & Values	**N**
Type	**U**

/TYPE=(BATCH, DETACHED, DIALUP,-
INTERACTIVE, NETWORK)

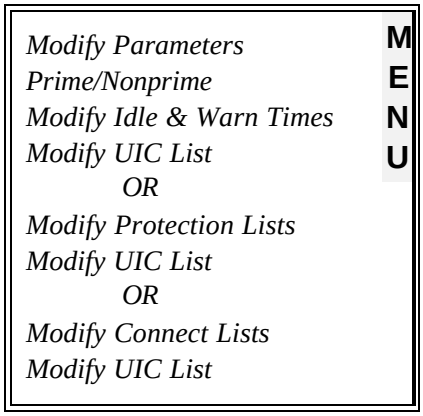
Use the /TYPE qualifier to specify which types of processes are eligible to be hit. By default, only INTERACTIVE processes are eligible. This qualifier allows you to define which processes you want to terminate. **NOTE:** This is a powerful qualifier which can cause problems if it is used inappropriately. Specify /HIT=WATCH when you experiment with this qualifier. We STRONGLY recommend that /SYS_PROTECT be in effect anytime you are considering processes which are not interactive!

Example

$ HITMAN/TYPE=(DIALUP)

Only DIALUP processes are eligible to be terminated.

/UIC=u



/UIC=("u",...,"u") /PROTECT
/NOUIC=("u",...,"u") /PROTECT (Default)

/UIC=u /FIRST_WARN=n /SECOND_WARN=n -
/WAIT=n
/NOUIC=u (Default)

/UIC=u /CONNECT /FIRST_WARN=n -
/SECOND_WARN=n -/WAIT=n
/NOUIC=u /CONNECT (Default)

Use the /UIC qualifier to set wait or warning times (or both) for a specific UIC group. You must enclose the UIC in double quotes. It is used in conjunction with the /WAIT, /FIRST_WARN, /SECOND_WARN, /CONNECT and /PROTECT qualifiers. If you do not specify values for wait, warning or connect times the system default will be used. You can specify a wildcard to set wait or warning times for a group of UIC's. There is no maximum number of UIC entries you can add to the idle and warn or connect lists. See the chapter "Warning Idle Users" for the order of precedence of wait and warning time. When specifying a list of UIC's be sure to enclose each UIC in double quotes or the DCL interpreter will not be able to distinguish the individual UICs properly.

NOTE: Because of the DCL Syntax rules, you may not specify /FILE=BOTH with the /UIC qualifier. Please add UICs to both files if necessary.

Example

$ HITMAN/UIC="[100,*]"/FIRST_WARN=0/SECOND_WARN=0/WAIT=120

The file transfer accounts are in group 100. Give no warnings, but do not kick them off until they have been idle for 120 minutes.

$ HITMAN/UIC="[SAIGA]"/FIRST_WARN=2/SECOND_WARN=13/WAIT=15

Warn group SAIGA at 2 minutes and 13 minutes and terminate them at 15 minutes.

$ HITMAN /NOUIC="[34,3]" /FILE=PRIME
$ HITMAN /NOUIC="[34,3]" /FILE=NONPRIME

Remove UIC group [34,3] special wait and warning times from the prime and nonprime files. Because of DCL syntax rules and UICs containing commas, ",", this command will not work with /FILE=BOTH.

Use the /PROTECT/UIC qualifiers to specify UIC's which are not to be hit regardless of inactivity. Protected UIC's will not be killed, no matter how much idle or connect time they have. You can specify a wildcard in group or member to protect a set of UIC's. You must enclose the UIC in double quotes. To remove a UIC from the protected list, use the /PROTECT/NOUIC qualifiers. There are no defaults for UIC protection. All interactive processes are eligible to be killed. There is no maximum number of UICs you can protect.

NOTE: /PROTECT/UIC overrides the operations of /AUTHORIZED and /AUTOHIT. If you protect a UIC, HITMAN will not touch it regardless of its activities.

Example

$ HITMAN/PROTECT/UIC="[410,*]"

Protect group 410 from being killed no matter how inactive they are.

$ HITMAN/PROTECT/NOUIC="[SAIGA]"

Remove group SAIGA from the UIC protect list.

/UPDATE_INTERVAL=n

/UPDATE_INTERVAL=n (Default = 15)

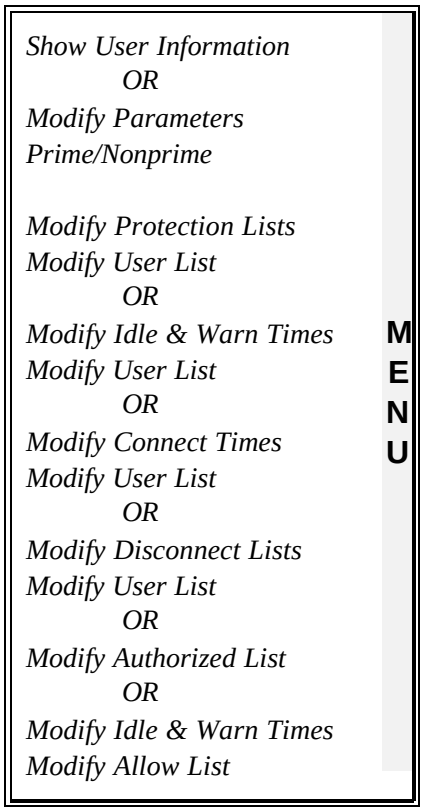
Use the /UPDATE_INTERVAL qualifier to specify the interval between log file updates. The log file is closed and re-opened every Update Interval so you can take a look at what HITMAN has done. The minimum allowed is /INTERVAL. The maximum allowed is 60 minutes.

Example

$ HITMAN/UPDATE_INTERVAL=20

Update the Log File every 20 minutes.

/USER=(u,...,u)



/AUTHORIZED /USER=(u,...,u)
/AUTHORIZED /NOUSER=(u,...,u)

/DUMP /USER=u

/PROTECT /USER=(u,...,u)
/PROTECT /NOUSER=(u,...,u) (Default - none)

/USER=(u,...,u) /ALLOW
/NOUSER=(u,...,u) /ALLOW (Default - none)

/USER=(u,...,u) /DISCONNECT
/NOUSER=(u,...,u) /DISCONNECT (Default - none)

/USER=(u,...,u) /FIRST_WARN=n /SECOND_WARN=n-
/WAIT=n
/NOUSER=(u,...,u) (Default - no special user times)

/USER=(u,...,u) /CONNECT /FIRST_WARN=n-
/SECOND_WARN=n /WAIT=n
/NOUSER=(u,...,u) /CONNECT (Default - no special user connect times)

Use the /USER qualifier to set wait, warning or connect times for a specific user. It is used in conjunction with the /WAIT, /FIRST_WARN, /SECOND_WARN, /PROTECT, /DUMP, /AUTHORIZE, /CONNECT and /DISCONNECT qualifiers, and is invalid if used with any other qualifiers. If you do not specify a qualifier, the system default is used. You can specify a wildcard to set wait, warning or connect times for a group of users. There is no maximum number of users you can add to any of these lists. You may only specify one user for each dump performed.

See the chapter "Warn idle users" for the order of precedence of wait and warning time.

Example

$ HITMAN/USER=LARRY/FIRST_WARN=5/WAIT=25

Set user LARRY's first warning time to 5 minutes and hit time to 25 minutes.

$ HITMAN /NOUSER=(LARRY,MOE,SHEMP) /CONNECT

Remove the special connect times defined for LARRY, MOE and SHEMP.

Use the /PROTECT/USER= qualifiers to specify users who are not to be hit regardless of inactivity. Protected users will not be killed, no matter how much idle or connect time they have. You can specify a wildcard to protect a group of users. To remove a username from the protected

list, use the /PROTECT/NOUSER= qualifiers. There are no defaults for protection. All interactive processes are eligible to be killed. There is no maximum number of users that can be protcted.

NOTE: /PROTECT/USER= overrides the operations of /AUTHORIZED and /AUTOHIT. If you protect a user, HITMAN will not touch them regardless of their activities.

Example

$ HITMAN/PROTECT/USER=LARRY

Protect user LARRY from being hit.

$ HITMAN/PROTECT/USER=ENG*

Protect all usernames starting with the characters "ENG".

$ HITMAN/PROTECT/NOUSER=MOE

Remove user MOE from the protected list. If MOE is inactive, he will be hit.

The /USER qualifier is also used with /ALLOW to add users to the list of users who are allowed to set their own idle and warning times. It is used with /AUTHORIZED to add users to the list of users who are authorized for special privileges in the ALL category. It is used with /DUMP to specify which user HITMAN should dump all its table information to the dump report for. See the /ALLOW, /AUTHORIZED and /DUMP qualifiers for a more detailed explanation and examples.

/USER_EXIT

/USER_EXIT
/NOUSER_EXIT (Default)

Use the /USER_EXIT qualifier to take site-specific actions instead of, or in addition to, HITMAN's normal actions. Please refer to the chapter "Setting up your own idle process procedures" before using this qualifier.

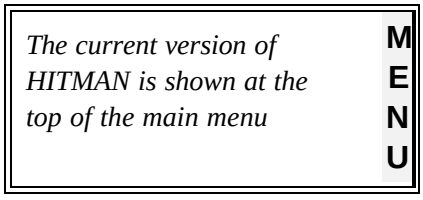
Example

$ HITMAN/USER_EXIT

Execute the user exit routine.

/VERSION

Use this qualifier to check the version of HITMAN you are running. The /VERSION qualifier must be issued as a separate command.



/VERSION

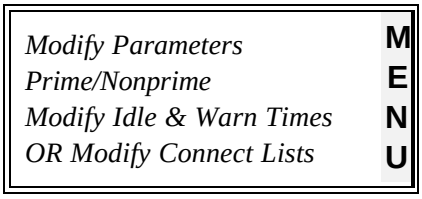
Example

$ HITMAN/VERSION

Display the current version of HITMAN you are running in the message:

%HITMAN-I-VERSION, Saiga Systems - HITMAN 10.0.0

/WAIT=n



/WAIT=n (Default = 30)

Use the /WAIT qualifier to specify the number of minutes of idle or connect time a process is allowed before it is hit. The minimum allowed is twice the data collection interval or /FIRST_WARN or /SECOND_WARN. The maximum allowed is 999 minutes. The /WAIT qualifier can be used alone to specify system wide idle times or in conjunction with other qualifiers.

If it is used in conjunction with the /DIALUP qualifier, it defines the number of minutes that a dialup process must be idle, or connected if used with the /CONNECT qualifier, before it will be killed.

If it is used in conjunction with the /UIC qualifier, it defines the number of minutes that a process in that UIC group must be idle, or connected if used with the /CONNECT qualifier, before it will be killed.

If it is used in conjunction with the /USER qualifier, it defines the number of minutes that a specified individual's process must be idle, or connected if used with the /CONNECT qualifier, before it will be killed.

If it is used in conjunction with the /TERMINAL qualifier, it defines the number of minutes that a specified terminal must be idle, or connected if used with the /CONNECT qualifier, before it will be killed.

If it is used in conjunction with the /IMAGE qualifier, it defines the number of minutes that a process running the specified image must be idle, or connected if used with the /CONNECT qualifier, before it will be killed.

If it is used in conjunction with the /SERVER qualifier, it defines the number of minutes that a process using the specified terminal server port must be idle, or connected if used with the /CONNECT qualifier, before it will be killed.

If it is used in conjunction with the /PATH qualifier, it defines the number of minutes that a process running a specific copy of an image must be idle, or connected if used with the /CONNECT qualifier, before it will be killed.

If it is used in conjunction with the /IDENTIFIER qualifier, it defines the number of minutes that a process with the specified identifier must be idle, or connected if used with the /CONNECT qualifier, before it will be killed.

If used in conjunction with /DIALUP, /UIC, /USER, /TERMINAL, /SERVER, /IMAGE, /PATH or /IDENTIFIER and the /CONNECT qualifier, it defines the maximum amount of time the user can be connected before HITMAN will terminate them regardless of activity.

See the chapter "Warning Idle Users" for the order of precedence of wait and warning times.

When you lower a wait time, there may be processes on the system which have already accumulated the reduced amount of idle or connect time. HITMAN will immediately warn them. If they have not become active by the next data collection, it will kill them. To avoid problems, we recommend you stop HITMAN, lower the wait times and restart HITMAN. When HITMAN changes permanent datafiles this may also occur if the wait and warn times are different; HITMAN doesn't reset its internal process table where it keeps a record of idle time when it changes files.

Example

$ HITMAN/WAIT=30

Set the wait time for all processes to 30 minutes.

$ HITMAN/UIC="[20,*]"/WAIT=10

Set the wait time for processes in the UIC group 20 ([20,*]) to 10 minutes.

$ HITMAN/USER=LARRY/WAIT=999

Set user LARRY to the maximum amount of wait time 999 minutes.

/WSDEFAULT=n

Modify Parameters	**M**
Prime/Nonprime	**E**
Modify Flags & Values	**N**
Working Set	**U**

/WSDEFAULT=n (No default)

Use the WSDEFAULT qualifier, in conjunction with /WSEXTENT and /WSQUOTA to control the working set for HITMAN's detached process. At sites that experience a lot of page faults with HITMAN's detached process these qualifiers can be used to increase the working set and minize the faults. The /WSDEFAULT sets the value for HITMAN's detached processes in exactly the same way the WSDEFAULT authorize parameter does for interactive processes. This qualifier does not take

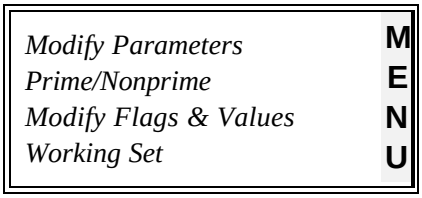
effect until the next time HITMAN is started. Normally you would increase all three values at the same time.

Example

$ HITMAN /WSDEFAULT=1024 /WSEXTENT=2048 /WSQUOTA=4096 - /FILE=BOTH

Change the working set values for HITMAN whether it is started in prime or nonprime.

/WSEXTENT=n



/WSEXTENT=n (No default)

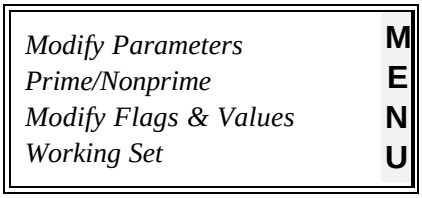
Use the WSEXTENT qualifier, in conjunction with /WSDEFAULT and /WSQUOTA to control the working set for HITMAN's detached process. At sites that experience a lot of page faults with HITMAN's detached process these qualifiers can be used to increase the working set and minize the faults. The /WSEXTENT sets the value for HITMAN's detached processes in exactly the same way the WSEXNT authorize parameter does for interactive processes. This qualifier does not take effect until the next time HITMAN is started. Normally you would increase all three values at the same time.

Example

$ HITMAN /WSDEFAULT=1024 /WSEXTENT=2048 /WSQUOTA=4096 /FILE=BOTH

Change the working set values for HITMAN whether it is started in prime or nonprime.

/WSQUOTA=n



/WSQUOTA=n (No default)

Use the WSQUOTA qualifier, in conjunction with /WSDEFAULT and /WSEXTENT to control the working set for HITMAN's detached process. At sites that experience a lot of page faults with HITMAN's detached process these qualifiers can be used to increase the working set and minize the faults. The /WSQUOTA sets the value for HITMAN's detached processes in exactly the same way the WSQUOTA authorize parameter does for interactive processes. This qualifier does not take effect until the next time HITMAN is started. Normally you would increase all three values at the same time.

Example

$ HITMAN /WSDEFAULT=1024 /WSEXTENT=2048 /WSQUOTA=4096-
/FILE=BOTH

Change the working set values for HITMAN whether it is started in prime or nonprime.

How HITMAN handles errors

This chapter shows you how HITMAN handles errors.

Because the HITMAN process which monitors your system is detached, (not connected to a terminal), it sends all error messages to the console and to the HITMAN_DAT:HITMAN_ERROR.{node} file. The HITMAN_ERROR file is an ASCII text file which you can type or edit. HITMAN appends error messages to the existing file rather than creating a new one.

When an error occurs, there will be one or more messages. The first message gives the HITMAN error message for what happened, for example, "unable to open log file". The next error message will give the reasons, for example, "insufficient quota".

All errors from the detached process are signalled on the operator's console regardless of the setting of the console logging switch. This is to ensure that you will know as soon as possible when the HITMAN detached process encounters a problem.

Signalling these errors is done by establishing an error handler which sends the error to the specified operator class, then continues with the error handling. All errors pass through the error handler and it decides whether the error comes from the detached process and has to be signalled.

NOTE: If you want to receive messages that are sent to the console, you can enable yourself as a security operator using the REPLY/ENABLE= command.

The following are examples of HITMAN error messages as they appear on the console:

```
%%%%%%%%%%  OPCOM   7-APR-2002 19:24:04.00  %%%%%%%%%%
Message from user LARRY
%HITMAN-W-ERRCREMBX, Error creating mailbox

%%%%%%%%%%  OPCOM   7-APR-2002 19:24:05.36  %%%%%%%%%%
Message from user LARRY
%SYSTEM-F-NOPRIV, no privilege for attempted operation
```

Troubleshooting HITMAN problems

This chapter shows you how to resolve problems with HITMAN.

General checks

If you have a problem running HITMAN, please do the following:

Verify that HITMAN logicals are defined
Verify that your process has the necessary privileges
Verify that the command definition is installed and compatible
Verify that your data files are okay
Verify that HITMAN is currently running
Obtain information about HITMAN's status
Check for HITMAN errors
Call Saiga Systems' Technical Support

Verify that HITMAN logicals are defined

HITMAN uses five logicals to access data files and executables. Normally, these logicals are defined as part of the system startup. However, to ensure the logicals are defined, type the following command:

```
$ @SYS$MANAGER:HITMAN_SYSTEM_LOGICALS
```

This procedure is created as part of the HITMAN installation procedure. If the logicals were already defined, you will see the following message:

%DCL-I-SUPERSEDE, previous value of {logical_name} has been superseded

repeated five times.

To verify that the HITMAN logicals are defined, type the following command:

```
$ SHOW LOGICAL HIT*
```

You should see output similar to the following:

(LNM$PROCESS_TABLE)

(LNM$JOB_xxxxxx)

(LNM$GROUP_xxxxxx)

(LNM$SYSTEM_TABLE)

```
 "HITMAN_CDU"   = "ddcu:[HITMAN.CDU]"
 "HITMAN_COM"   = "ddcu:[HITMAN.COM]"
 "HITMAN_DAT"   = "ddcu:[HITMAN.DAT]"
 "HITMAN_DOC"   = "ddcu:[HITMAN.DOC]"
 "HITMAN_EXE"   = "ddcu:[HITMAN.EXE]"
```

where "ddcu:" is the disk drive on which HITMAN is installed. This output indicates that the HITMAN logicals are properly defined in the SYSTEM table.

Verify that your process has the necessary privileges

HITMAN consists of an interactive program you use to control HITMAN's activity and a detached process which monitors your system. Both of these programs requires OpenVMS privileges to run.

To submit batch jobs, the HITMAN detached process must be started from an account which has the privilege SYSPRV as a default privilege.

To modify or list HITMAN parameters, you need the privileges SECURITY and SYSPRV.

To start the HITMAN detached process, you need either:

SETPRV

or

ALTPRI,CMKRNL,DETACH,EXQUOTA,OPER,PRMMBX,SYSNAM,SYSPRV,WORLD

To ensure your process has all the necessary privileges, type the following command:

```
$ @HITMAN_COM:SET_PRV
```

If you see the following message:

%SYSTEM-W-NOTALLPRIV, not all requested privileges authorized

the account you are logged into does not have all the privileges necessary to run HITMAN. You must log into an account which does have all the privileges. Normally, the SYSTEM account will have (or be able to acquire) all the necessary privileges to run HITMAN.

If your process does not have all the necessary privileges, HITMAN will list the additional privilege(s) you need when you enter a command. Here is a sample error message indicating that the process does not have the DETACH privilege:

%HITMAN-E-REQDETACH, Operation requires DETACH privilege

To find out what privileges your process has, type the command:

```
$ SHOW PROCESS/PRIVILEGES
```

To give your process specific privileges, type the command:

```
$ SET PROCESS/PRIVILEGE=privilege_name
```

Verify that command definitions are installed and compatible.

If you receive either of the following messages when you type a command:

%DCL-W-IVVERB, unrecognized command verb - check validity and spelling
\HITMAN\

%CLI-F-SYNTAX, error parsing 'qualifier_name'
-CLI-F-ENTNF, specified entity not found in command tables

where "qualifier_name" is one of the HITMAN qualifiers, HITMAN's command definition has not been installed in the system or process table or the command definition version you are using is not compatible with the HITMAN executable version.

To add the command definition to your process table, type the following command:

```
$ @HITMAN_COM:INSTALL_CDU
```

Verify your HITMAN data files
A simple check for Hitman data files is to list the files. To list them to the screen for visual verification:

```
$ hitman /list /file=both
```

To list them to a file for printing or e-mail:

```
$ hitman /list /file=prime /output=prime.lis
$ hitman /list /file=nonprime /output=nonprime.lis
```

If Hitman is running but not performing as expected you can send an e-mail to our support department with an explanation of the problem and attach a copy of the permanent data file listings for us to review. This helps us respond with concrete suggestions for resolving the issue.

Verify that HITMAN is currently running on your system

To verify that HITMAN is running on your system, type:

```
$ SHOW SYSTEM
```

and verify that there is a HITMAN process on your system. HITMAN spends most of its time in a HIB state, except for a very short time when it collects data.

Obtain information about HITMAN
To obtain information about HITMAN, type:

```
$ HITMAN/STATUS
```

This will produce a screen full of information and may indicate the source of your problem.

Check for HITMAN errors
Check for HITMAN errors by typing:

```
$ DIRECTORY HITMAN_DAT:HITMAN_ERROR.*
```

The file HITMAN_ERROR.{nodename} contains a history of all the errors HITMAN has encountered. If it exists, stop HITMAN and type it out. Please note, this file is locked while HITMAN is running.

If the HITMAN detached process keeps disappearing
If the HITMAN detached process is disappearing,

Check the HITMAN_ERROR file for error messages and take appropriate action.

If there are no messages in the error file it usually indicates that the detached processes has failed to start. Examining the VMS accounting record for the detached processes may provide a clue to the reason; for example if HITMAN failed to start at 10:15 am on May 1, 2002:

```
$ ACCOUNTING/FULL/SINCE=1-MAY-2002:10:14/PROCESS=DETACHED
```

the text message portion of the display may be useful, make a note of it and contact technical support at Saiga Systems.

When HITMAN does not hit users you think it should

- Ensure HITMAN is in HIT mode (HITMAN/HIT=HIT)

- Ensure data collection is on (HITMAN/ON)

- If /MINIMUM_PROCESS is specified, ensure there are sufficient interactive processes

- Check whether the user is protected by doing a HITMAN/DUMP/USER=username while the user is logged in. There are several ways a process can get protected. A complete description of the HITMAN/DUMP command and the resulting output is contained in the next section of this manual, "Getting information on processes HITMAN is monitoring" (page 211).

- Check whether the user is using small amounts of CPU, BIO's or DIO's which makes HITMAN think it is active. You may have to adjust the thresholds.

- Check the output from HITMAN/STATUS for a message that only processes at the DCL prompt will be terminated. In this mode HITMAN will only hit idle users at the DCL level; to tell HITMAN to hit idle users that aren't at the DCL level enter HITMAN/IMAGE_NAME.

- Check whether the user is protected because their process is not interactive by doing a HITMAN/DUMP/USER=username while the user is logged in. A complete description of the HITMAN/DUMP command and the resulting output is contained in the next section of this manual, "Getting information on what HITMAN is monitoring" (page 211).

When HITMAN hits users you think it should not

1. Check the log file to see why the user was hit.
   There are four reasons:
   - idle
   - connect
   - unauthorized
   - autohit

   The log record indicates why the user was terminated.

2. Check the /CPU_THRESHOLD, /BIO_THRESHOLD and /DIO_THESHOLD settings

3. Check the value of the /TYPE qualifier; setting this incorrectly may result in processes being hit unexpectedly.

Getting information on what HITMAN is monitoring

This chapter shows you how to get information about a process HITMAN is monitoring.

HITMAN allows you to see information on any process it is monitoring. If a user is not being terminated when you think they should be or vice-versa, use this function to find out why.

NOTE: The HITMAN detached process must be running before you can use this command.

To display user information,

1. Select the "Show User Information" option on the main menu.

2. Select the default permanent data file.

3. Enter a username (wildcards are not allowed)

4. Specify a file name into which HITMAN will write the user information (or take the default).

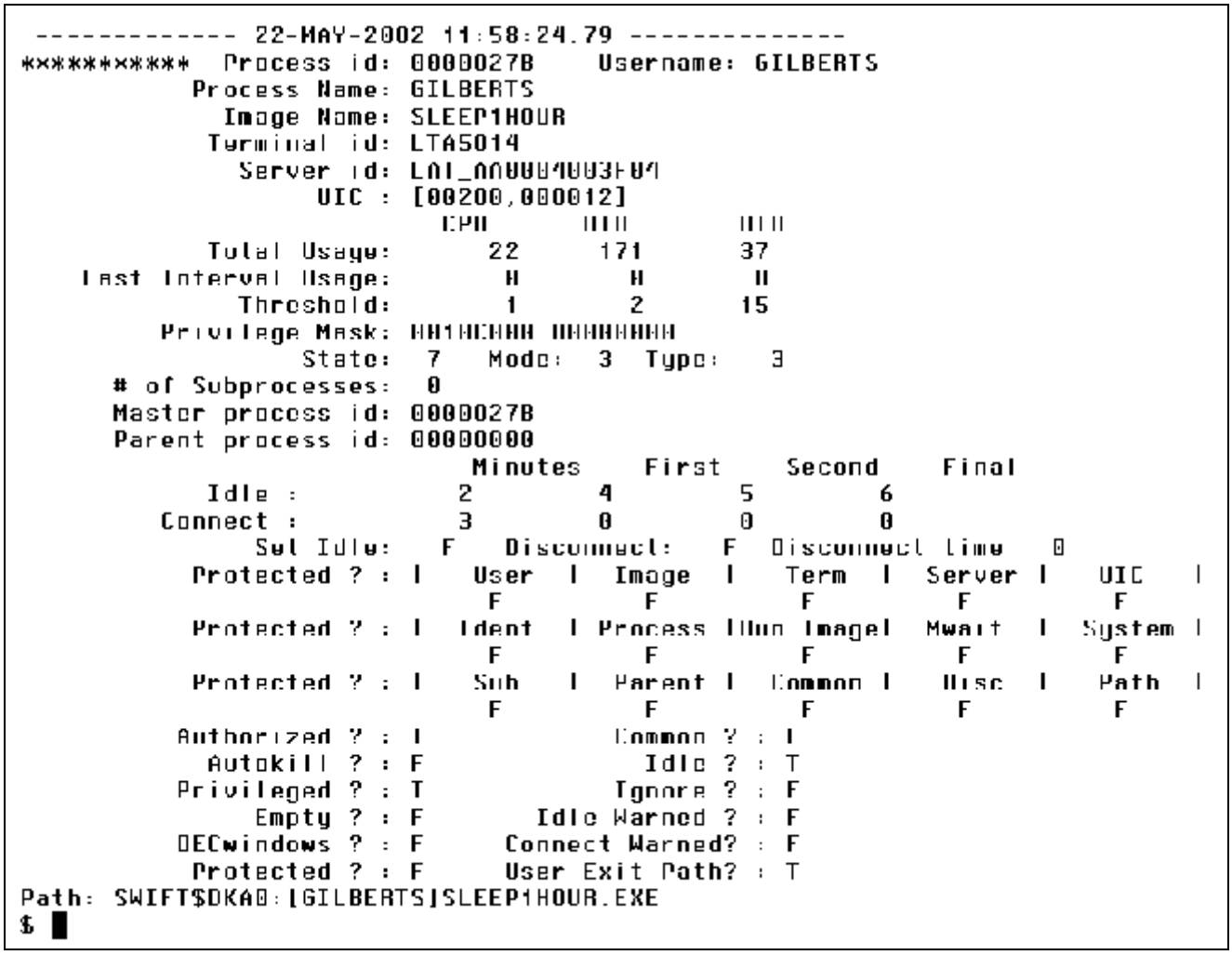
HITMAN will generate an ASCII text file containing all the information it has on processes matching the username you specified. By default, the file is called "DUMP.DMP" and is created in the directory pointed to by the logical "HITMAN_DAT". Since the file is an ASCII text file, you can TYPE it, PRINT it or use a text editor to look at it.

To show user information using the command line interface, type:

```
$ HITMAN/DUMP/USER=username
```

The following page shows the contents of the user dump report and explains the fields on the dump report.

NOTE When you do a HITMAN/DUMP/USER, the report is appended to the end of the existing DUMP.DMP file if one exists. Otherwise, a new file is created.



Sample output from a Hitman User Dump

The first line of the dump records the date and time when the dump was done.

The next few lines contain basic information about the process such as username, UIC, image name, terminal ID, server ID, UIC, resource usage, privileges, etc.

The block with Idle and Connect shows the number of minutes HITMAN has seen the user idle or connect, if that option is in effect. It also shows when the user would receive the first warning, second warning and be terminated.

The lines starting with the text "Protected ? :" show if the process is protected
If the “Protected flag" field at the bottom of the dump is T then a process is seen as protected. These individual protection flags should show which protection list contains the entry that is currently protecting this process.

"Authorized" shows whether the process is authorized to have privileges.

"Autokill" shows whether the process is running a restricted image.

"Privileged" shows whether the process has privileges in the ALL category.

"Empty" shows whether the process is running an image.

"DECWindows" shows whether HITMAN is configured to link the multiple processes in the DECWindows environment together.

"Common" shows whether the process is linked via the /COMMON qualifier.

"Idle" shows whether the process was idle during the data collection interval.

"Ignore" shows whether HITMAN ignores the process because it is not a process type that is currently eligible to be terminated.

"Warned" shows whether the process has been warned at least once since it became idle.

"Connect Warned" shows whether the process has been warned at least once because it is approaching its' connect time limit.

“User Exit Path” shows whether the path, which is included on the last line(s) of the dump matches a path for which a special exit entry has been added.

The fields "Process Jobtype" and "Mode" give a numeric value. The following tables provide the translation for those values:

PROCESS JOBTYPES

Value	Jobtype
0	Detached
1	Network
2	Batch
3	Local
4	Dialup
5	Remote

PROCESS MODES

Value	Process Mode
0	Other (includes detached)
1	Network
2	Batch
3	Interactive

Any processes that should be hit but are not, because they are not an eligible process type, have the IGNORE flag set to "T". These processes can be made eligible for termination using the appropriate /TYPE keywords.

/TYPE Keyword	Process Mode	Process Jobtype	Description
INTERACTIVE	3	Any	Interactive processes of any jobtype
BATCH	2	2	Batch jobs
NETWORK	1	1	Network processes
DETACHED	0	Any	Detached processes of any jobtype
DIALUP	3	4	Interactive processes of the jobtype DIALUP

Event dump

For an explanation of the event dump report with an example report please refer to the section on troubleshooting events at page 106.

System wide process dump

A dump with some key information about every process on the system that Hitman is monitoring is available to help with isolating problems with Hitman. To generate this dump:

```
$ hitman /dump=list
```

Sample output from this listing is included below.

```
********************* List Dump: 22-MAY-2002 12:22:21.85 *********************

                                                                                                                  Spec
PID      Process         Username     Terminal  Image                                Protect Ignore Idle 1st  2nd  Wait Exit
-------- --------------- ------------ --------- ------------------------------------ ------- ------ ---- ---- ---- ---- ----
      41 SWAPPER         SYSTEM                                                          Y       Y      0   20   25   30   F
      44 LANACP          SYSTEM                 LANACP                                   N       Y      0   20   25   30   F
     266 _LTA5012:       SYSTEM       LTA5012                                            N       N     13   20   25   30   F
      6B PWRK$MONITOR    SYSTEM                 PWRK$MONITOR                             N       Y      0   20   25   30   F
     2AD BATCH_29        GILBERTS                                                        N       N      0    4    5    6   F
      77 DECW$SERVER_0   SYSTEM                 DECW$SERVER_MAIN                         N       Y      0   20   25   30   F
      78 DTLOGIN         <login>                LOGINOUT                                 N       Y      0   20   25   30   F
      79 DTGREET         SYSTEM                 DTGREET                                  N       Y      0   20   25   30   F
     27B GILBERTS        GILBERTS     LTA5014   SLEEP1HOUR                               N       N      1    4    5    6   T
      BD SYSTEM          SYSTEM       LTA5009   HITMAN_DETACHED                          N       N      0   20   25   30   F
$
```

For sample purposes most of the detached system processes have been deleted from this report but normally a line is generated for each process on the system including; Process ID (PID), Process Name, Username of process, terminal process is connected through, image process is running, a flag for whether or not the process is protected, a flag indicating whether or not the process is being ignored (if this flag is True it indicates the process is a type of process that is currently

ignored by Hitman, the number of minutes the process has been idle if it is not being ignored, the current values for 1st warning time, 2nd warning time and final idle time in effect for this process, and finally a flag indicating whether or not this process would be terminated using a user defined special exit procedure.

Summary of current processes and actions taken since last start

A summary report showing totals for current processes on the system as well as totals for the number of warnings sent and processes deleted can be generated at anytime:

```
$ hitman /dump=summary
```

A sample of the output is included below:

```
********************* Summary Dump: 22-MAY-2002 12:38:37.53 ****

   CEF     0
   COM     0
  COMO     0
   CUR     1
 COLPG     0
   FPG     0
   HIB    32
  HIBO     0
   LEF     9   DETACHED    36
  LEFO     0   NETWORK      0
 MWAIT     0      BATCH     2   INTERACTIVE     5
   PFW     0      LOCAL     4         BATCH     2   PROTECTED     1
  SUSP     0    DIALUP      0       NETWORK     0     IGNORED    35
 SUSPO     0    REMOTE      0         OTHER    35       OTHER     6
     ----              ----                 ----                ----
TOTAL    42              42                   42                  42

               LAST DATA COLLECTION

                                 First Warning:     0
                                Second Warning:     1
                              Multiple Warning:     0
                                       Deleted:     0
                                Delete Pending:     0

                  RUNNING TOTALS

                          Total First Warnings:           20
                         Total Second Warnings:           20
                            Total Force Exits:            0
                       Total Idle Terminations:            2
                    Total Connect Terminations:            0
                  Total DecWindow Terminations:            0
               Total Image Autohit Terminations:            0
   Total UnAuthorized Privilege Terminations:               0
$
```

This report shows the number of processes in each state, the number of processes of each type, the number of processes of each mode, the number of processes protected, ignored and eligible to be terminated. It also gives some general statistics for the last data collection and running totals.

Resource quotas

Since HITMAN uses asynchronous system traps (AST) to time when to check events, you may have to increase a SYSGEN default AST limit parameter. The process AST limit is the number of ASTs that a process can have outstanding at one time. The detached process's AST limit defaults to the SYSGEN parameter PQL_DASTLM. If you see the detached process go into RWAST (resource wait for AST), you may want to increase this parameter.

Stopping and starting HITMAN

If the system is heavily loaded, and you issue the command HITMAN/STOP immediately followed by the command HITMAN/START, you may receive the error message "HITMAN is already running". This is because the detached process may not have actually stopped running. If this happens, execute the command HITMAN/START again.

We recommend that you stop and restart HITMAN once a week to prevent the LOG_FILE from using up too much disk space. You can use a TIME event to do this automatically.

Clusters

When running HITMAN on a cluster, you can either install HITMAN on one common disk or install one copy on each node. If HITMAN is installed on a common disk, HITMAN can use the same permanent data file for each node. All you have to do is define the HITMAN logicals on each node, and start HITMAN.

Technical Support

Calling Saiga Systems

If you try the above suggestions and still continue to have problems, please contact Saiga Systems Technical Support at 1-800-561-8876. Technical support is provided between 8:00 a.m. and 4:30 p.m. Mountain Time.

When you call 1-800-561-8876, please have the following information available:

Alpha AXP or VAX Model
OpenVMS version
Company Name
HITMAN version (from HITMAN/STATUS output)

The operator will verify that you have a support agreement and pass you on to the Technical Support group.

Support is also available by e-mail; send a message to support@saiga.com or visit our web page at http://www.saiga.com/ and fill in the support form.

On-line Customer Support

Saiga Systems maintains a web page at http://www.saiga.com/ . With it you can access technical information, software upgrades and product information. You can also send us e-mail with questions and suggestions. Product patches, special kits, release notes and documentation are all available at this site. If you wish to upload something for us you will need to make special arrangements since uploads are disabled for security purposes.

From here you will find links to current and past, supported, versions of all our products. The documentation for the current version of each product is also online in html format.

Appendix A, Error messages

7PRIVSALLOW, Only seven privileges are allowed in one event

You specified more than seven privileges in a PRIV_CHECK event which allows a maximum of seven. Please specify seven or fewer privileges.

AUTHNOTFOUND, Authorized user: *username* was not found in Authorized list

The specified username was not found in the authorized list. Please check for typing or spelling mistakes, and re-enter the command.

AUTOIMGEXISTS, Autohit *image_name* already exists in Autohit list

The image name you specified in the /AUTOHIT/IMAGE qualifiers already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

AUTUSEEXISTS, Authorized user *username* already exists in Authorized list

The username you specified in the /AUTHORIZED/USER qualifier already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

BADWEEKDAY, Invalid weekday or weekday keyword specified, please call Saiga Systems

The keyword you specified for /DAY is invalid. This may indicate an internal error in HITMAN. Please call Saiga Systems.

BLOCKSTOLNG, Specified free blocks text was too long, value truncated to 10 characters

The free blocks text you entered was too long to contain a valid number of blocks. Please omit leading blanks and re-enter the command.

CLASSTOOLNG, Specified operator class text was too long, value truncated to 8 characters

The operator class text you entered was too long to contain a valid operator class. Please omit leading blanks, check for typing or spelling mistakes, and re-enter the command.

CMDIGNORED, The last hitman command is ignored since a data collection is running

The last command entered was requesting action from the detached process. The detached process was in the middle of a data collection and therefore the command was ignored. Please re-enter the command.

DESCTOOLNG, Specified description text was too long, value truncated to 50 characters

The event description text you entered was too long. Please omit leading blanks, check for typing or spelling mistakes, and re-enter the command.

DETACHNOTRUN, HITMAN's detached process is not running, this is required for dump

You have issued a /DUMP/USER command and the HITMAN detached process is not running. Start HITMAN and re-issue the /DUMP/USER command.

DEVICEREQ, Device required for this event

A device name is required when entering a Check_free or Error event. Please check for typing or spelling mistakes, and re-enter the command.

DEVNAMTOLNG, Specified device name was too long, this field can only accept 59 characters

The device name you entered was too long. Please omit leading blanks, check for typing or spelling mistakes, and re-enter the command.

DSTLSTTOLNG, Specified distribution list is more than 39 characters, please re-enter

The distribution list text you entered is too long. Please check for typing or spelling mistakes, and re-enter the command.

ERRCENTEXT, Error centring text in SMG display

There has been an error centring text in the SMG display. Check if you have entered a string greater than 78 characters.

ERRCLOLIS, Error closing list file

An error occurred while closing the listing file. The error message that follows will further explain the problem.

ERRCLOLOG, Error closing log file

An error occurred while closing the log file. The error message that follows will further explain the problem.

ERRCLOMAI, Error closing mail message text file

An error occurred while closing the temporary mail message file. The error message that follows will further explain the problem.

ERRCLOPERM, Error closing permanent data file

An error occurred while closing the permanent data file. The error message that follows will further explain the problem.

ERREADLOGFIL, There has been an error reading the log file

An error occurred while reading the log file. The error message that follows will further explain the problem.

ERROPEAUDIT, Error opening audit file

An error occurred while opening the audit file. The error message that follows will further explain the problem.

ERROPEDUMP, Error opening dump file for output

An error occurred while opening the output file for the dump command. The error message that follows will further explain the problem.

ERROPELIS, Error opening list file

An error occurred while opening the listing file. The error message that follows will further explain the problem.

ERROPELOG, Error opening log file

An error occurred while opening the log file. The error message that follows will further explain the problem.

ERROPEMAI, Error opening mail message text file

An error occurred while opening the temporary mail message file. The error message that follows will further explain the problem.

ERROPEPERM, Error opening permanent data file

An error occurred while opening the permanent data file. The error message that follows will further explain the problem.

ERRREAPERM, Error reading permanent data file

An error occurred while reading from the permanent data file. The error message that follows will further explain the problem.

ERRSETAST, Error setting up mailbox AST

An error occurred while creating the mailbox link to the detached process. The error message that follows will further explain the problem.

ERRSTRTDET, Error starting detached process

An error occurred while starting the detached process. Please check HITMAN_DAT:HITMAN_ERROR.*node_name* for error messages.

ERRWRITLIS, Error writing list file

An error occurred while writing to the listing file. The error message that follows will further explain the problem.

ERRWRITLOG, Error writing log file

An error occurred while writing to the log file. The error message that follows will further explain the problem.

ERRWRITMAI, Error writing mail message

An error occurred while writing the temporary mail message. The error message that follows will further explain the problem.

ERRWRITPERM, Error writing permanent data file

An error occurred while writing to the permanent data file. The error message that follows will further explain the problem.

EXITEXISTS, Image *image_name* already exists in the image exit list

The image name you specified is already in the exit list. Please check the name and re-enter the command.

EXTRAIMGINFO, Extra image name information provided, only file name used

The image name you specified contained more than an image name. Only the file name was used. No action necessary.

FILENAMTOLNG, Specified file name was too long, truncated to 50 characters

The file name you specified was too long. Filename was truncated to 50 characters. No action necessary.

FILETRUNC, The file loaded was truncated

A couple of reports are now loaded into the menu for review if desired instead of simply being written to disk. This message indicates that the line length of some line(s) in the report exceeded the allowable size and the line was truncated. If necessary exit Hitman and use an editor to view the complete file contents.

FIRGEWAIT, First warn time is greater than or equal to wait time

The first warn time must be less than the wait time. Please check for typing or spelling mistakes, and re-enter the command.

FNF, File *filename* not found

A file you specified does not exist. Please check for typing or spelling mistakes, and re-enter the command.

FREEREQ, Free blocks required for this event

The number of free blocks is required when entering a check free event. Please check for typing or spelling mistakes, and re-enter the command.

HMREQ, Please specify the hours and minutes hh:mm

Please specify the start or end time in the format HH:MM. There must be at least four characters. Here are some examples: "5:00" or "17:30".

HOLIDAYEXISTS, Holiday: *date* already exists in Holiday list

The date you tried to add to the holiday list is already in the list. Please check for typing or spelling mistakes, and re-enter the command.

HOLNOTFOUND, Holiday: *date* was not found in Holiday list

The date you tried to remove from the holiday list is not in the list. Please check for typing

or spelling mistakes, and re-enter the command.

HOLTOOLONG, Holiday: *date* is too long

The date text you specified is too long to be a valid date. Please omit leading blanks, check for typing or spelling mistakes, and re-enter the command.

IDENTEXISTS, Identifier: *identifier* already exists in the Identifier list

The identifier you specified using the /PROTECT/IDENTIFIER qualifiers already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

IDENTNOTFOUND, Identifier: *identifier* was not found in the Identifier list

The identifier you specified using the /PROTECT/IDENTIFIER qualifiers does not exist in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

IDENTNOTVLD, Not a valid UIC identifier, please re-enter

The UIC identifier that you have entered is invalid. Please check for typing or spelling mistakes, and re-enter the command.

IDENTUIC, Identifier is a UIC, use UIC list for this identifier

The identifier you specified using the /PROTECT/IDENTIFIER qualifiers is a UIC. Use the /UIC qualifier rather than the /IDENTIFIER qualifier. Please check for typing or spelling mistakes, and re-enter the command.

IDISCNOTFOUND, Image: *image_name* was not found in the image disconnect list

The image you specified using the /DISCONNECT/NOIMAGE qualifiers does not exist in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

IDLENOTFOUND, User: *username* was not found in the Set Idle list

The user you specified using the /SET_IDLE/NOUSER qualifiers does not exist in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

IDLETOOHIGH, Idle times have been set to default

The maximum idle time you specified using the /MAX_IDLE qualifier is too high. Please check for typing mistakes, and re-enter the command.

IDLETOOLOW, Idle times have been set to default

The minimum idle time you specified using the /MIN_IDLE qualifier is too low. Please check for typing mistakes, and re-enter the command.

IDLIMGEXISTS, Idle image already exists, please re-enter

The image you specified already has wait and warning times defined. Please check for typing or spelling mistakes, and re-enter the command.

IDLSERVEXISTS, Idle server already exists, please re-enter

The server you specified already has wait and warning times defined. Please check for typing or spelling mistakes, and re-enter the command.

IDLTERMEXISTS, Idle terminal already exists, please re-enter

The terminal you specified already has wait and warning times defined. Please check for typing or spelling mistakes, and re-enter the command.

IDLUSREXISTS, Idle username already exists, please re-enter

The user you specified already has wait and warning times defined.

IEXITCNOTFND, Image *image* not found in image exit code table

The image was not found in this table to be deleted. Be sure to specifying the correct /file when deleting entries that may not exist in both data files. The image may already have been deleted, in which case no action is necessary.

IMAGEEXISTS, Image: *image_name* already exists in the Image protect list

The image name you specified using the /PROTECT/IMAGE qualifiers already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

IMAGENOTFOUND, Image: *image_name* was not found in the Image list

The image you specified using the /PROTECT/NOIMAGE qualifiers does not exist in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

IMAGEREQ, Image required for this event

An image name is required for the event you are trying to enter. Please enter event with a valid image name.

IMGDISCEXISTS, Image: *image_name* already exists in the Image disconnect list

The image name you specified using the /DISCONNECT/IMAGE qualifiers already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

IMGNAMTOLNG, Specified image name was too long, truncated to 39 characters

The image name you specified was too long. Image name truncated to 39 characters. Please check for typing or spelling mistakes, and re-enter.

IMGNOTFOUND, Autohit image: *image_name* was not found in Autohit list

The image name you specified in the /AUTOHIT/NOIMAGE qualifiers does not exist in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

IMGTHRESEXISTS, Image already exists in the Image threshold list

The image name you specified for the image threshold list already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

INDIVNOTFOUND, User: *username* was not found in the Individual list

The username you specified in the /NOUSER qualifier does not exist in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

INOUTREQ, This event requires /IN, /OUT or both

You must specify either IN or OUT for this event. Please add the required text and re-enter the command.

INVALIDBIO, Invalid value for runaway event BIO's

You have entered an invalid value for the runaway buffered I/O threshold. Please check for typing mistakes, and re-enter the command.

INVALIDCPU, Invalid value for runaway event CPU

You have entered an invalid value for the runaway CPU threshold. Please check for typing mistakes, and re-enter the command.

INVALIDDIO, Invalid value for runaway event DIO's

You have entered an invalid value for the runaway direct I/O threshold. Please check for typing mistakes, and re-enter the command.

INVALIDDEVICE, Invalid disk device, please re-enter

You have entered an invalid disk device. Please check for typing or spelling mistakes, and re-enter the command.

INVALIDEVENT, Invalid event type specified, please re-enter

You have tried to enter an invalid event type. Please check for typing or spelling mistakes, and re-enter the command.

INVALIDFIR, Invalid first warning, time specified must be multiple of interval

The time specified for the first warning must be a multiple of the data collection interval. Please re-enter the command.

INVALIDGROUP, Invalid UIC group, please specify between 1 and 37776

The UIC group you specified was invalid. Please check for typing or spelling mistakes, and re-enter the command.

INVALIDINT, Invalid data collection interval

The interval you specified for /UPDATE_INTERVAL is not between 1 and 60. Please check for typing or spelling mistakes, and re-enter command.

INVALIDKEY, Invalid key entered, valid keys listed in information window below

The key you pressed is not valid with this screen. The valid keys are listed at the bottom of the screen.

INVALIDMEMBER, Invalid UIC member, please specify between 0 and 177776

The UIC member you specified was invalid. Please check for typing or spelling mistakes, and re-enter the command.

INVALIDOCTAL, Invalid octal value, please re-specify

The group and/or member in a UIC contain digits other than 0-7 and is not a valid octal number.

INVLDPATHNAM, Invalid path name specified, please reenter

Path names should include a device, diretory, filename and filetype with no version number. Verify that the path name was specified correctly and verify that the file actually exists; reenter.

INVALIDQUEUE, Invalid queue specified, please re-enter

The batch queue you specified was invalid. Please check for typing or spelling mistakes, and re-enter the command.

INVALIDSEC, Invalid second warning, time specified must be a multiple of interval

The time specified for the second warning must be a multiple of the data collection interval. Please re-enter the command.

INVALIDTIME, Prime start time is greater than or equal to Prime end time

The time you specified for prime start must be less than the prime end time. Please check for typing or spelling mistakes, and re-enter command.

INVALIDUPD, Invalid update interval

The update interval must be equal to or greater than the data collection interval and less than or equal to 60. Please re-enter the command.

INVALIDVAL, Invalid value, must be greater than 0

You have entered a value that is less than 0. Please enter a positive value or check for typing mistakes.

INVALIDWAIT, Invalid wait, time specified must be multiple of interval

The wait time you specified must be a multiple of the interval. Please check for typing or spelling mistakes, and enter command again.

INVALIDWARN, Invalid warn, must be between 1 and 999. Please re-enter

The warn time you specified must be a multiple of the interval. Please check for typing or spelling mistakes, and enter command again.

INVALNUMMESS, Invalid number of messages, must be between 1 and 99

You have entered an invalid number for the number of messages. Please enter a value between 1 and 99.

INVALPARAM, Invalid interval or wait time, leaving HITMAN

HITMAN stopped because either the interval is less than the minimum allowable interval or the wait time is less than the minimum allowable wait times. Please respecify interval or wait times.

INVDAY, Invalid day specified, please re-enter

The day you entered in not valid. Please check for typing or spelling mistakes, and re-enter the command.

INVHOLIDAY, Invalid holiday specified

The date you entered in not valid. Please check for typing or spelling mistakes, and re-enter the command.

INVIDENT, Invalid identifier, wildcards not allowed

You have entered a wildcard with a UIC identifier. Please re-enter the command without wildcard or check for typing or spelling mistakes.

INVIDLELIM, Invalid set idle limits specified, max. must be greater than min.

The maximum or minimum idle limits you specified are not valid. Please check for typing mistakes, and re-enter the command.

INVKEY, Last key entered was not valid for this screen

The last key entered is not defined as a function key on this screen. Refer to the bottom of the screen for which keys are currently defined.

INVLDBIO, Invalid bio threshold, default is 1

The BIO threshold you specified is invalid. Please check command line for typing or spelling mistakes, and re-enter the command.

INVLDBRDCST, Invalid broadcast type, must be USER1 to USER16

You have specified an invalid broadcast class. Check for typing or spelling mistakes, and re-enter the command.

INVLDCPU, Invalid CPU threshold, default is 3

The CPU threshold you specified is invalid. Please check command line for typing or spelling mistakes, and re-enter the command.

INVLDDEV, *device* is an invalid device for this node

The device you specified is not available on this node. Please check command line for typing or spelling mistakes, and re-enter the command.

INVLDDIO, Invalid DIO threshold, default is 0

The DIO threshold you specified is invalid. Please check command line for typing or spelling mistakes, and re-enter the command.

INVLDDSTLST, Invalid distribution list specified, please re-enter

The text you entered as a distribution list does not start with "@" or the length is equal to 1. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVLDFILNAM, Invalid file name

The file name you entered was invalid. Please check for typing or spelling mistakes, and re-enter the command.

INVLDFRCWT, Invalid force wait interval, please re-specify

The force wait you specified is not between 5 and 60. Please check for typing or spelling mistakes, and enter command again.

INVLDIMGNAM, Invalid image name specified, please re-enter

The image name you specified was invalid. Please check for typing or spelling mistakes, and re-enter the command.

INVLDMINPRC, Invalid number of processes, re-enter between 0 and 999

The value for minimum number of processes must be between 0 and 999. Please re-enter.

INVLDOPCLASS, Invalid operator class specified, default is SECURITY

The operator class you specified is invalid. Please check command line for typing or spelling mistakes, and re-enter the command.

INVLDOPERCLS, Invalid operator class: *class_name*

The operator class you specified is invalid. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVLDPRIO, Invalid base priority, default is 6

The base priority you specified is invalid. Please check command line for typing or spelling mistakes, and re-enter the command.

INVLDPRIV, Unknown Privilege: *privilege*, please check for typing mistakes

One of the privileges you tried to enter in the mask was invalid. Please check for typing or spelling mistakes, and re-enter.

INVLDTABLE, Invalid table, *table* appears to be corrupted

During reading of the permanent data file a table was found to be corrupted. Please restore your Hitman permanent data files from a backup.

INVLDTERM, Invalid terminal, terminals must end with a colon or have a wildcard

The terminal you specified either contains a colon and a wildcard or does not end in a colon. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVLDUSRNAM, Invalid username specified, please re-enter

The username you specified contains an illegal character or you specified a list when only a single username was allowed. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVLDVALUE, Invalid value entered, please re-enter with valid value

The value you entered was invalid. Please check for typing or spelling mistakes, and re-enter the command.

INVMAXIDLE, Invalid maximum idle time, must be between 2 and 999

The value you entered was invalid. Please check for typing or spelling mistakes, and re-enter the command.

INVMINIDLE, Invalid minimum idle time, must be between 2 and 999

The value you entered was invalid. Please check for typing or spelling mistakes, and re-enter the command.

INVMINUTES, Invalid value specified for minutes.

There was an invalid value specified for the number of minutes. Please re-enter with a valid value.

INVMONTH, Invalid month specified, please re-enter

The value you entered was invalid. Please check for typing or spelling mistakes, and re-enter the command.

INVOUTFILE, Invalid output file name, please re-enter

The file name you specified contains illegal characters or is too long. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVPRIMEDAY, You must enter /PRIME_START and /PRIME_END with /DAY

You must specify prime start and end times when you specify a day to be prime. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVPRIMETIME, Invalid time entered for a prime time, please re-enter

The time you specified is not valid. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVSERVERNAME, Invalid server name, please re-enter

The server name you specified is not valid. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVSETIDLE, You have entered an invalid idle time, please re-enter

The time you specified is not valid. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVSTARTEND, You must enter /DAY to specify which day to modify

You must specify a day when you specify prime start and end times. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVSTATE, Invalid value specified for state.

The state specified is not valid. Please re-enter with a valid state,

INVTIME, Invalid time specified, please re-enter

The time you specified is not valid. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVTIMEOUT, Invalid timeout specified, must be between 1 and 999

The time interval you specified is not valid. Please check for typing or spelling and typing mistakes, and re-enter the command.

INVUICFORMAT, Invalid UIC format, please enter in the format of [n,n]

The UIC format that you entered was invalid. Please check for typing or spelling mistakes, and re-enter the command.

INVUSERNAME, Invalid username, please re-enter

The username you specified contains illegal characters or you did not enter a username where one was required. Please check for typing or spelling and typing mistakes, and re-enter the command.

JOBREQ, Job required for this event

You must specify a job name for this event. Please check for typing or spelling and typing mistakes, and re-enter the command.

LIMITREQ, Limit required for this event

There must be a limit specified with this event. Please re-enter the event with a valid limit.

LOGCLOSE, Log file is already closed

The log file is already closed, type /LOG to reopen it.

LOGNTDFND, Logicals not defined, please check system table

Logical HITMAN_DAT is not defined. Please check system logicals and define if necessary.

MESSAGETOLNG, Specified message was too long, truncated to 200 characters

The message that you entered was too long. Message was truncated to 200 characters. No action necessary.

MINGEMAX, Minimum set idle time must be less than maximum set idle time

The minimum idle time must be less than the maximum idle time. Please check for typing mistakes, and re-enter the command.

MUSTSUPDSTLST, Must supply distribution list

You must specify a distribution list for this event.

NAMECHAN, Saiga Systems Idle Process Killer name changed

You have successfully changed HITMAN's name.

NOAUTH, No authorized users in list

You specified delete in the screen interface but there are no authorized users in the list.

NOAUTOIMG, No autohit images in list

You specified delete in the screen interface but there are no autohit images in the list.

NODELDEFAULT, Not allowed to delete default event records

You cannot delete default event records. You can only modify them.

NODETOOLNG, Specified nodename text was too long, truncated to 15 characters

You specified more than 15 characters for the nodename. Please check for typing or spelling and typing mistakes and re-enter the command.

NODISCONNECT, Disconnecting processes not allowed on AXP machines

You cannot use the disconnect feature on Alpha AXP systems.

NODISKINFO, Unable to collect disk information on *device_name*

HITMAN is unable to collect information on the device because the disk is not mounted or not available on this node. Please check for typing or spelling and typing mistakes, and re-enter the command.

NOEVENTFND, Event ID not found, no event deleted

The event ID you specified to delete does not exist. Please check for typing or spelling and typing mistakes, and re-enter the command.

NOHOLIDAYS, No holidays in list

You specified delete but there are no holidays in the list.

NOIDLE, No idle and warn users in list

You specified delete but there are no users in the list.

NOIMAGEEXIT, No image exit codes in list

You specified delete but there are no image exits in the list.

NOIMGDISC, No image disconnect entries in list

You specified delete but there are no images in the disconnect list.

NOLOGFILE, Log file does not exist in HITMAN_DAT: directory

The current log file is not found in the HITMAN_DAT directory. Please check HITMAN_DAT directory.

NONEXPR, Failure starting HITMAN, check console for typing or spelling mistakes

The HITMAN detached process had trouble starting, check console and HITMAN_ERROR.LOG for more information.

NONONPRIME, Non-Prime data file does not exist, please check HITMAN_DAT directory

The nonprime data file PERM_DATA_NONPRIME.DAT does not exist in the HITMAN_DAT directory. Please check for typing or spelling and typing mistakes, and re-enter the command.

NOPROTENTRY, No protected entries in list

You specified delete but there are no protected entries in the list.

NORIGHTS, Problem with SYS$FIND_HELD, possibly no rights database

You specified identifiers but you do not have a rights database.

NORMAL, Normal successful completion

The operation has been successfully completed.

NOSETIDLE, You are not allowed to use SET_IDLE

You have not been authorized by the system manager to use the set idle function.

NOSETIDLEENTRY, No set idle users in list

There are no users in the set idle list.

NOSUCHIDENT, No such identifier, please check for spelling mistakes, and re-specify

The identifier you specified does not exist. Please check for typing or spelling mistakes, and re-enter the command.

NOSUCHNODE, No such node is currently in this cluster

The node name you specified is unknown. Please check for typing or spelling and typing mistakes, and re-enter the command.

NOTDET, Image must be run as a detached process

The image HITMAN_DETACHED.EXE must be run as a detached process. It cannot be run interactively. To start the detached process, use $ HITMAN/START.

NOTPERMFILE, HITMAN tried to read a file that is not a permanent data file

The file HITMAN tried to use as a permanent data file is not a permanent data file. Ensure that the HITMAN logicals are properly defined.

NOTRUN, HITMAN is not running

The detached process for HITMAN is not running. This is an informational message. It does not represent an error. To start the process, use HITMAN/START.

NOUSRDISC, No user disconnect entries in list

You specified delete but there are no entries in the disconnect list.

NOWILD, Cannot specify wildcard for this field

You cannot specify a wildcard for this field.

NUMMESSTOLNG, Specified number of messages text too long, truncated to 2 characters

The text you specified for the number of messages contained more than 2 characters. Please check for typing mistakes.

PATHEXISTS, Image *path* already exists in the Path protection list

No action is necessary.

PATHNAMTOLNG, Specified path name was too long, truncated

The specified path name was too long and was truncated. Review your parameters to make sure path will still function as saved; if not, the only fix is to move the file to a location which results in a shorter path length.

PATHNOTFOUND, Path *path* was not found in the path list

The specified path was not found. This usually occurs when attempting to delete a path that has already been deleted. Verify that the path name was entered correctly and reenter if necessary.

PATHTOOLONG, A path name exceeded 139 characters

Path names cannot exceed 139 characters. Verify that the path was specified correctly and reenter if necessary. If the path was correct you will need to move the image to a location which results in a shorter path.

PRCSSTP, Stopping HITMAN, Please wait ...

HITMAN requires some time to shut down. This should be no longer than 5 seconds.

PRCSSTRT, Starting HITMAN, Please wait ...

HITMAN requires some time to start up. This should be no longer than 5 seconds.

PRIVILEGEREQ, Privilege required for this event

A privilege is required for a Privilege Check event. Please check for typing or spelling mistakes, and re-enter the command.

PROCESSEXISTS, Process : *process_name* already exists in process protect list

The username you specified using the /PROTECT/PROCESS= qualifiers already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

PROCESSNOTFOUND, Process : *process_name* was not found in process protect list

The username you specified using the /PROTECT/PROCESS= qualifiers does not exist in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

PROCESSREQ, Process required for this event

A process name is required for a Missing event. Please check for typing or spelling mistakes, and re-enter the command.

PROCID, Identification of created process is *nnnnnnnn*

The process identification of the detached process is the number that was displayed. This is displayed as a result of the HITMAN/START command.

PROCNAMTOLNG, Specified process name too long, truncated to 15 characters

The process name you specified exceeds 15 characters. Please re-enter with fewer characters.

PROCTOOLONG, Process name too long, must be under 15 characters

The process name you specified exceeds 15 characters. Please re-enter with fewer characters.

PROUSEEXISTS, User : *username* already exists in user protect list, not added

The username you specified using the /PROTECT/USER= qualifiers already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the

command.

QUEERROR, Error processing a monitor queue event

There was an error in processing the queue for the monitor queue event. Please verify the queue name.

QUENAMTOLNG, Specified queue name was too long, truncated to 31 characters

The queue name you specified was too long. Queue name was truncated to 31 characters. No action necessary.

QUEUEREQ, Queue required for this event

A queue name is required for this event. Please check for typing or spelling mistakes, and re-enter the command.

RECIGNOR, Record ignored in permanent data file

A record in the Permanent data file could not be understood by HITMAN. Please check HITMAN/LIST to make sure that all qualifiers are correct.

REQDETACH, Operation requires DETACH privilege

HITMAN requires the DETACH privilege to run. Make sure that you have the appropriate privilege and re-enter the command.

REQEXQUOTA, Operation requires EXQUOTA privilege

HITMAN requires the EXQUOTA privilege to run. Make sure that you have the appropriate privilege and re-enter the command.

REQOPER, Operation requires OPER privilege

HITMAN requires the OPER privilege to run. Make sure that you have the appropriate privilege and re-enter the command.

REQPRMMBX, Operation requires PRMMBX privilege

HITMAN requires the PRMMBX privilege to run. Make sure that you have the appropriate privilege and re-enter the command.

REQPSWAPM, Operation requires PSWAPM privilege

HITMAN requires the PSWAPM privilege to run. Make sure you have the appropriate privilege and re-enter the command.

REQSECURITY, Operation requires SECURITY privilege

HITMAN requires the SECURITY privilege to run. Make sure that you have the appropriate privilege and re-enter the command.

REQSYSNAM, Operation requires SYSNAM privilege

HITMAN requires the SYSNAM privilege to run. Make sure that you have the

appropriate privilege and re-enter the command.

REQSYSPRV, Operation requires SYSPRV privilege

HITMAN requires the SYSPRV privilege to run. Make sure that you have the appropriate privilege and re-enter the command.

REQWORLD, Operation requires WORLD privilege

HITMAN requires the WORLD privilege to run. Make sure that you have the appropriate privilege and re-enter the command.

RESEQEVENTS, Events have been resequenced

This informational message indicates that the HITMAN events have been resequenced.

RLINV, Right and left arrows invalid, press return to go to next field

You cannot use right and left arrows at this point.

RUNNING, HITMAN is already running

You specified HITMAN/START, and the detached process was already running.

SECGEWAIT, Second warn time is greater than or equal to wait time

The second warning time must be less than the wait time. Please check for typing mistakes, and re-enter the command.

SERVEREXISTS, Server: *server_name* already exists in Server protect list

You have tried to enter a duplicate server name. Please check for typing or spelling mistakes, and re-enter.

SERVERNOTFOUND, Server: *server_name* not found in Server protect list

A server ID that you tried to remove from the list does not exist. Please check for typing or spelling mistakes, and re-enter.

SERVNAMTOLNG, Specified server name was too long, truncated to 256 characters

The server name you specified was too long. Server name was truncated to 256 characters. Please check for typing or spelling mistakes, and re-enter the command.

SETIDLEEXISTS, User already exists in set idle list

The username you specified already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

STATEREQ, State required for this event.

You have entered an event which requires a STATE. Please re-enter the event with a valid STATE.

STOP, Hitman is stopping during a data collection

This message informs you that HITMAN was asked to shut down during a data collection. No action is required as the message is informational purposes.

SUBMITJOBNF, Specified submit job was not found, check spelling or create file

You specified a job to be submitted as an event action but the file does not exist.

TABLESIZE, Table size: xxx, MAXPROCESSCNT: xxx

This message is informational and is generated whenever Hitman is started to indicate how many processes it has been configured to monitor.

TBLSPCEXCEEDED, Number of processes has exceeded available space

This message is *critical* and indicates that the number of processes on the system has exceeded the table space defined within Hitman. Either increase the percentage value specified for /table_size (values are from 1 to 99) or set /table_size to zero (0) which allocates space for all possible processes on the system, stop hitman and restart it. Do not continue running Hitman after this error without updating the table size and restarting it.

TERMEXISTS, Terminal: *terminal* already exists in Terminal protect list

The terminal you specified using the /PROTECT/TERMINAL qualifiers already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

TERMINALREQ, Terminal required for this event

A terminal name is required for this event. Please check for typing or spelling mistakes, and re-enter the command.

TERMNAMTOLNG, Specified terminal name was too long, truncated to 7 characters

The terminal name you specified was too long. Terminal name truncated to 7 characters. Please check for typing or spelling mistakes, and re-enter the command.

TERMNOTFOUND, Terminal: *terminal* was not found in Terminal protect list

The terminal you specified in the /PROTECT/NOTERMINAL qualifiers does not exist in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

TEXTTOLNG, Specified text was too long, must be less than 60 characters

The text you specified was too long. Please check for typing or spelling mistakes, and re-enter the command.

TIMETOLNG, Specified time too long, use HH:MM format

The time you specified was too long. Please check for typing or spelling mistakes, and re-enter the command.

TOOMANYAUTH, User: *username* not added to Authorize list, too many users specified

More than the maximum number of authorized users were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer users.

TOOMANYAUTO, Image: *image_name* not added to Autohit list, too many images specified

More than the maximum number of autohit images were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer images.

TOOMANYEVEN, Too many events specified

More than the maximum number of events were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer events.

TOOMANYHOL, Holiday: *date* not added to Holiday list, too many holidays specified

More than the maximum number of holidays were specified. Maximum number is the parameter MAX_HOLIDAYS, currently 30. Please specify fewer holidays.

TOOMANYIDENT, Identifier: *identifier_name* not added to list, too many identifiers

More than the maximum number of protected identifiers were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer identifiers.

TOOMANYIDISC, Image: *image_name* not added to list, too many images disconnects

More than the maximum number of images were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer images.

TOOMANYIDLE, User: *username* not added to list, too many idle users specified

More than the maximum number of set idle users were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer users.

TOOMANYIMAGE, Image: *image_name* not added to Image list, too many images specified

More than the maximum number of protected images were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer images.

TOOMANYIMG, Image: *image_name* not added to Image protect list, too many images

More than the maximum number of protected images were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer images.

TOOMANYINDIV, User: *username* not added to list, too many individuals specified

More than the maximum number of users were specified with the qualifier /USER. Maximum number is the parameter MAX_INDIV_LEN, currently 256. Please specify fewer individuals.

TOOMANYPRIVS, Too many privileges are in the event to modify

You specified more than seven privileges in a PRIV_CHECK event which allows a

maximum of seven. Please specify seven or fewer privileges.

TOOMANYPROCESS, Process: *process_name* not added to list, too many processes

More than the maximum number of processes were specified with the qualifier /PROCESS. Maximum number is the parameter MAX_INDIV_LEN, currently 256. Please specify fewer processes.

TOOMANYPROCS, MAXPROCESSCNT is greater than 1000, please call Saiga Systems

Your SYSGEN parameter, MAXPROCESSCNT, is greater than 1000 and HITMAN only handles 1000 processes. Please call Saiga Systems and we can customize HITMAN to suit your needs.

TOOMANYPROT, User: *username* not added to list, too many usernames specified

More than the maximum number of protected users were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer users.

TOOMANYRUN, Too many runaway processes are already defined

More than the maximum number of runaway processes were specified. Maximum number is the parameter MAX_LIST_LEN, currently 8. Please specify fewer runaway processes.

TOOMANYSERVER, Server: *server_name* not added to list, too many servers specified

More than the maximum number of servers were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer servers.

TOOMANYSETIDL, User not added to list, too many usernames specified

More than the maximum number of set idle users were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer users.

TOOMANYUDISC, User: username not added to list, too many user disconnects specified

More than the maximum number of users were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer users.

TOOMANYTERM, Terminal: *terminal* not added to list, too many terminals specified

More than the maximum number of terminals were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer terminals.

TOOMANYUDISC, User: username not added to list, too many user disconnects specified

More than the maximum number of users were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer users.

TOOMANYUIC, UIC: *[g,m]* not added to list, too many UIC's specified

More than the maximum number of UICs were specified. Maximum number is the parameter MAX_LIST_LEN, currently 256. Please specify fewer UICs.

TYPEREQ, TYPE qualifier required with /EVENT

You have tried to enter an event without the TYPE qualifier. Please re-enter the event with this qualifier.

UDISCNOTFOUND, User: *username* was not found in the user disconnect list

The user you specified using the /DISCONNECT/USER qualifiers does not exist in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

UICEXISTS, UIC: *[g,m]* already exists in the UIC Protect list

The UIC you specified using the /PROTECT/UIC qualifiers already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

UICNOTFOUND, UIC: *[g,m]* was not found in UIC list

The UIC that you specified in the /NOUIC qualifier does not exist in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

UNKWNPERMVER, Unknown permanent data file version

HITMAN has tried to read a permanent data file for which it cannot establish the version. If you have just installed a new version of HITMAN, you need to convert your permanent data file to the current version. Please refer to the section on conversions programs in this manual.

UPDPATHCONNECT
UPDPATHWAIT
UPDPROCCONNECT
UPDPROCWAIT
UPDSERVCONNECT
UPDSERVWAIT
UPDTERMCONNECT
UPDTERMWAIT
UPDUSERCONNECT
UPDIDENTCONNECT
UPDIDENTWAIT
UPDUICCONNECT
UIDUICWAIT

Connect: *value* already exists, connect times updated
Wait: *value* already exists, wait times updated

Each of the above messages indicates that an entry that was being added to a connect time limit table or a special idle and warn time limit table already existed in that table. The message indicates that the entry was updated to reflect the /first_warn, /second_warn and /wait values specified on the command line. This message may indicate a mistake was made entering the value or, if the value was correct, it is simply an informational message.

UPINTCHANGE, Invalid update interval, value has been modified

The update interval you entered was not a multiple of the data collection interval. This value has been rounded up to the nearest multiple of the data collection interval. No action necessary.

USERNAMTOLNG, Specified user name is more than 12 characters, please re-enter

The username you specified was too long. Username was truncated to 12 characters. No action necessary.

USERNOTFOUND, User: *username* was not found in user protect list

The username you specified in the /PROTECT/NOUSER= or /AUTHORIZED/NOUSER qualifiers does not exist in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

USERREQ, User required for this event

A username is required for this event. Please re-enter event with a username.

USRDISCEXISTS, User already exists in the user disconnect list

The user you specified using the /DISCONNECT/USER qualifiers already exists in the permanent data file. Please check for typing or spelling mistakes, and re-enter the command.

VERIFYNODE, Please verify the specified node name, it is over 6 characters in length

Although Digital provides 15 characters, only 6 characters are currently allowed for nodenames. Please check for typing or spelling and typing mistakes.

VERSION, Saiga Systems - HITMAN 7.0.4

You are running HITMAN V7.0.4. No action is necessary.

WAITFORPERM, Data file is currently locked, please reissue command

HITMAN cannot access the permanent data file. It may be locked by another user or have an incorrect protection. Please check that the file protection for system is "S:RWED", wait a few seconds, and re-enter the command.

WAITTOOHIGH, Wait time specified is too high

The wait time you specified is greater than 999. Please specify a number less than or equal to 999 and re-enter the command.

WAITTOOLOW, Wait time specified is too low

The wait time you specified is lower than 5 minutes. Please specify a number greater than or equal to 5, and re-enter the command.

WARNGEWAIT, Warn time is greater than or equal to wait time

The wait time you specified is greater than or equal to the time specified for warning time.

Please specify a greater time for the wait, or a smaller time for warning, to ensure that the process will get a warning.

WARNGEWARN, First warn time is greater than or equal to second warn time

The first warning time you specified is greater than or equal to the second warning time. If omitted, the current system default is used.

WATCHNODEREQ, Watch node required for this event

An event that monitors another node was entered abd this event requires you to specify which node to watch. Please re-enter the event with a valid watch node.

WILDTERM, Terminals specified with wildcards should not have colons :

The terminal you specified contains both a wildcard and a colon. It can only contain one or the other.

WRNGPERMVERS, Incorrect permanent data file, wrong version number

HITMAN has just tried to read a permanent data file that is the incorrect version. If you have just upgraded HITMAN, you need to convert your permanent data file to the current version. Please refer to the section on conversion programs for more information.

WRONGQUETYP, Wrong queue type specified, expected batch queue

You have entered a queue that is not for batch jobs. Please check for typing or spelling mistakes, and re-enter the command.

YORN, Please enter a Y or a N

You must enter either "Y" or "N" in response to this question. Please check for typing or spelling mistakes, and re-enter the command.

Appendix B, Moving HELP text

This appendix shows you how to move HITMAN HELP screens between your system help library and a private, user help library. You would do this if you had previously installed HITMAN help into the system help library and did not want public access to the help screens.

Moving HELP from system to user library
To move the HITMAN HELP screens from the system HELP library to a private, user help library:

1. Delete help text from the system help library:

```
$ LIBRARY/DELETE=HITMAN SYS$HELP:HELPLIB.HLB
```

2. Define a user help library logical.

2.1 Enter the command "$ SHOW LOGICAL HLP$*" to find a user library logical name that has not previously been assigned (logical names may be HLP$LIBRARY, HLP$LIBRARY_1, HLP$LIBRARY_2,...,HLP$LIBRARY_n)

2.2 Assign the help logical as follows:

```
$ DEFINE/SYSTEM HLP$LIBRARY_n HITMAN_DOC:HITMAN.HLB
```

If you define the logical in the SYSTEM table, everyone will be able to access the HELP. If you define it in your PROCESS table, only you will be able to access the HELP. You should add a line to your LOGIN.COM if you use the PROCESS table.

Moving HELP from user to system library
To move the HITMAN HELP screens from a private, user help library to the system help library:

1. Find the logical pointing to the user help library for HITMAN and deassign it.

```
$ SHOW LOGICAL HLP$LIBRARY*
$ DEASSIGN HLP$LIBRARY_n
```

2. Add HITMAN help library to system help library.

```
$ LIBRARY/REPLACE/HELP SYS$HELP:HELPLIB.HLB -
_$ dev:[HITMAN.DOC]HITMAN.HLP
```

Appendix C, Moving command definitions

This appendix shows you how to install the HITMAN command definition in your system DCLTABLES or access them for the HITMAN directory. You would do this if you had previously installed the HITMAN command definition in the system command definition library and did not want public access to the command definition.

To install the command definition in the system table, type:
To install the HITMAN command definitions in the system table:

1. Check which version of the table is installed

```
$ INSTALL
INSTALL> LIST SYS$LIBRARY:DCLTABLES.EXE
INSTALL> EXIT
```

2. Replace the HITMAN command definitions in the table:

```
$ SET COMMAND/TABLE=SYS$LIBRARY:DCLTABLES.EXE -
_$ /OUTPUT=SYS$LIBRARY:DCLTABLES.EXE -
_$ /REPLACE HITMAN_CDU:HITMAN
```

3. Install the new table:

```
$ INSTALL
INSTALL> REPLACE SYS$LIBRARY:DCLTABLES.EXE
INSTALL> EXIT
```

4. Check that the new version of the table is installed and verify that it still has the Open Hdr and Shar characteristics.

```
$ INSTALL
INSTALL> LIST SYS$LIBRARY:DCLTABLES.EXE
INSTALL> EXIT
```

5. Verify that DCLTABLES has world read and execute access. Set this access if required.

```
$ DIR/PROTECTION SYS$LIBRARY:DCLTABLES.EXE
$ SET FILE/PROTECTION=WORLD:RE SYS$LIBRARY:DCLTABLES.EXE
```

6. Log out and back in to get a copy of the new table.

To remove the command definition from the system table, type:

```
$ SET COMMAND/DELETE=HITMAN/TABLE=SYS$LIBRARY:-
_$ DCLTABLES.EXE/OUTPUT=SYS$LIBRARY:DCLTABLES.EXE
```

To access the command definition from the HITMAN directory, type:

```
$ @HITMAN_COM:INSTALL_CDU
```

Appendix D, Removing HITMAN from your system

This appendix shows you how to remove HITMAN from your system if you received a demo, decided to return it, and want to remove HITMAN from your system.

Stop HITMAN
The first step in removing HITMAN from your system is to stop the detached process. Type the following command:

```
$ HITMAN/STOP
```

Delete directory
To remove the HITMAN directories from your system:

```
$ DELETE SYS$SYSDEVICE:[HITMAN...]*.*;*/EXCLUDE=(*.DIR)
```

This removes all non-directory files from the directory.

To delete the HITMAN directory file from your system, you can set the protection to allow delete.

```
$ SET FILE/PROTECTION=S:RWED SYS$SYSDEVICE:[HITMAN...]*.DIR
$ DELETE SYS$SYSDEVICE:[HITMAN...]*.*;*
```

Remove HELP screens
If help screens were added into the system help library, remove them by typing:

```
$ LIBRARY/DELETE=HITMAN SYS$HELP:HELPLIB.HLB
```

If help screens were added into a private, user help library, deassign the library logical name associated with the HITMAN help library. Enter the command "$ SHOW LOGICAL" to determine the library logical name that was previously assigned (logical name may be HLP$LIBRARY, HLP$LIBRARY_1, HLP$LIBRARY_2..., HLP$LIBRARY_n), then deassign as follows:

```
$ DEASSIGN/SYSTEM HLP$LIBRARY_n
```

Remove command definition
If the HITMAN command verb was added to the DCLTABLES, remove it by typing:

```
$ SET COMMAND/DELETE=HITMAN/TABLE=SYS$LIBRARY:-
```

```
_$ DCLTABLES.EXE/OUTPUT=SYS$LIBRARY:DCLTABLES.EXE
```

Note: Remember to remove the command:

```
@HITMAN_COM:INSTALL_CDU
```

from your LOGIN.COM and any other command procedures.

Deassign system logical names
Deassign the system logicals required by HITMAN:

```
$ DEASSIGN/SYSTEM HITMAN_CDU

$ DEASSIGN/SYSTEM HITMAN_COM

$ DEASSIGN/SYSTEM HITMAN_DAT

$ DEASSIGN/SYSTEM HITMAN_DOC

$ DEASSIGN/SYSTEM HITMAN_EXE
```

Remove SYS$MANAGER files
Delete the files HITMAN_STARTUP.COM and HITMAN_SYSTEM_LOGICALS created by the installation procedure from the SYS$MANAGER directory and remove the line that executes this procedure from SYS$MANAGER:SYSTARTUP_V5.COM (SYS$MANAGER:SYSTARTUP_VMS.COM under OpenVMS 6.x).

Appendix E, Virtual terminals

This appendix describes how the OpenVMS virtual terminal feature interacts with the /DISCONNECT feature. It also describes how to set up "secure" terminals. While Virtual terminals are available on both VAX and AXP platforms it should be noted that the /DISCONNECT feature of Hitman is only available on VAX because of changes to the Operating System kernel which made implementing this feature impractical on AXP systems.

Virtual terminals allow you to maintain more than one disconnected process at a time. You must have the virtual terminal feature enabled to use the HITMAN /DISCONNECT feature.

Once you have enabled terminals as virtual, they can also be made "secure" which means that users must press the BREAK key before they log in. This prevents "password grabbers" from operating and allows users to disconnect from within an application by pressing the BREAK key. The "secure" feature is **not** required to use HITMAN/s /DISCONNECT feature.

To enable all terminals as virtual terminals:

1. Modify SYSTARTUP_V5.COM (SYSTARTUP_VMS.COM for OpenVMS 6.x)
   1.1 Add the following command:
   VAX $ MCR SYSGEN CONNECT VTA0/NOADAPTER/DRIVER=TTDRIVER

   Add the following command on the AXP:
   AXP $ MCR SYSMAN IO CONNECT VTA0 /NOADAPTER -
   /DRIVER=SYS$TTDRIVER

   1.2 Set the disconnect bit in the SYSGEN parameter TTY_DEFCHAR2.
   The disconnect bit is indicated by the HEX value (%X20000), add this value to your existing TTY_DEFCHAR2 to set it. You should note that the value displayed by SYSGEN is in DECIMAL.

   1.3 If you want to use the secure feature as well, set the secure bit in the SYSGEN parameter TTY_DEFCHAR2. The secure bit is indicated by the HEX value (%X10000), add this value to your existing TTY_DEFCHAR2.

2. Reboot your system

To enable specific terminals as virtual terminals:

1. Modify SYSSTARTUP_V5.COM (SYSTARTUP_VMS.COM for OpenVMS 6.x):
   1.1 Add the following command:

VAX $ MCR SYSGEN CONNECT VTA0/NOADAPTER/DRIVER=TTDRIVER
AXP $ MCR SYSMAN IO CONNECT VTA0 /NOADAPT /DRIVER=SYS$TTDRIVER

1.2 Add the following command for each virtual terminal:

$ SET TERMINAL/PERMANENT/DISCONNECT *terminal_name*:

1.3 If you want to use the secure feature as well,
add the following command for each virtual terminal:

$ SET TERMINAL/PERMANENT/DISCONNECT/SECURE *terminal_name*:

Using virtual terminals on a server

If you want to use virtual terminals on a server (LTA device), you must enable all terminals as virtual terminals. If you want to have a SECURE LTA terminal, you must allow the terminal to handle the BREAK key rather than the LAT. To do this, enter the following command at the LOCAL> prompt: SET PORT BREAK REMOTE

When a user presses the BREAK key, the process will be disconnected and the user returned to the LOCAL> prompt. To reconnect to the disconnected process, users must reconnect with the same username and ATTACH to the process.

Appendix F, Relinking HITMAN after Operating System Upgrades or with Tracebacks Enabled, at the Request of Saiga Staff

If HITMAN must ever be relinked because of an operating system upgrade or at the request of a member of Saiga System's Technical Staff, follow the procedure outlined below.

The command procedures necessary to do this are provided in the HITMAN_COM directory. Please follow the steps given below:

1. Set your default to the command procedure directory for HITMAN:

    $ **SET DEFAULT HITMAN_COM:**

2. Check if HITMAN is installed as a root level directory on sys$sysdevice. To check this:

    $ **SHOW LOGICAL HITMAN***

3. The command procedure DEFINE_LOGICALS.COM defines logicals for all the directories the link may access including the definition, library, object and executable directories. It attempts to determine intelligent default values by translating one of the HITMAN system logicals and parse out the appropriate values. Execute this procedure and enter the appropriate values or <enter> to accept the default displayed. For example if you installed HITMAN on disk DISK2 in the UTILITIES directory you would:

```
$ @HITMAN_COM:DEFINE_LOGICALS
Enter Disk name [DISK2:]:
Enter Directory name [UTILITIES.HITMAN]:
Enter the product name [HITMAN]:
Do you want the logicals defined for VAX box (TRUE/FALSE) [TRUE]:
```

4. Use one of the following commands depending on whether or not you are enabling tracebakcs:

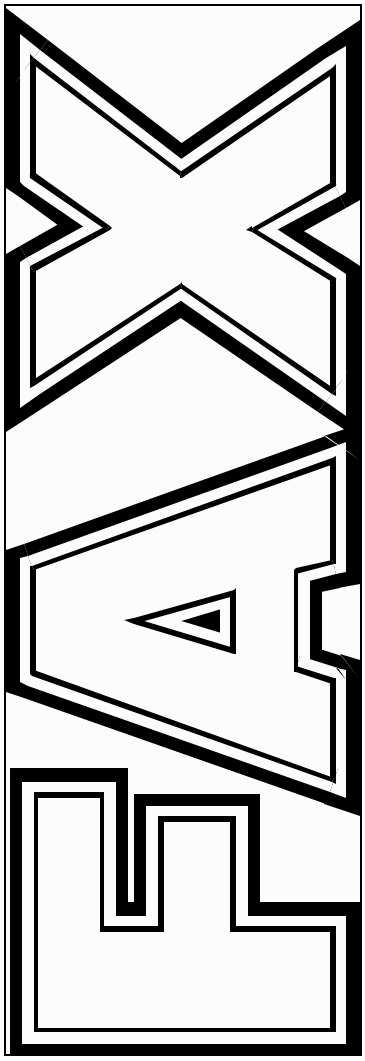
    $ @hitman_com:define_symbols hitman trace (enabled)
    $ @hitman_com:define_symbols hitman (disabled)

5. Execute the appropriate command procedure to relink HITMAN:

    $ @CUSTOMER_LINK (to relink the product)

NOTE: It is normally not necessary to relink HITMAN when you upgrade your operating system. The only known exceptions are when upgrading from the OpenVMS V5.x series to V6.x or V6.x to V7.x because of significant changes in the Operating System Kernel and system services run-time libraries.

INDEX



T R A N S M I T T A L

To: Saiga Systems, Technical Support

From:

Date:

Pages:

Re: **Reader's Comments**

Did you find this document understandable, usable and well organized? Please make suggestions for improvement.

Did you find errors in this document? If so, specify the error and the page number.

Saiga Systems Software
#215, 801 - 6 Street SW
Calgary, AB Canada
T2P 3V8

Phone: (403) 263-1151
(800) 561-8876
Fax: (403) 263-0744

Internet: support@saiga.com
http://www.saiga.com/