

HPUX I/O Subsystem EVM Events



1	Introduction	2
2	I/O Terms and Definitions	2
3	EVM Eventing Mechanism	3
4	Components of an EVM Event	4
4.1	Standard Data	4
4.2	Variable Data	5
5	EVM Events Generated by I/O Subsystem	6
5.1	I/O Node State Change Event	7
5.2	Property Modify Event	8
5.3	Health Property Modify Event	10
5.4	I/O Node Instance Number Change	11
5.5	Replacement of a Stale Device Special File with a New Device Special File	12
5.6	Automatic PCI Error Recovery	13
6	References	15

1 Introduction

The HP-UX I/O subsystem uses the Event management system (EVM) mechanism to report changes in I/O components' state. The EVM subsystem provides services to distribute, store and retrieve this information. This document provides the details of the EVM events generated by the I/O subsystem. It covers the circumstances during which the event is generated, the format of the events, and the information contained in the event. A generic example of the EVM subsystem is also provided.

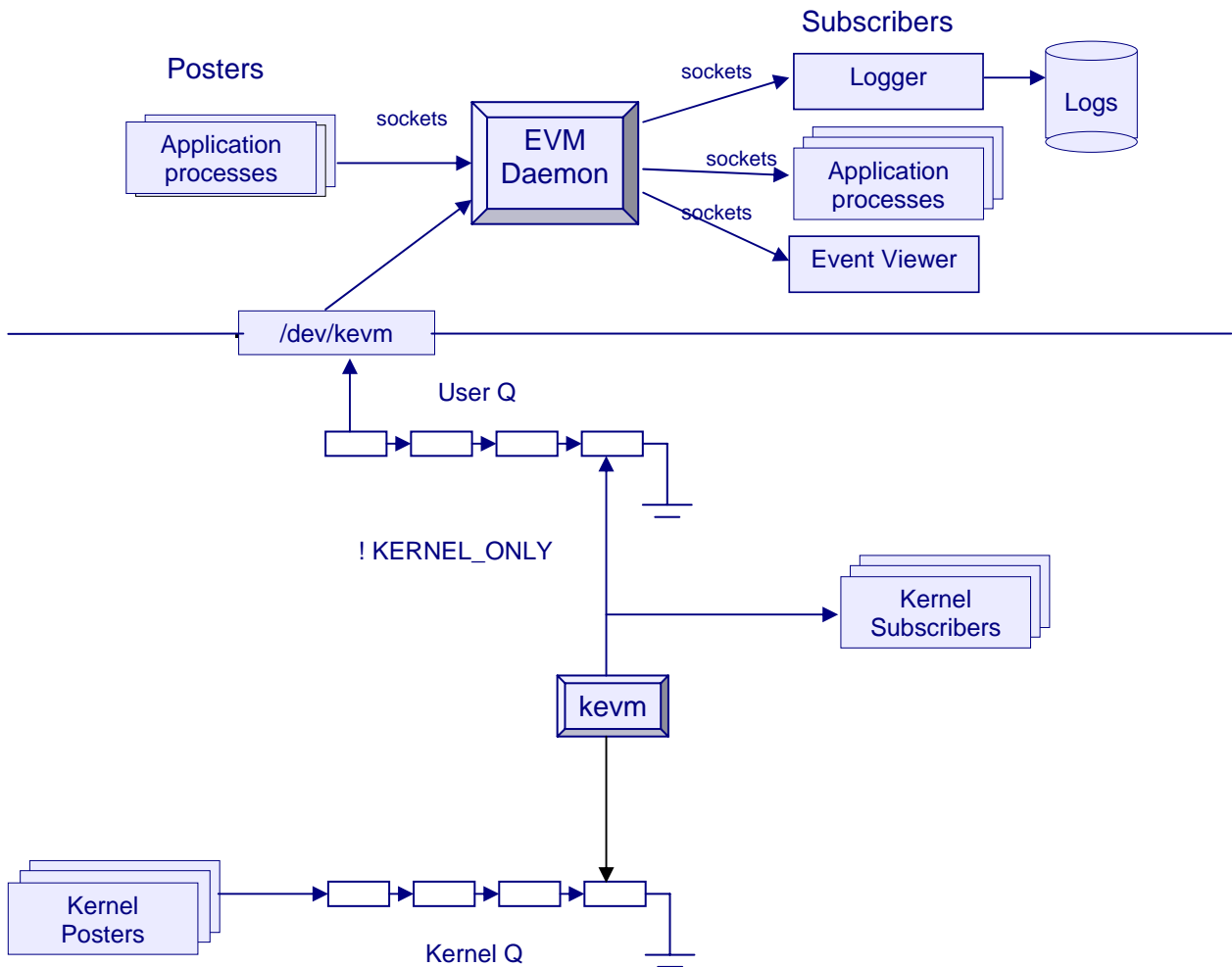
The document is intended for use by the subscribers of the EVM events. One such subscriber to I/O subsystem generated EVM events is the *setboot* command.

2 I/O Terms and Definitions

The following table contains the I/O terms and definitions that are used in the document.

Terms	Definitions
I/O node	A component in the I/O tree that corresponds to a physical or virtual device.
I/O Tree	A runtime kernel I/O data structure on system's device configuration. There are two views of the I/O Tree in HP-UX 11i v3: legacy I/O view and agile I/O view.
Legacy I/O Node	A component in the I/O Tree representing a physical or virtual I/O device in the legacy view.
Agile I/O Node	A component in the I/O Tree representing a physical or virtual I/O device in the agile view.
Stale I/O Node	A stale node corresponds to the entries that are present in the I/O configuration file but a corresponding device is not found.
Class	Set of I/O nodes that fall under the same category of devices, such as disk class or tape class.
Instance	A uniquely assigned number of an I/O node within a class.
Device Special File (DSF)	A special file used to communicate with a device driver.
EVM	Event Management System.
Hardware Path	Each I/O node is assigned a hardware address that is unique to the node. The hardware path is a sequence of I/O nodes that share a hierarchical relationship.
Legacy Hardware Path	Hardware path of an I/O node in the legacy view of the I/O tree.
Agile Hardware Path	Hardware path of an I/O node in the agile view of the I/O tree.
Property of I/O node	A property is a name and the data value associated with the I/O node.
WBEM	Web-Based Enterprise Management. An industry-wide standards-based initiative to aid the management of large scale systems

3 EVM Eventing Mechanism



An EVM subsystem is a means through which any system component or application program can report the status of an operation. The EVM subsystem consists of an EVM daemon which is a resident process responsible for distributing the events to the subscribing processes. The component that makes this information available to the EVM daemon is known as an Event poster. The component that is interested in the posted information is known as the Event Subscriber.

An Event Poster must declare an EVM template before posting the event. An EVM template consists of the event name and other data items whose values are consistent across the instances. The EVM daemon matches the posted event with the event template. If the EVM template is not available it returns an error. If the EVM template is available, the Event poster appends the variable data such as the class, instance, and so forth, and the posting API appends the environmental information such as the username and timestamp. This whole package of data consisting of the standard data from the template, the variable data and the environmental information is known as the Event. This event is distributed to the subscribing clients.

Once the Event Poster posts the event to the EVM daemon, the EVM daemon notifies the subscribed clients of the occurrence of the event. For example, subscribers can be system administrators or other kernel subsystem components. An event posted in the kernel space can be subscribed by kernel as well as user processes. User space events cannot be subscribed by the kernel processes. The EVM API library provides the event name matching mechanisms that can be used by the Event subscribers. EVM filters can be used to filter events based on the interest. The Event Subscriber then determines what action to take with the notified event.

4 Components of an EVM Event

An EVM event consists of the following fields.

- 1) The standard data or the fixed data component:
The value of the standard data remains constant across different instances of the event.
- 2) The variable data:
Variable data items vary from event to event.

A typical EVM event looks as follows.

Event Full Name: sys.unix.io.state_change.new_state._class.ccc._instance.iii. _cim.HP_EVMEventIndication
Standard Data: Event Name: sys.unix.io.state_change.new_state Priority: 200 Format "An I/O node(name = \$node_name) at hardware path(\$hw_path) has changed state"
Variable Data: node_name type STRING value "" class type STRING value "" _instance type UINT64 value 0 _cim type STRING value "" hw_attr type EvmTYPE_OPAQUE value hw_path type STRING value ""

The fields of the EVM event are explained below.

4.1 Standard Data

The standard data of an EVM event consists of the following:

Name The name of an event indicates the origin of an event and the reason for which the event was generated. An event name is a series of one or more components separated by dots. There is no restriction on the number of components in an event.

For example consider the event name:

```
sys.unix.io.state_change.suspended._class.ext_bus._instance.5
```

sys.unix Indicates that the event is a system generated event.

io Indicates that the event is generated by the I/O subsystem.

state_change Indicates that the event is generated because the I/O node has changed its state.

suspended	Indicates that the I/O node is suspended.
_class.ext_bus	Indicates that the I/O node belongs to the class ext_bus.
_instance.5	Indicates that the I/O node instance is 5.

Format A human readable string that summarizes the event.

Priority A description of the order in which the events are delivered to the subscribing clients.

Note: The I/O subsystem mainly uses the **Name**, **Format** and **Priority** fields in its posted events. For more information on the standard fields of an EVM event see the *HP-UX Event Manager Programmer's Guide: HP-UX 11i v3, Edition 1*, available at <http://docs.hp.com/en/5991-6661/index.html>.

4.2 Variable Data

Variable data can take different values for different instances of the generated event. Listed below are the variable data items that are used in the events posted by the I/O subsystem.

Node_name	The driver of the I/O node.
Class	The class to which the I/O node belongs.
Instance	The instance number of the I/O node.
Pname	The name of the property that is modified, for example class, disk.
Pvalue	The value of the property of the I/O node.
Plen	The length of property field.
Cim type	Applicable for events that must be converted to WBEM indication.
Hw_path	The hardware path of the I/O node.
Stale_hwpath	The hardware path of the stale node.
Stale_devt	The device number of the stale node.
Stale_dsf	The device special file of the stale node.

5 EVM Events Generated by I/O Subsystem

The following table gives the events that are generated by the I/O subsystem based on the operation initiated on an I/O node.

I/O command	Events generated	Explanation of the event
Destroy legacy node (rmsf <legacy node>)	sys.unix.io.state_change.unclaimed	See 5.1
Destroy agile node (rmsf <agile node>)	sys.unix.io.state_change.unclaimed sys.unix.io.health.offline._class.disk sys.unix.io.state_change.unclaimed._class.lunpath sys.unix.io.health.disabled._class.lunpath	See 5.1, 5.3
Scan legacy H/W path (ioscan unclaimed_ legacy_node)	sys.unix.io.state_change.claimed	See 5.1
Scan agile H/W (ioscan unclaimed_agile_ node)	sys.unix.io.state_change.claimed sys.unix.io.health.online._class.disk sys.unix.io.state_change.claimed._class.lunpath sys.unix.io.health.online._class.lunpath	See 5.1, 5.3
Suspend an I/O slot (olrad -r slot_id)	sys.unix.io.health.disabled sys.unix.io.state_change.suspended	See 5.1,5.3
Resume an I/O slot (olrad -R slot_id)	sys.unix.io.health.online sys.unix.io.state_change.claimed	See 5.1,5.3
Change H/W instance (ioinit -f infile legacy_node)	sys.unix.io.instance_change	See 5.4
Change H/W instance agile path (ioinit -f infile agile_node)	sys.unix.io.instance_change. sys.unix.io.health.online	See 5.3,5.4
Replace stale DSF with new DSF (io_redirect_dsf -d old_dsf -n new_dsf)	sys.unix.io.devt_changed	See 5.5
Automatic PCI error recovery (success)	sys.unix.io.pci_er.succeeded sys.unix.io.state_change.claimed	See 5.6

5.1 I/O Node State Change Event

An I/O node is said to change its state when the software state of the node changes. An I/O node can assume the following software states:

CLAIMED	The I/O node is bound to a driver.
UNCLAIMED	The I/O node is not bound to any driver.
UNUSABLE	The hardware at this address has become unusable as a result of some error condition.
SUSPENDED	The associated hardware and software are in a suspended state.
NO_HW	The hardware at this address is no longer responding to scan requests.
ERROR	The hardware at this address is in an error state.
REMOVED	The hardware at this address has been removed.

An EVM event is generated when the state of the I/O node is changed to one of the above mentioned states.

Given below is the format in which the I/O node state change event is generated:

Event Full name: sys.unix.io.state_change.new_state._class.ccc._instance. iii._cim.HP_EVMEventIndication
Standard data: Event Name: sys.unix.io.state_change.new_state Priority: 200 Format "An I/O node(name = \$node_name) at hardware path(\$hw_path) has changed state"
Variable Data: node_name type STRING value "" _class type STRING value "" _instance type UINT64 value 0 _cim type STRING value "" hw_attr type EvmTYPE_OPAQUE value hw_path type STRING value ""

Example: Suspend the driver associated with the card using the command "olrad -r <slot_id>". This command prepares a slot for replacement. The associated driver of the card is suspended. For more information on the *olrad* command, see the *olrad(1M)* manpage.

```
# olrad -r 3
```

The execution of this command causes the software state of the I/O node to change to **SUSPENDED**. As a result an EVM event is generated. Given below is the output obtained from the *evmshow* command.

EVM event generated after an I/O node changes state	
Formatted Message:	An I/O node "c8xx" at hardware path "0/4/1/1" has changed state.
Event Data Items:	Event Name: sys.unix.io.state_change.suspended._class.ext_bus._instance.5 Priority : 200 PID : -1 PPID : -1 Event Id : 1077 Timestamp: 19-Feb-2008 11:35:18 Format : An I/O node "\$node_name" at hardware path "\$hw_path" has changed state
Variable Items:	node_name (STRING) = "c8xx" class (STRING) = "ext_bus" _instance (UINT64) = 5 hw_path (STRING) = "0/4/1/1" _cim (STRING) = ""

5.2 Property Modify Event

Property is a way of associating name and data value with the I/O nodes. Some of the properties that can be associated with a node are health, minor number, major number, class, instance etc. Whenever one of these property is modified a property modify event is generated.

Given below is the format in which the property modify event is generated.

Event Full Name:	sys.unix.io.property.modify._pname.nnn._class.ccc._instance.iii._cim.HP_EVMEventIndication
Standard Data:	Event Name: sys.unix.io.property.modify Priority: 200 Format: "The property of I/O node(name = \$node_name) has been modified"

Variable Data:

```
node_name type STRING value ""
pname type STRING value ""
pvalue type EvmTYPE_OPAQUE value
plen type UNIT64 value 0
class type STRING value ""
_instance type UINT64 value 0
_cim type STRING value ""
hw_attr type EvmTYPE_OPAQUE
hw_path type STRING value ""
```

Example: The following property modify EVM event is generated when an *rmsf* is performed on a hardware path. The "id bytes" property of the *tgtpath* associated with the I/O node is modified.

EVM event generated after the property of I/O node is modified**Formatted Message:**

A property of I/O node "estp" has been modified

Event Data Items:

```
Event Name:
sys.unix.io.property.modify._pname.id._class.tgtpath._instance.0
Priority   : 200
PID       : -1
PPID     : -1
Event Id  : 81182
Timestamp : 21-Feb-2008 16:46:33
Format    : A property of I/O node "$node_name" has been modified
```

Variable Items:

```
node_name (STRING) = "estp"
_class (STRING) = "tgtpath"
_instance (UINT64) = 0
hw_path (STRING) = "0/0/3/0.0.0x0"
_pname (STRING) = "id"
pvalue (OPAQUE) = [OPAQUE VALUE: 16 bytes]
plen (UINT64) = 16
_cim (STRING) = ""
```

5.3 Health Property Modify Event

The health property of an I/O node denotes the state of the node as defined by the subsystem that manages this node (for example, the driver). The values that the health property can take are:

Online	The I/O node is online and is ready to issue or receive I/O transfers.
Offline	The I/O node is offline and cannot be accessed.
Limited	The I/O node is operating in a performance degraded (sub-optimal) state. For example, an aggregate group (LUN or an iscsi session) has one of its links offline and one or more lunpaths to a LUN are offline.
Unusable	An error condition has occurred that requires manual intervention. Examples include authentication failure and hardware failure.
Testing	The I/O node is in a diagnostic mode.
Disabled	The I/O node is disabled or suspended.
Standby	The I/O node is functionally online but is not in use and is ready to serve as a standby in case of failure of the active in-use I/O node.

The following EVM event is generated when the health property of the I/O node is modified:

Event Full name sys.unix.io.health.value._class.ccc._instance.iii._cim.HP_EVMEvent Indication
Standard data: Event: Name sys.unix.io.health.value Priority: 200 Format: "The health property of I/O node(name = \$node_name) has been changed"
Variable Data: node_name type STRING value "" _class type STRING value "" _instance type UINT64 value 0 _cim type STRING value "" hw_attr type EvmTYPE_OPAQUE hw_path type STRING value ""

Example: When an I/O node is deleted using *rmsf*, one of the events generated is a health property change event. In this case the health property of the node is changed to **Offline**.

EVM event generated after health property of I/O node is modified	
Formatted Message:	The health property of I/O node "esdisk" has been changed
Event Data Items:	<pre> Event Name: sys.unix.io.health.offline._class.disk._instance.5 Priority : 400 PID : -1 PPID : -1 Event Id : 80913 Timestamp : 20-Feb-2008 20:38:19 Format : The health property of I/O node "\$node_name" has been changed </pre>
Variable Items:	<pre> node_name (STRING) = "esdisk" _class (STRING) = "disk" _instance (UINT64) = 5 hw_path (STRING) = "64000/0xfa00/0x2" _cim (STRING) = "" </pre>

5.4 I/O Node Instance Number Change

The instance of an I/O node can be changed using the `-f` option of the `ioinit` command. The EVM event generated is in the following format:

Event Full name:	<pre> sys.unix.io.instance_change._instance.iii._class.ccc._orig_instance.iii._cim.HP_EVMEventIndication </pre>
Standard data:	<pre> Event Name: sys.unix.io.instance_change Priority: 200 Format: "A I/O node(name=\$node_name) at hardware path(\$hw_path) has change instance number" </pre>
Variable Data:	<pre> node_name type STRING value "" _class type STRING value "" _instance type UINT64 value 0 _orig_instance type UINT64 value 0 _cim type STRING value "" hw_attr EvmTYPE_OPAQUE hw_path type STRING value "" </pre>

Example: The following event is generated when the instance number is changed using the `-f` option of the `ioinit` command. For more information see the `ioinit(1M)` man page.

EVM event generated when instance number of the I/O node is changed

Formatted Message:

An I/O node "esdisk" at hardware path "64000/0xfa00/0x2" has changed instance number

Event Data Items:

Event Name:
sys.unix.io.instance_change._instance.200._class.disk._orig_instance.5
Priority : 200
PID : -1
PPID : -1
Event Id : 81103
Timestamp : 21-Feb-2008 14:10:01
Format: An I/O node "\$node_name" at hardware path "\$hw_path" has changed instance number

Variable Items:

node_name (STRING) = "esdisk"
_class (STRING) = "disk"
_instance (UINT64) = 5
hw_path (STRING) = "64000/0xfa00/0x2"
_cim (STRING) = ""

5.5 Replacement of a Stale Device Special File with a New Device Special File

The following EVM event is generated when the dev_t of a stale DSF is replaced by the dev_t of a new DSF:

Event Full name: sys.unix.io.devt_changed.class.ccc.instance.iii
Standard data: Event Name: sys.unix.io.devt_changed Priority: 200 Format "DSF \"\$stale_dsf\" has been replaced with \"\$new_dsf\""
Variable Data: _class type STRING value "" _instance type UINT64 value 0 stale_devt type UINT64 value 0 new_devt type UINT64 value 0 _cim type STRING value "" stale_hwpath type STRING value "" new_hwpath type STRING value "" name stale_dsf type STRING value "" new_dsf type STRING value ""

Example: The following event is generated when the *io_redirect_dsf* command is used to replace the dev_t of a device special file with the dev_t of a stale device special file.

EVM event when stale device special file is replaced by a new device special file
<pre>Formatted Message: DSF "/dev/disk/disk151" has been replaced with "/dev/disk/disk288"</pre>
<pre>Event Data Items: Event Name : sys.unix.io.devt_changed._class.disk._instance.288 Priority : 200 PID : -1 PPID : -1 Event Id : 1928 Timestamp : 14-Apr-2008 11:22:10 Format : DSF "\$stale_dsf" has been replaced with "\$new_dsf"</pre>
<pre>Variable Items: _class (STRING) = "disk" _instance (UINT64) = 288 stale_devt (UINT64) = 50331839 new_devt (UINT64) = 50331700 stale_hwpath (STRING) = "64000/0xfa00/0xbf" new_hwpath (STRING) = "64000/0xfa00/0x34" stale_dsf (STRING) = "/dev/disk/disk151" new_dsf (STRING) = "/dev/disk/disk288" _cim (STRING) = ""</pre>

5.6 Automatic PCI Error Recovery

PCI error recovery is supported on HPUX systems to ensure that certain PCI errors caused by a hard partition on high end systems do not crash the system. This functionality helps in recovering most of the parity errors.

Example: The following event is generated after a successful PCI error recovery.

EVM event generated after successful Automatic PCI error recovery
<pre>Formatted Message: Base I/O event</pre>
<pre>Event Data Items: Event Name: sys.unix.io.pci_er.succeeded._class.ba._instance.11._cim.HP_DeviceIndi cation Priority : 200 PID : -1 PPID : -1 Event Id : 706 Timestamp : 14-Apr-2008 16:34:05</pre>

Format : Base I/O event
<p>Variable Items:</p> <pre> node_name (STRING) = "lba" class (STRING) = "ba" instance (UINT64) = 11 hw_path (STRING) = "1/0/6" cim (STRING) = "HP_DeviceIndication" hw_attr (OPAQUE) = [OPAQUE VALUE: 1282 bytes]</pre>

Example: The following event is generated if PCI error recovery fails.

EVM event generated when PCI error recovery fails
<p>Formatted Message:</p> <pre>Base I/O event</pre>
<p>Event Data Items:</p> <pre> Event Name: sys.unix.io.pci_er.failed._class.ba._instance.11._cim.HP_DeviceIndi cation Priority : 200 PID : -1 PPID : -1 Event Id : 706 Timestamp: 14-Apr-2008 16:34:05 Format : Base I/O event</pre>
<p>Variable Items:</p> <pre> node_name (STRING) = "lba" class (STRING) = "ba" instance (UINT64) = 11 hw_path (STRING) = "1/0/6" cim (STRING) = "HP_DeviceIndication" hw_attr (OPAQUE) = [OPAQUE VALUE: 1282 bytes]</pre>

6 References

For more information, see the following documents and man pages:

- *HP-UX Event Manager Programmer's Guide: HP-UX 11i v3 Edition 1:*
<<http://docs.hp.com/en/5991-6661/index.html>>
- *PCI Error Recovery Product Note: HP-UX 11i v3:*
<<http://docs.hp.com/en/5992-1722/index.html>>
- ioscan(1M) manpage
- rmsf(1M) manpage
- ioinit(1M) man page
- EVM(5) manpage
- evmget(1) manpage
- evmshow(1) manpage
- olrad(1M) manpage
- io_redirect_dsf(1M) manpage

© 2008 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

