HP Network Node Manager i Software

Policies for Integrating HP NNMi with HP Operations Manager

Release 9.20

This whitepaper describes HPOM policies for NNMi incidents received through the HP Operations agent implementation of the HP NNMi–HPOM integration. This whitepaper presents a brief summary of the NNMi default policy conditions and describes alternative policy approaches.

CONTENTS

Introduction

The recommended method for integrating HP Network Node Manager i Software (NNMi) with HP Operations Manager (HPOM) is to enable the NNMi northbound integration module to forward NNMi incidents as SNMPv2c traps to an HP Operations agent. This method requires an HPOM policy file to interpret the NNMi incidents so HPOM can recognize and process them.

The NNMi–provided `nnmopcexport.ovpl` script generates a default policy file for the currently configured NNMi incidents. The default policy file is a good starting point when integrating NNMi with HPOM. Many customers find the default policy file to be sufficient with little or no modification. Some customers benefit from modifying or replacing the default policy file.

This whitepaper reviews the approach for managing NNMi incidents using the default policy file and suggests alternate policy conditions that can resolve specific problems.

NNMi Incident-Specific Policy

This section describes the default integration behavior as defined in the default policy file.

In HPOM, messages are created and acknowledged based on the NNMi incident UUID (universal unique identifier). Every incident created by NNMi is unique and is identified by its UUID. If multiple traps of the same type are sent from the same node, each trap sent is considered unique. For example, if two `SnmpLinkDown` traps are sent from the same device for the same interface, NNMi generates two `SnmpLinkDown` incidents, each with a distinct UUID. The creation and acknowledgement of incidents by UUID is preserved with the default policy file.

An HPOM policy is registered with the HPOM management server and then deployed to the HP Operations agent that receives incidents from NNMi. The HPOM policy describes how the HP Operations agent interprets the NNMi traps. Traps not matching conditions specified by the policy are discarded. Traps matching conditions in the policy are transformed into HPOM messages and then forwarded to the HPOM management server.

The ability to automatically create and close messages received from NNMi is an important feature when integrating with HPOM as an event consolidator. This functionality requires the correlation of messages within HPOM. With the NNMi default policy file, message correlation is accomplished by including a `MSGKEY` attribute in the message create condition and a `MSGKEYRELATION` ACK statement in the message close condition.

Message Create

When an NNMi incident trap matches an HPOM policy condition, the trap is transformed into an HPOM message. A `MSGKEY` attribute specifies the pattern used by HPOM to correlate messages together. In an NNMi default policy file, messages are correlated using the NNMi incident UUID. The `MSGKEY` for an NNMi default message create condition must contain the NNMi incident UUID and can contain additional information to explain the purpose of the message. Figure 1 shows the pattern for the NNMi default `MSGKEY`.

```
MSGKEY   "<nnmiIncidentUuid>:Create"
```

Figure 1

When an HP Operations agent processes the NNMi incident using the HPOM policy, it substitutes the actual incident UUID in place of *nnmiIncidentUuid*. The ":Create" is additional information identifying the message as created.

Figure 11 in Appendix A shows the trap definition for an NNMi NodeDown (*nnmiMgmtEvNodeDown*) incident. The nnmiIncidentUuid variable is found in position 6 of the trap definition. When processing a trap, HPOM does not know about the variable name for a trap but can access a variable based on the position of the variable in the trap. This position, or index, is used in the policy condition. Figure 2 shows the actual MSGKEY definition used for NNMi default policy file create message conditions.

```
MSGKEY   "<$6>:Create"
```

Figure 2

Figure 3 shows a complete message create condition for the NNMi NodeDown incident. As an exercise, examine Figure 11 to identify the nnmiMgmtEvNodeDown trap variable names for the variables referenced by index in this condition (for example, $21 is *nnmiIncidentSourceNodeHostname*).

```
# from EVENT NodeDown .1.3.6.1.4.1.11.2.17.19.2.0.32 "Fault" Critical
            DESCRIPTION "NodeDown"
            CONDITION_ID "3963982f-ffe5-4b5d-bbf2-30af25a4fee0"
            CONDITION
                  $e ".1.3.6.1.4.1.11.2.17.19.2"
                  $G 6
                  $S 32
                  $2
"^<@.nnmiprotocol>://<@.nnmiserver>:<@.nnmiport>/nnm$"
            SET
                  SEVERITY Critical
                  NODE     IP 0.0.0.0 "<$21>"
                  OBJECT   "<$25>"
                  MSGKEY   "<$6>:Create"
                  MSGKEYRELATION ACK "^<$6>:<*>$" ICASE
                  CUSTOM   "nnm.incident.uuid"  "<$6>"
                  CUSTOM   "nnm.server.name"  "<nnmiserver>"
                  CUSTOM   "nnm.server.port"  "<nnmiport>"
                  CUSTOM   "nnm.name"  "<$5>"
                  CUSTOM   "nnm.priority"  "<$13>"
                  CUSTOM   "nnm.assignedTo"  "<$19>"
                  CUSTOM   "nnm.category"  "<$7>"
                  CUSTOM   "nnm.origin"  "<$9>"
                  CUSTOM   "nnm.source.name"  "<$25>"
                  CUSTOM   "nnm.source.uuid"  "<$27>"
                  CUSTOM   "nnm.source.type"  "<$26>"
                  CUSTOM   "nnm.emittingNode.uuid"  "<$22>"
                  CUSTOM   "nnm.emittingNode.name"  "<$21>"
                  CUSTOM   "RelatedCiHint" "<$43>"
                  CUSTOM   "EtiHint"  "NodeStatus:Down"
                  TEXT     "<$11>"
```

```
                    HELPTEXT "This incident indicates that NNMs Advanced
Problem Analyzer has determined the node is down based on the following
analysis: 1) 100% of the addresses assigned to this node are unreachable,
and 2) The SNMP agent installed on this machine is not responding. At
least two of the neighboring devices can be reached and are reporting
problems with connectivity to this node."
```

Figure 3

Message Close (Acknowledge)

When an incident is closed in NNMi, the NNMi northbound interface sends an `EventLifecycleStateClosed` trap to the HP Operations agent. Figure 12 in Appendix A describes the trap definition for the `nnmiEvClosed`[1] trap. The `EventLifecycleStateClosed` trap contains much of the same information available in the original NNMi incident including the UUID of the original incident. This information is used to create a message close condition in the NNMi default policy file.

Messages that are identifiable through the `MSGKEY` attribute can be correlated and acted upon when the correlation condition is satisfied. For the HP NNMi–HPOM integration, the intended action is to automatically close, or acknowledge, messages in HPOM when an incident is closed in NNMi. A `MSGKEYRELATION ACK` statement in the message close condition forms the correlation condition and specifies the acknowledge action. When the correlation condition is satisfied, the correlated message (or messages) is acknowledged.

The form of a `MSGKEYRELATION ACK` statement is similar to the `MSGKEY` attribute. For NNMi default policy file close conditions, the close condition must match messages with a `MSGKEY` composed of an NNMi incident UUID and the text "`:Create`". Figure 4 shows the `MSGKEYRELATION ACK` statement used by the NNMi default policy file.

```
MSGKEYRELATION ACK "^<nnmiIncidentUuid>:<*>$" ICASE
```

Figure 4

The statement in Figure 4 correlates and acknowledges all messages with a MSGKEY matching "*nnmiIncidentUuid***:***any-text*".

Figure 5 shows the complete message condition for closing a `NodeDown` incident. The variable in position 6 is the NNMi incident UUID of the original incident. Active messages in HPOM that match the pattern **^<$6>:<*>$** are closed on receipt of this message.

```
# from EVENT EventLifecycleStateClosed .1.3.6.1.4.1.11.2.17.19.2.0.1000
"LOGONLY" Normal
            DESCRIPTION "EventLifecycleStateClosed"
            CONDITION_ID "6d7a85f3-81f5-4daa-9279-a60c074dc34d"
            CONDITION
                  $e ".1.3.6.1.4.1.11.2.17.19.2"
                  $G 6
                  $S 1000
                  $14 "4"
                  $5 "NodeDown"
                  $2
```

[1] nnmiEvClosed and EventLifecycleStateClosed are synonymous. The HP-NNMi-NBI-MIB notification names follow a naming convention required for well-formed MIBS and are subject to length limitations.

```
"^<@.nnmiprotocol>://<@.nnmiserver>:<@.nnmiport>/nnm$"
            SET
                  NODE     IP 0.0.0.0 "<$21>"
                  OBJECT   "<$25>"
                  SERVERLOGONLY
                  MSGKEY   "<$6>:Close"
                  MSGKEYRELATION ACK "^<$6>:<*>$" ICASE
                  CUSTOM   "nnm.incident.uuid"  "<$6>"
                  CUSTOM   "nnm.server.name"  "<nnmiserver>"
                  CUSTOM   "nnm.server.port"  "<nnmiport>"
                  CUSTOM   "nnm.name"  "<$5>"
                  CUSTOM   "nnm.priority"  "<$13>"
                  CUSTOM   "nnm.assignedTo"  "<$19>"
                  CUSTOM   "nnm.category"  "<$7>"
                  CUSTOM   "nnm.origin"  "<$9>"
                  CUSTOM   "nnm.source.name"  "<$25>"
                  CUSTOM   "nnm.source.uuid"  "<$27>"
                  CUSTOM   "nnm.source.type"  "<$26>"
                  CUSTOM   "nnm.emittingNode.uuid"  "<$22>"
                  CUSTOM   "nnm.emittingNode.name"  "<$21>"
                  CUSTOM   "RelatedCiHint" "<$30>"
                  CUSTOM   "EtiHint"  "NodeStatus:Up"
                  TEXT     "Event (<$5>,<$6>) is closed.
<$21>:<$26>=<$25>"
                  HELPTEXT "NNMi event is closed."
```

Figure 5

The `MSGKEYRELATION ACK` statement appears in both the create and close message conditions. The attribute is included in the create message conditions for the unlikely case that a duplicate incident is sent. If a duplicate is sent, the previously active message is acknowledged and only the most recent message remains active.

Up Events

This section describes how to modify the NNMi default policy file for customers accustomed to receiving `NodeDown/NodeUp` and `InterfaceDown/InterfaceUp` event pairs.

Note: For NNMi 8.1x, additional modifications are required to the NNMi default policy file. When making changes to an NNMi 8.1x policy, also see NNMi 8.1x Policy Considerations on page10.

NNMi manages incident state through an incident lifecycle. Incidents move through a series of lifecycle states. (For more information on incident lifecycle states, see the NNMi help.) Ultimately, when an incident has been resolved, it is closed. An incident can be closed by NNMi Causal Engine root cause analysis, NNMi Event System correlations or actions, an NNMi console user, or an external program through the NNMi SDK. When an incident is closed in NNMi, the NNMi northbound interface sends a close trap to the HP Operations agent. HPOM processes this trap as an `EventLifecycleStateClosed` message. Some customers might want to replace this message with a more specific close message that corresponds to the situation when a node or an interface returns to an operational state that is `NodeUp` or `InterfaceUp`.

NodeUp

When the NNMi Causal Engine determines that a node has resumed normal operation after having been down, it closes the original `NodeDown` incident and assigns a close reason of `NodeUp` to the incident.

The close reason is available in the `EventLifecycleStateClosed` trap that NNMi sends. Using this information, you can create a new close condition that replaces the default policy file EventLifecycleStateClosed condition. For a description of the `EventLifecycleStateClosed` (`nnmiEvClosed`) trap definition, see Figure 12 in Appendix A. The original `EventLifecycleStateClosed` condition can be copied with the following modifications:

1. Test that trap variable #5 (***nnmiIncidentName***) is equal to `"NodeDown"`.
2. Test that trap variable #29 (***nnmiIncidentClosedReason***) explicitly matches `"NodeUp"`.
3. Optionally update the `HELPTEXT` attribute with a more descriptive message.

```
# from EVENT EventLifecycleStateClosed .1.3.6.1.4.1.11.2.17.19.2.0.1000
"LOGONLY" Normal
DESCRIPTION "NodeUp"
CONDITION_ID "886fd429-2a29-41b6-a483-5024e43aae79"
CONDITION
       $e ".1.3.6.1.4.1.11.2.17.19.2"
       $G 6
       $S 1000
       $14 "4"
       $5 "NodeDown"
       $2 "^<@.nnmiprotocol>://<@.nnmiserver>:<@.nnmiport>/nnm$"
       $29 "^NodeUp$"
SET
       SERVERLOGONLY
       NODE IP 0.0.0.0  "<$21>"
       OBJECT "<$25>"
       MSGKEY "<$6>:Close"
       MSGKEYRELATION ACK  "^<$6>:<*>$" ICASE
       CUSTOM  "nnm.incident.uuid"  "<$6>"
       CUSTOM  "nnm.server.name"  "<nnmiserver>"
       CUSTOM  "nnm.server.port"  "<nnmiport>"
       CUSTOM  "nnm.name"  "<$5>"
       CUSTOM  "nnm.priority"  "<$13>"
       CUSTOM  "nnm.assignedTo"  "<$19>"
       CUSTOM  "nnm.category"  "<$7>"
       CUSTOM  "nnm.origin"  "<$9>"
       CUSTOM  "nnm.source.name"  "<$25>"
       CUSTOM  "nnm.source.uuid"  "<$27>"
       CUSTOM  "nnm.source.type"  "<$26>"
       CUSTOM  "nnm.emittingNode.uuid"  "<$22>"
       CUSTOM  "nnm.emittingNode.name"  "<$21>"
       CUSTOM  "OPR_CI_INFO"  "UCMDB:<$28>"
       TEXT "Event (<$5>,<$6>) is closed. <$21>:<$26>=<$25>"
       HELPTEXT "NNMi NodeDown incident is closed."
```

Figure 6

Because a NodeDown incident can be closed without a close reason of NodeUp, it is a good idea to include an EventLifecycleStateClosed condition that matches close conditions other than NodeUp. The modifications are similar to the NodeUp condition. The original EventLifecycleStateClosed condition can be copied with the following modifications:

1. Test that trap variable #5 is equal to "NodeDown".
2. Test that trap variable #29 explicitly excludes "NodeUp" by using the pattern string "^<![NodeUp]>$" .
3. Optionally update the HELPTEXT attribute with a more descriptive message.

```
# from EVENT EventLifecycleStateClosed .1.3.6.1.4.1.11.2.17.19.2.0.1000
"LOGONLY" Normal
DESCRIPTION "EventLifecycleStateClosed"
CONDITION_ID "6d7a85f3-81f5-4daa-9279-a60c074dc34d"
CONDITION
      $e ".1.3.6.1.4.1.11.2.17.19.2"
      $G 6
      $S 1000
      $14 "4"
      $5 "NodeDown"
      $2 "^<@.nnmiprotocol>://<@.nnmiserver>:<@.nnmiport>/nnm$"
      $29 "^<![NodeUp]>$"
SET
      NODE     IP 0.0.0.0 "<$21>"
      OBJECT   "<$25>"
      SERVERLOGONLY
      MSGKEY   "<$6>:Close"
      MSGKEYRELATION ACK "^<$6>:<*>$" ICASE
      CUSTOM   "nnm.incident.uuid"  "<$6>"
      CUSTOM   "nnm.server.name"  "<nnmiserver>"
      CUSTOM   "nnm.server.port"  "<nnmiport>"
      CUSTOM   "nnm.name"  "<$5>"
      CUSTOM   "nnm.priority"  "<$13>"
      CUSTOM   "nnm.assignedTo"  "<$19>"
      CUSTOM   "nnm.category"  "<$7>"
      CUSTOM   "nnm.origin"  "<$9>"
      CUSTOM   "nnm.source.name"  "<$25>"
      CUSTOM   "nnm.source.uuid"  "<$27>"
      CUSTOM   "nnm.source.type"  "<$26>"
      CUSTOM   "nnm.emittingNode.uuid"  "<$22>"
      CUSTOM   "nnm.emittingNode.name"  "<$21>"
      CUSTOM   "RelatedCiHint" "<$30>"
      CUSTOM   "EtiHint"  "NodeStatus:Up"
      TEXT     "Event (<$5>,<$6>) is closed. <$21>:<$26>=<$25>"
      HELPTEXT "NNMi event is closed."
```

Figure 7

InterfaceUp

When the NNMi Causal Engine determines that an interface has resumed normal operation after having been down, it closes the original InterfaceDown incident and assigns a close reason of InterfaceUp to the incident.

The same approach used to create the `NodeUp` condition can be used to create the `InterfaceUp` condition. The original `EventLifecycleStateClosed` condition can be copied with the following modifications:

1. Test that trap variable #5 (***nnmiIncidentName***) is equal to `InterfaceDown`.
2. Test that trap variable #29 (***nnmiIncidentClosedReason***) explicitly matches `"InterfaceUp"`.
3. Optionally update the `HELPTEXT` attribute with a more descriptive message.

```
# from EVENT EventLifecycleStateClosed .1.3.6.1.4.1.11.2.17.19.2.0.1000
"LOGONLY" Normal
DESCRIPTION "InterfaceUp"
CONDITION_ID "886fd429-2a29-41b6-a483-5024e43aae79"
CONDITION
       $e ".1.3.6.1.4.1.11.2.17.19.2"
       $G 6
       $S 1000
       $14 "4"
       $5 "InterfaceDown"
       $2 "^<@.nnmiprotocol>://<@.nnmiserver>:<@.nnmiport>/nnm$"
       $29 "^InterfaceUp$"
SET
       SERVERLOGONLY
       NODE IP 0.0.0.0  "<$21>"
       OBJECT "<$25>"
       MSGKEY "<$6>:Close"
       MSGKEYRELATION ACK  "^<$6>:<*>$" ICASE
       CUSTOM  "nnm.incident.uuid"  "<$6>"
       CUSTOM  "nnm.server.name"  "<nnmiserver>"
       CUSTOM  "nnm.server.port"  "<nnmiport>"
       CUSTOM  "nnm.name"  "<$5>"
       CUSTOM  "nnm.priority"  "<$13>"
       CUSTOM  "nnm.assignedTo"  "<$19>"
       CUSTOM  "nnm.category"  "<$7>"
       CUSTOM  "nnm.origin"  "<$9>"
       CUSTOM  "nnm.source.name"  "<$25>"
       CUSTOM  "nnm.source.uuid"  "<$27>"
       CUSTOM  "nnm.source.type"  "<$26>"
       CUSTOM  "nnm.emittingNode.uuid"  "<$22>"
       CUSTOM  "nnm.emittingNode.name"  "<$21>"
       CUSTOM  "OPR_CI_INFO"  "UCMDB:<$28>"
       TEXT "Event (<$5>,<$6>) is closed. <$21>:<$26>=<$25>"
       HELPTEXT "NNMi InterfaceDown incident is closed."
```

Figure 8

Because an `InterfaceDown` incident can be closed without a close reason of `InterfaceUp`, it is a good idea to include an `EventLifecycleStateClosed` condition that matches close conditions other than `InterfaceUp`. The modifications are similar to the `InterfaceUp` condition. The original `EventLifecycleStateClosed` condition can be copied with the following modifications:

1. Test that trap variable #5 is equal to `"InterfaceDown"`.
2. Test that trap variable #29 explicitly excludes `"InterfaceUp"` by using the pattern string `"^<![InterfaceUp]>$"`.
3. Optionally update the `HELPTEXT` attribute with a more descriptive message.

```
# from EVENT EventLifecycleStateClosed .1.3.6.1.4.1.11.2.17.19.2.0.1000
"LOGONLY" Normal
DESCRIPTION "EventLifecycleStateClosed"
CONDITION_ID "e0b6eb99-ba0e-4fa9-89ac-489ad6ba910e"
CONDITION
      $e ".1.3.6.1.4.1.11.2.17.19.2"
      $G 6
      $S 1000
      $14 "4"
      $5 "InterfaceDown"
      $2 "^<@.nnmiprotocol>://<@.nnmiserver>:<@.nnmiport>/nnm$"
      $29 "^<![InterfaceUp]>$"
SET
      NODE     IP 0.0.0.0 "<$21>"
      OBJECT   "<$25>"
      SERVERLOGONLY
      MSGKEY   "<$6>:Close"
      MSGKEYRELATION ACK "^<$6>:<*>$" ICASE
      CUSTOM   "nnm.incident.uuid"  "<$6>"
      CUSTOM   "nnm.server.name"  "<nnmiserver>"
      CUSTOM   "nnm.server.port"  "<nnmiport>"
      CUSTOM   "nnm.name"  "<$5>"
      CUSTOM   "nnm.priority"  "<$13>"
      CUSTOM   "nnm.assignedTo"  "<$19>"
      CUSTOM   "nnm.category"  "<$7>"
      CUSTOM   "nnm.origin"  "<$9>"
      CUSTOM   "nnm.source.name"  "<$25>"
      CUSTOM   "nnm.source.uuid"  "<$27>"
      CUSTOM   "nnm.source.type"  "<$26>"
      CUSTOM   "nnm.emittingNode.uuid"  "<$22>"
      CUSTOM   "nnm.emittingNode.name"  "<$21>"
      CUSTOM   "RelatedCiHint" "<$30>"
      CUSTOM   "EtiHint"  "InterfaceCommunicationStatus:Available"
      TEXT     "Event (<$5>,<$6>) is closed. <$21>:<$26>=<$25>"
      HELPTEXT "NNMi event is closed."
```

Figure 9

NNMi 8.1x Policy Considerations

In NNMi 8.1x, the generated policy requires only one `EventLifecycleStateClosed` condition to handle closing all event types. The conditions presented in the previous two sections still apply, but one more modification is required to update the "catch-all" `EventLifecycleStateClosed` condition to ignore close traps for the `NodeDown` and `InterfaceDown` incidents. Modify the condition as shown in Figure 10.

```
# from EVENT EventLifecycleStateClosed .1.3.6.1.4.1.11.2.17.19.2.0.1000
"LOGONLY" Normal
DESCRIPTION "EventLifecycleStateClosed"
CONDITION_ID "3bb75622-50eb-4778-966b-6c08e1d86cb4"
CONDITION
      $e ".1.3.6.1.4.1.11.2.17.19.2"
      $G 6
      $S 1000
      $14 "4"
      $5 "^<![InterfaceDown|NodeDown]>$"
      $2 "^<@.nnmiprotocol>://<@.nnmiserver>:<@.nnmiport>/nnm$"
SET
      NODE     IP 0.0.0.0 "<$21>"
      OBJECT   "<$25>"
      SERVERLOGONLY
      MSGKEY   "<$6>:Close"
      MSGKEYRELATION ACK "^<$6>:<*>$" ICASE
      CUSTOM   "nnm.incident.uuid"  "<$6>"
      CUSTOM   "nnm.server.name"  "<nnmiserver>"
      CUSTOM   "nnm.server.port"  "<nnmiport>"
      CUSTOM   "nnm.name"  "<$5>"
      CUSTOM   "nnm.priority"  "<$13>"
      CUSTOM   "nnm.assignedTo"  "<$19>"
      CUSTOM   "nnm.category"  "<$7>"
      CUSTOM   "nnm.origin"  "<$9>"
      CUSTOM   "nnm.source.name"  "<$25>"
      CUSTOM   "nnm.source.uuid"  "<$27>"
      CUSTOM   "nnm.source.type"  "<$26>"
      CUSTOM   "nnm.emittingNode.uuid"  "<$22>"
      CUSTOM   "nnm.emittingNode.name"  "<$21>"
      CUSTOM   "OPR_CI_INFO"  "UCMDB:<$28>"
      TEXT     "Event (<$5>,<$6>) is closed. <$21>:<$26>=<$25>"
      HELPTEXT "NNMi event is closed."
```

Figure 10

Appendix A

NNMi NodeDown Trap Definition

Name:	nnmiMgmtEvNodeDown
Type:	NOTIFICATION-TYPE
OID:	1.3.6.1.4.1.11.2.17.19.2.0.32
Full path:	iso(1).org(3).dod(6).internet(1).private(4).enterprises(1).hp(11).nm(2).openView(17).hpNNMi(19).nnmiNbiMIB(2).nnmiNbiNotifications(0).nnmiMgmtEvNodeDown(32)
Module:	HP-NNMI-NBI-MIB
Parent:	nnmiNbiNotifications
Prev sibling:	nnmiMgmtEvNnmClusterTransfer
Next sibling:	nnmiMgmtEvNodeOrConnectionDown
Status:	current
Objects:	1: nnmiApplicationId 2: nnmiNmsUrl 3: nnmiReserved1 4: nnmiReserved2 5: nnmiIncidentName 6: nnmiIncidentUuid 7: nnmiIncidentCategory 8: nnmiIncidentFamily 9: nnmiIncidentOrigin 10: nnmiIncidentNature 11: nnmiIncidentFmtMessage 12: nnmiIncidentSeverity 13: nnmiIncidentPriority 14: nnmiIncidentLifecycleState 15: nnmiIncidentOriginTime 16: nnmiIncidentDbCreateTime 17: nnmiIncidentDbModifiedTime 18: nnmiIncidentDupCount 19: nnmiIncidentAssignedTo 20: nnmiIncidentCias 21: nnmiIncidentSourceNodeHostname 22: nnmiIncidentSourceNodeUuid 23: nnmiIncidentSourceNodeUcmdbId 24: nnmiIncidentSourceNodeMgmtAddr 25: nnmiIncidentSourceName 26: nnmiIncidentSourceType 27: nnmiIncidentSourceUuid 28: nnmiIncidentSourceUcmdbId 29: nnmiIncidentSourceIfName 30: nnmiIncidentSourceIfAlias 31: nnmiIncidentSourceIfDesc 32: nnmiIncidentSourceIfIndex 33: nnmiReserved3 34: nnmiIncidentOtherNodeHostname 35: nnmiIncidentOtherNodeUuid 36: nnmiIncidentOtherNodeUcmdbId 37: nnmiIncidentOtherNodeMgmtAddr 38: nnmiIncidentOtherIfName 39: nnmiIncidentOtherIfAlias 40: nnmiIncidentOtherIfDesc 41: nnmiIncidentOtherIfIndex 42: nnmiReserved4
Description:	This incident indicates that NNMs Advanced Problem Analyzer

has determined the node is down based on the following analysis: 1) 100% of the addresses assigned to this node are unreachable, and 2) The SNMP agent installed on this machine is not responding. At least two of the neighboring devices can be reached and are reporting problems with connectivity to this node.

Figure 11

NNMi Event Closed Trap Definition

Name:	nnmiEvClosed
Type:	NOTIFICATION-TYPE
OID:	1.3.6.1.4.1.11.2.17.19.2.0.1000
Full path:	iso(1).org(3).dod(6).internet(1).private(4).enterprises(1).hp(11).nm(2).openView(17).hpNNMi(19).nnmiNbiMIB(2).nnmiNbiNotifications(0).nnmiEvClosed(1000)
Module:	HP-NNMI-NBI-MIB
Parent:	nnmiNbiNotifications
Prev sibling:	nnmiMgmtEvCardUndeterminedState
Next sibling:	nnmiEvLifecycleStateChanged
Status:	current
Objects:	1: nnmiApplicationId 2: nnmiNmsUrl 3: nnmiReserved1 4: nnmiReserved2 5: nnmiIncidentName 6: nnmiIncidentUuid 7: nnmiIncidentCategory 8: nnmiIncidentFamily 9: nnmiIncidentOrigin 10: nnmiIncidentNature 11: nnmiIncidentFmtMessage 12: nnmiIncidentSeverity 13: nnmiIncidentPriority 14: nnmiIncidentLifecycleState 15: nnmiIncidentOriginTime 16: nnmiIncidentDbCreateTime 17: nnmiIncidentDbModifiedTime 18: nnmiIncidentDupCount 19: nnmiIncidentAssignedTo 20: nnmiIncidentCias 21: nnmiIncidentSourceNodeHostname 22: nnmiIncidentSourceNodeUuid 23: nnmiIncidentSourceNodeUcmdbId 24: nnmiIncidentSourceNodeMgmtAddr 25: nnmiIncidentSourceName 26: nnmiIncidentSourceType 27: nnmiIncidentSourceUuid 28: nnmiIncidentSourceUcmdbId 29: nnmiIncidentClosedReason
Description:	Incident identified by nnmiIncidentUuid was closed in NNMi.

Figure 12

We appreciate your feedback!

If an email client is configured on this system, by default an email window opens when you click here.
If no email client is available, copy the information below to a new message in a web mail client, and then send this message to ovdoc-nsm@hp.com.

Product name and version: NNMi 9.20

Document title: Policies for Integrating HP NNMi with HP Operations Manager

Feedback:

Legal Notices

Warranty

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.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notices

Trademark Notices

Adobe® is a trademark of Adobe Systems Incorporated.

HP-UX Release 10.20 and later and HP-UX Release 11.00 and later (in both 32 and 64-bit configurations) on all HP 9000 computers are Open Group UNIX 95 branded products.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

UNIX® is a registered trademark of The Open Group.

Oracle Technology — Notice of Restricted Rights

Programs delivered subject to the DOD FAR Supplement are 'commercial computer software' and use, duplication, and disclosure of the programs, including documentation, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, programs delivered subject to the Federal Acquisition Regulations are 'restricted computer software' and use, duplication, and disclosure of the programs, including documentation, shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software-Restricted Rights (June 1987). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

For the full Oracle license text, see the license-agreements directory on the NNMi product DVD.

Acknowledgements

This product includes software developed by the Apache Software Foundation.

(http://www.apache.org)

This product includes software developed by the Indiana University Extreme! Lab.

(http://www.extreme.indiana.edu)

Support

Visit the HP Software Support web site at:

www.hp.com/go/hpsoftwaresupport

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp