HP Network Node Manager i Software

Monitoring Devices Located Behind a Static NAT Gateway

Software Version 9.00

You can configure NNMi to monitor devices using static Network Address Translation (Static NAT). This paper describes how to configure NNMi to monitor devices located *behind the NAT gateway* using SNMP polling and SNMP traps.

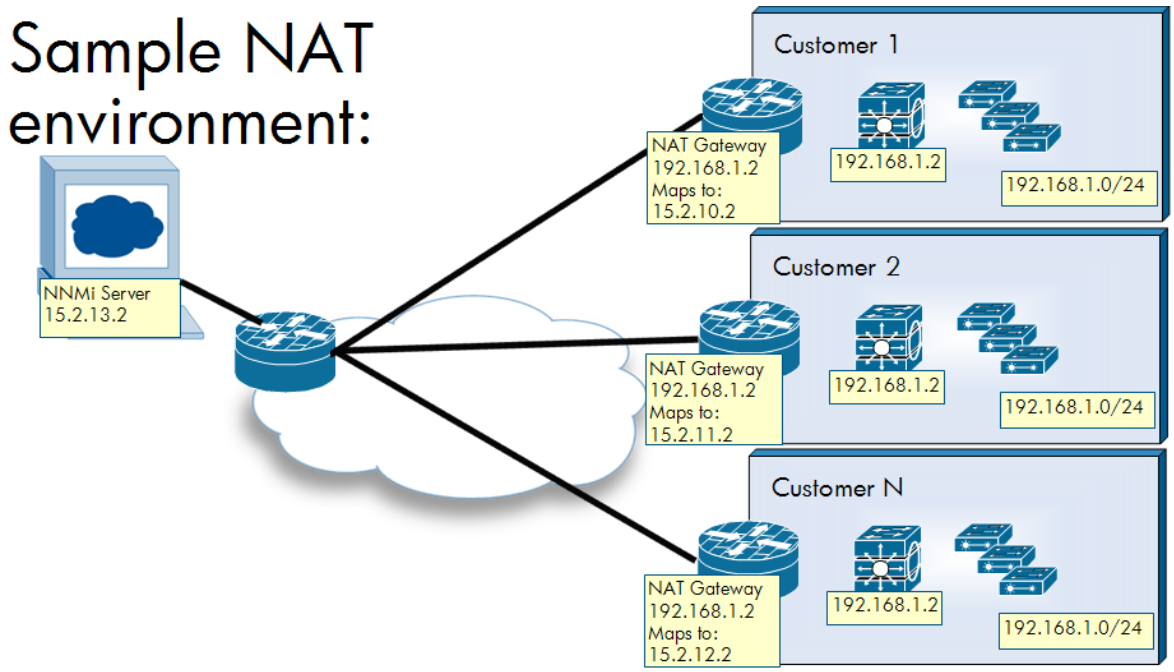
CONTENTS

Problem Statement

When NNMi discovers a node located behind a NAT gateway at a remote site, NNMi uses the public routable address that the NAT gateway assigned to it. However, the node itself is unaware of the address the NAT Gateway assigned to it. Typically, these nodes have non-globally-routable addresses assigned to them for routing within the remote site. A benefit of using NAT is that remote sites can have overlapping IP Addresses because their addresses are unique *within their local domain*. However, this causes challenges for NNMi.

By default, NNMi expects to find a public routable address for a node within a node's IP address table. However, this is not the case when using NAT, as a node assigned the NAT address is unaware that it has a NAT address. Under these conditions, NNMi may disqualify the node from discovery and discard the node.

NNMi can also get confused when receiving traps from nodes behind the NAT gateway as these nodes may have a source address of the non-routable address rather than the NAT assigned global address. NNMi is unable to distinguish which node actually sent the traps.



Solution

You can configure NNMi to honor the management address even if it does not appear in the IP address table of the node. This enables SNMP polling to continue to take place. **You must not configure NNMi to use Auto-discovery for nodes behind the NAT gateway** because of this SNMP configuration prerequisite.

NNMi discovers layer 2 topology for nodes behind the NAT without any additional configuration changes. This is due to protocols such as CDP and LLDP usually not being IP-based but being name-based. Forwarding Database analysis also works without change because it is MAC based rather than IP based.

For trap handling, you can configure managed nodes to provide NNMi with sufficient information to determine the source of the trap. You must do this configuration on the network devices rather than on NNMi. This paper provides general guidance; however, work with your network engineering team for exact details.

Limitations

This solution only supports static NAT. There is an alternative solution: you can assign a public routable address to each managed node. This paper does not cover this solution, as it is usually not feasible.

Presently you can only configure NNMi for SNMP based monitoring on nodes behind the NAT gateway if the nodes support SNMP. NNMi cannot use ICMP polling unless the nodes are marked as having no SNMP support. This limitation is due to NNMi requiring that the ICMP monitored addresses be present in the `ipAddrTable` for an SNMP managed node. If the nodes are non-SNMP, then NNMi can monitor them using ICMP without configuration changes.

There is a new polling feature referred to as `Enable ICMP Management Address Polling`. This feature does not work for nodes behind the NAT firewall.

If you have overlapping IP addresses, you need to filter layer 3 maps for proper viewing.

Summary of Steps

This paper shows a simple example configuration. The basic steps include the following:
Learn the routable address for each managed node from the NAT gateway administrator.
Set up SNMP communication for each node.
Consider disabling the `Small Subnets` connection rules.
Load each node into NNMi using a discovery seed.
Build a node group containing these nodes for better map representation.
Configure traps.

Obtain Routable Addresses

You need to know the routable address for each managed node that uses a NAT address. Obtain this information from your NAT gateway administrator.

Set up SNMP Communication

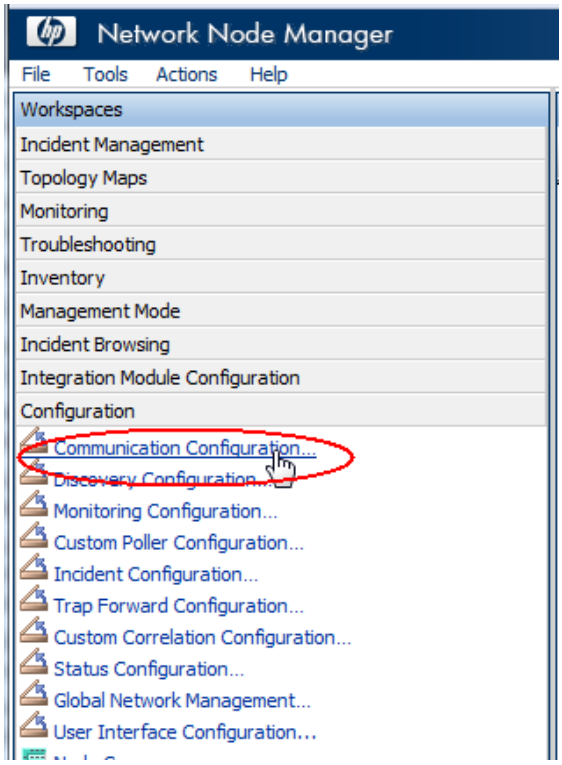
For each node behind the NAT gateway, you must set up SNMP communication to use the routable address, even if it is not in the IP address table. *You must complete this step before NNMi discovers any of these nodes*. To set up SNMP communication, create a `Specific Node Setting` for each node. Do this using either the NNMi console or the command line. You can see both approaches here, starting with the NNMi console approach.

Using the NNMi Console

As mentioned above, for each node you want to manage, you must learn its NAT assigned globally routable address. You can usually obtain this from the NAT gateway administrator.

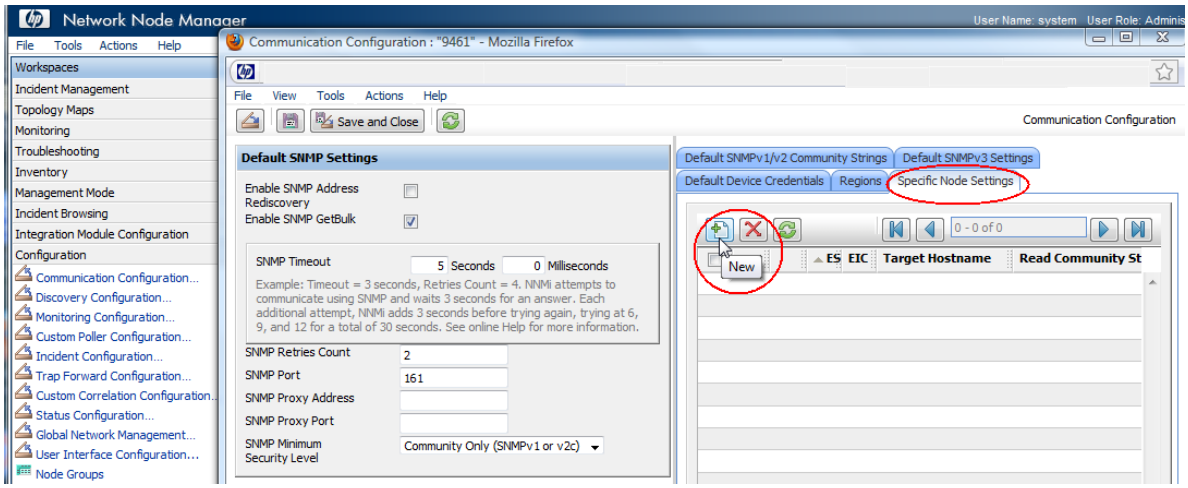
Go to the Configuration workspace and click Communication Configuration.

Figure 1: Communication Configuration



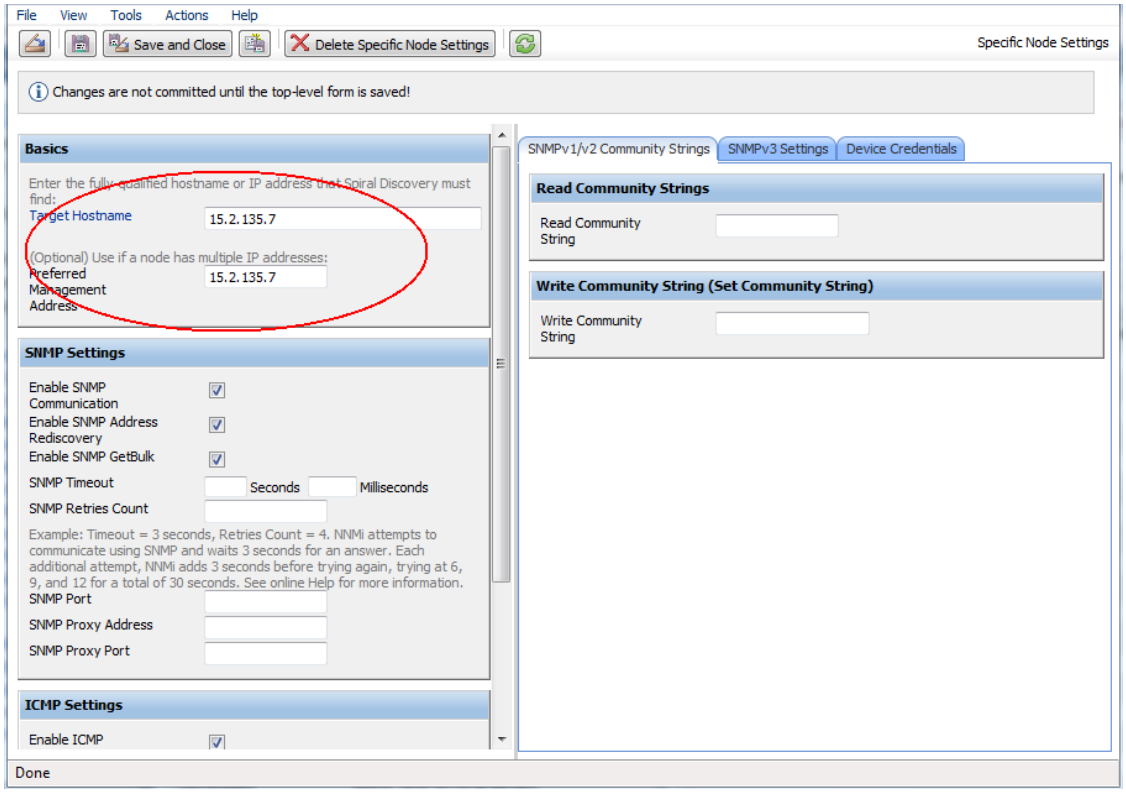
Click the **Specific Node Settings** tab; then click the **New** icon.

Figure 2: Specific Node Settings

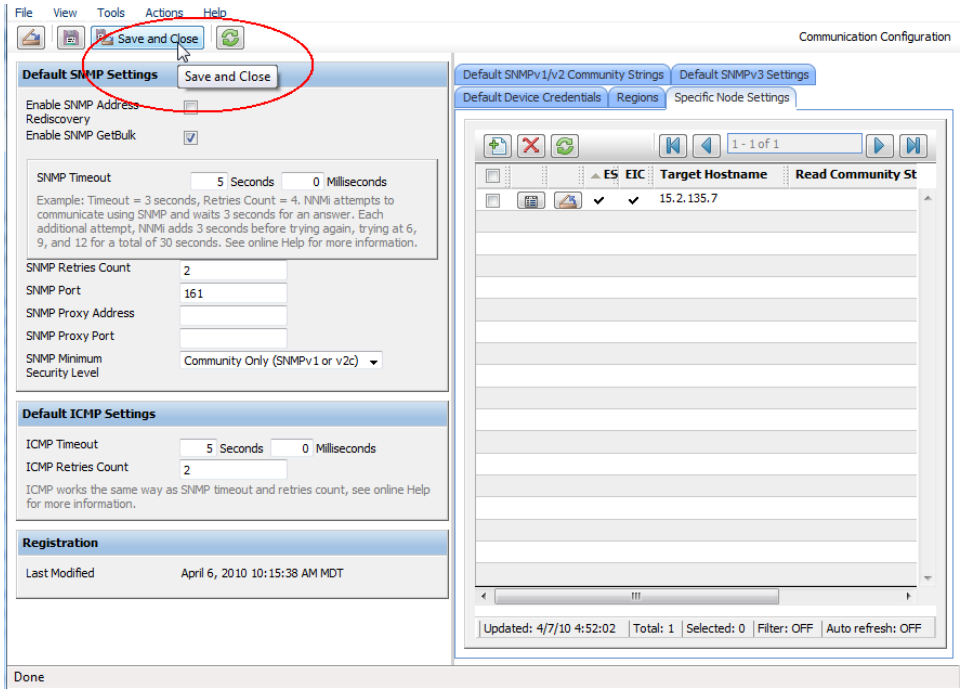


For the `Target Hostname` field, enter the routable address. In the `Preferred Management Address` field, enter the same routable address. You can enter other values here if you like. Usually just leave the other fields blank, using the default settings.

Figure 3: Specific Node Settings Fields



Click **Save and Close** to save this form; then click **Save and Close** to save the outer form.



Using the command line

NNMi provides a script called `nnmcommload.ovpl` that you can use to load many `SNMP Specific Node Settings` in bulk. See the `nnmcommload.ovpl` reference page, or the UNIX manpage for more information. Follow these steps:

Create a file containing all the nodes specified along with their routable addresses. The example below shows a file called nat_snmp.txt:

nat_snmp.txt:
15.2.135.7,,15.2.135.7
15.2.135.10,,15.2.135.10
15.2.135.11,,15.2.135.11
15.2.135.12,,15.2.135.12

After creating the file, run the command as shown here:

```
# nnmcommload.ovpl -u system -p password -file nat_snmp.txt
Processing 15.2.135.7
Processing 15.2.135.10
Processing 15.2.135.11
Processing 15.2.135.12
Resolving parameters and saving configuration entries
Processed 4 lines
```

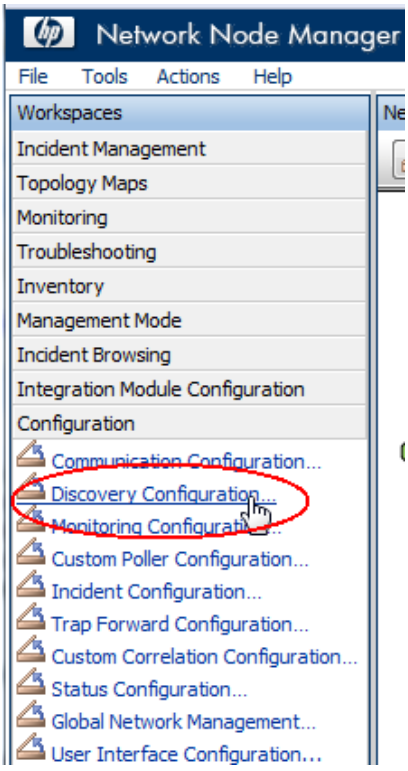
Disabling Small Subnets Connection Rule

Because your network likely contains nodes with duplicate IP addresses in NAT environments (typically on different sites), it is a good idea to disable the `Small Subnets` discovery rule. This rule allows NNMi to build connections based on IP addresses with /30 subnet masks. Disabling this feature may not be necessary in your environment, so see to the NNMi help for further details about this feature. However, if you anticipate that nodes behind the NAT gateway will have some duplicate /30 subnet masks, you should disable this feature. You should consider disabling other discovery rules as required by your environment.

To disable the Small Subnets connection rule, do the following:

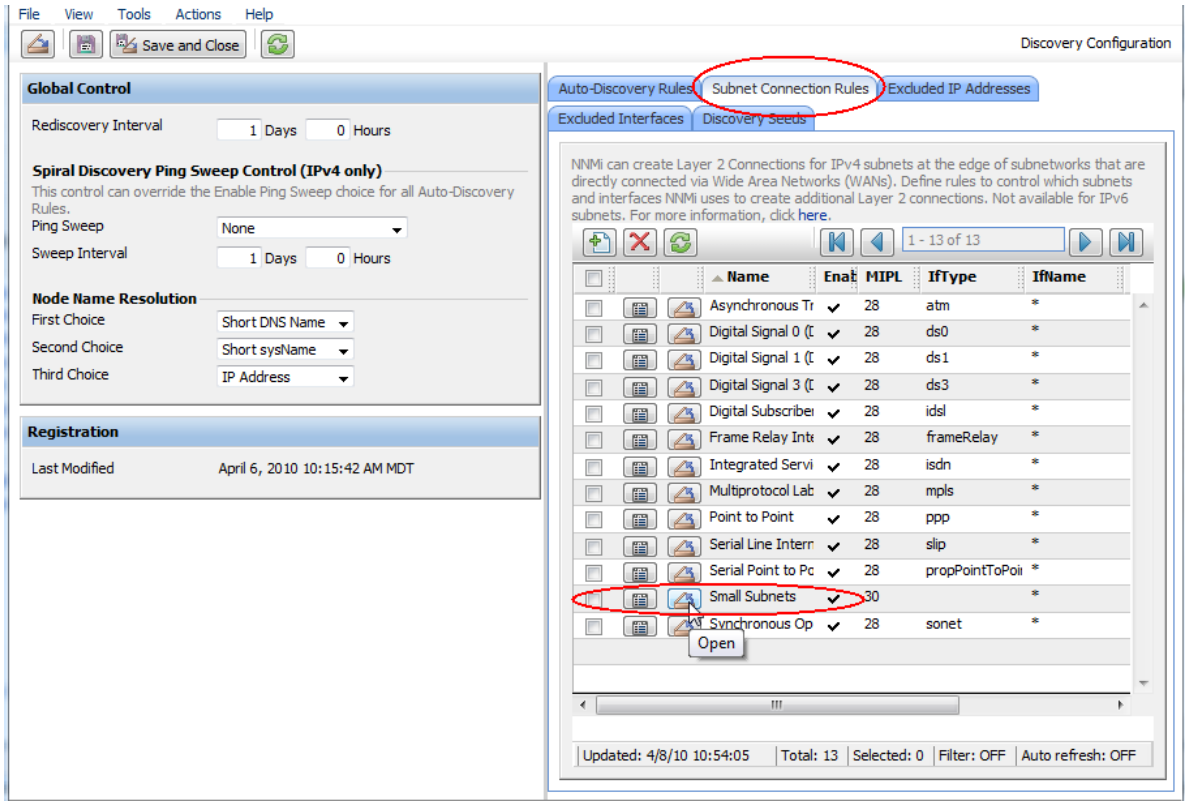
From to the Configuration workspace, click Discovery Configuration.

Figure 4: Discovery Configuration



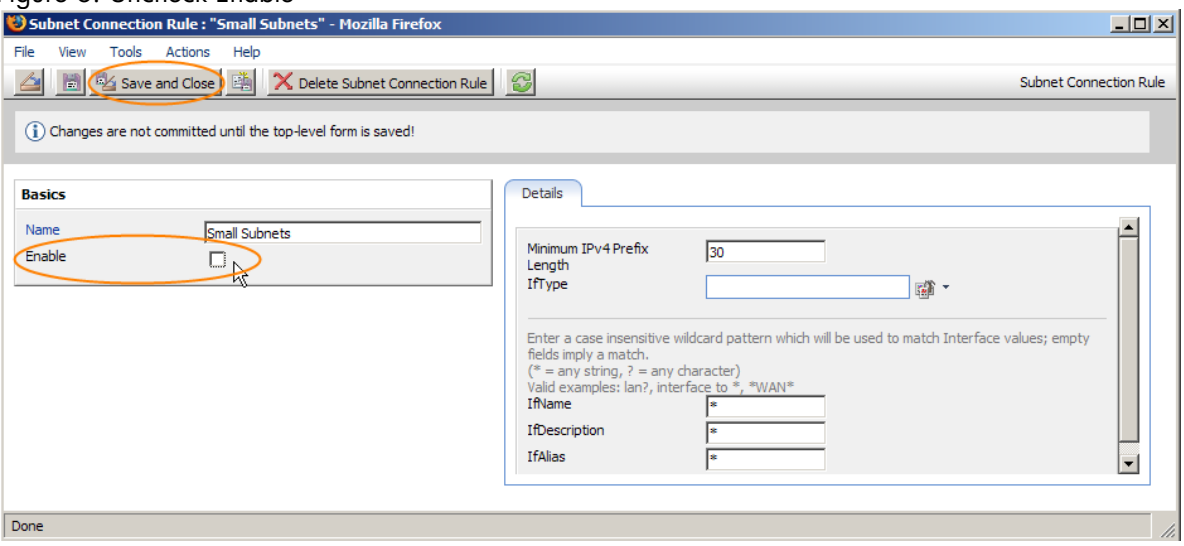
Click the **Subnet Connection Rules** tab; then open the **Small Subnets** rule.

Figure 5: Open the Small Subnets Rule



Uncheck the **Enable** box. Click **Save and Close** to save this form; then click **Save and Close** to save the outer form.

Figure 6: Uncheck Enable



Loading seeds for discovery

Now that you have set up SNMP configuration for the nodes behind the NAT gateway, you can configure discovery. You must *seed* all nodes behind the NAT gateway to guarantee NNMi performs as expected.

You can load the discovery seeds via the GUI (one at a time) or by using a command line. This example only shows the command line.

Create a file with a line for each node containing the routable address. For Example:

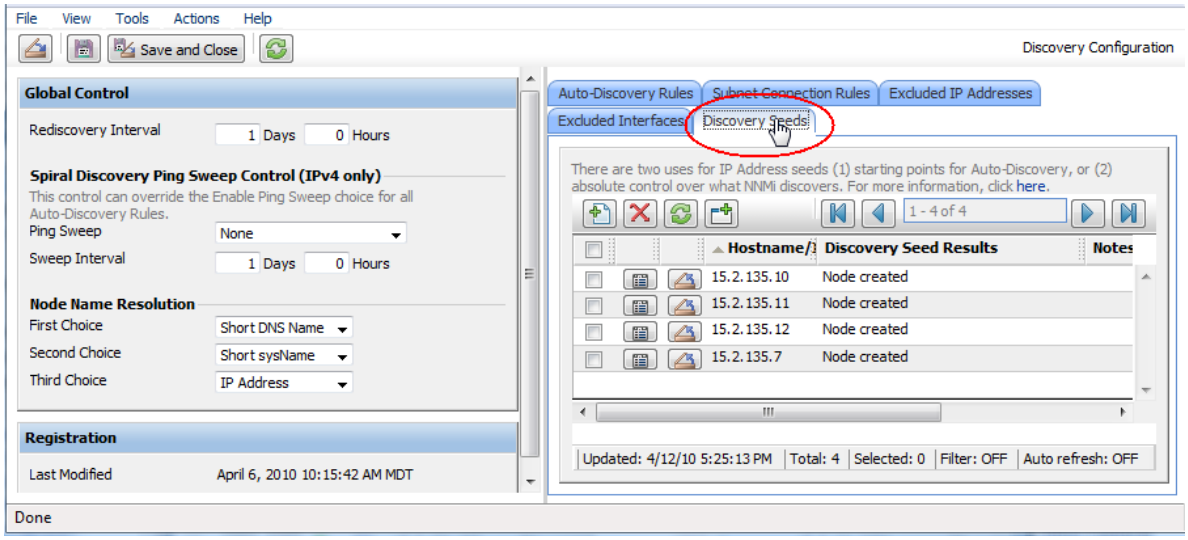
nat_seeds.txt:
15.2.135.7
15.2.135.10
15.2.135.11
15.2.135.12

Use the `nnmloadseeds.ovpl` command line tool to load these seeds into NNMi.

```
# nnmloadseeds.ovpl -f ./nat_seeds.txt
4 seeds added
0 seeds invalid
0 seeds duplicated
```

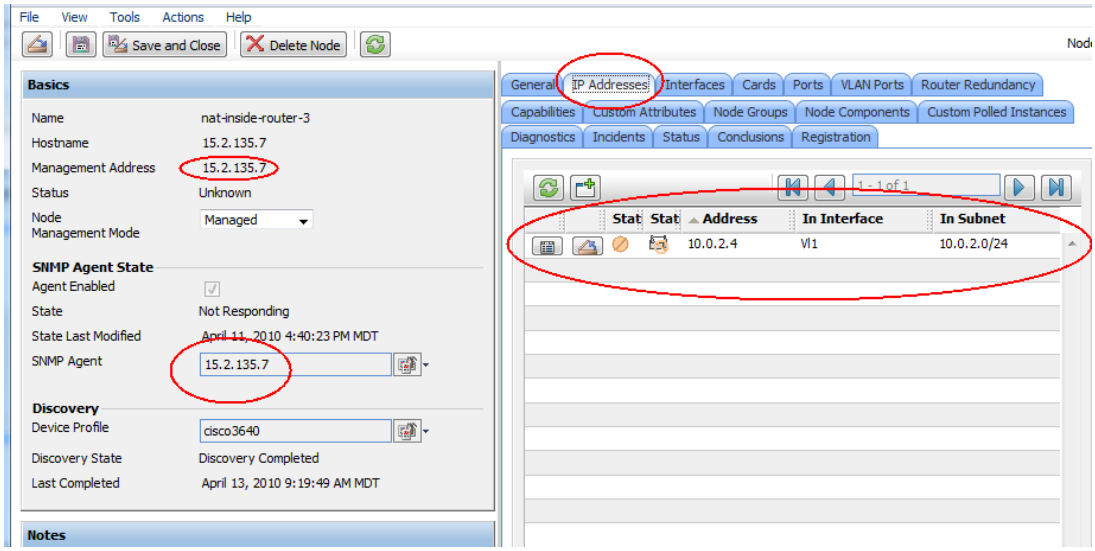
Track the discovery of the nodes using the NNMi console. Go to the **Configuration** workspace; then click **Discovery Configuration**.
Click the **Discovery Seeds** tab and see the nodes you just seeded. This table does not automatically refresh, so click **Refresh** periodically.

Figure 7: Newly Seeded Nodes



After some time, you see the nodes discovered in NNMi. In this example, you can open one of these nodes to see that it has a routable management address that is not in the `IP Addresses` table.

Figure 8: Has a Routable Management Address



Topology can take longer to discover, so allow plenty of time for NNMi to discover all of the connectivity accurately; this normally takes a few hours. In this example, you can see in Figure 9and Figure 10 that NNMi accurately discovered the connectivity of the four nodes. NNMi discovered some connections using CDP and others using FDB (forwarding database).

Figure 9: Properly Connected Nodes (CDP)

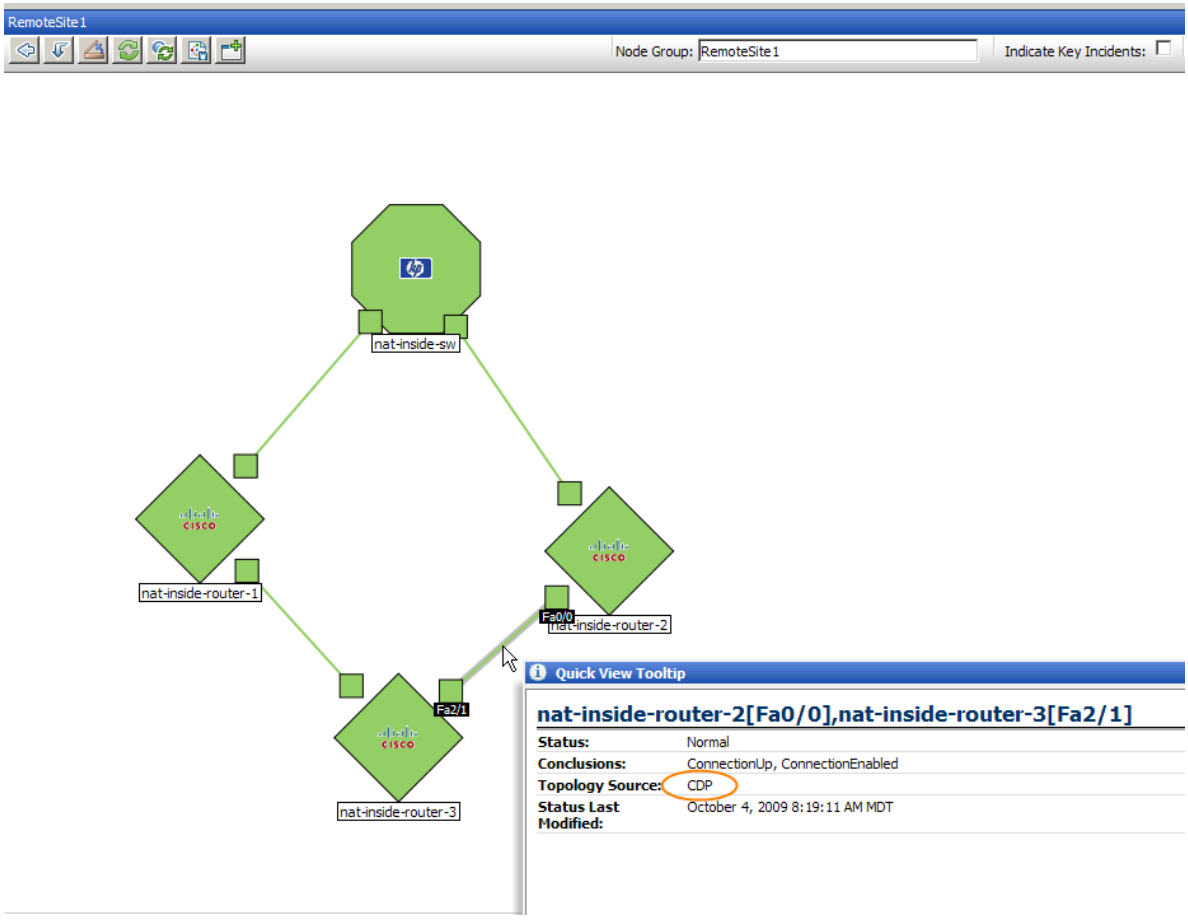
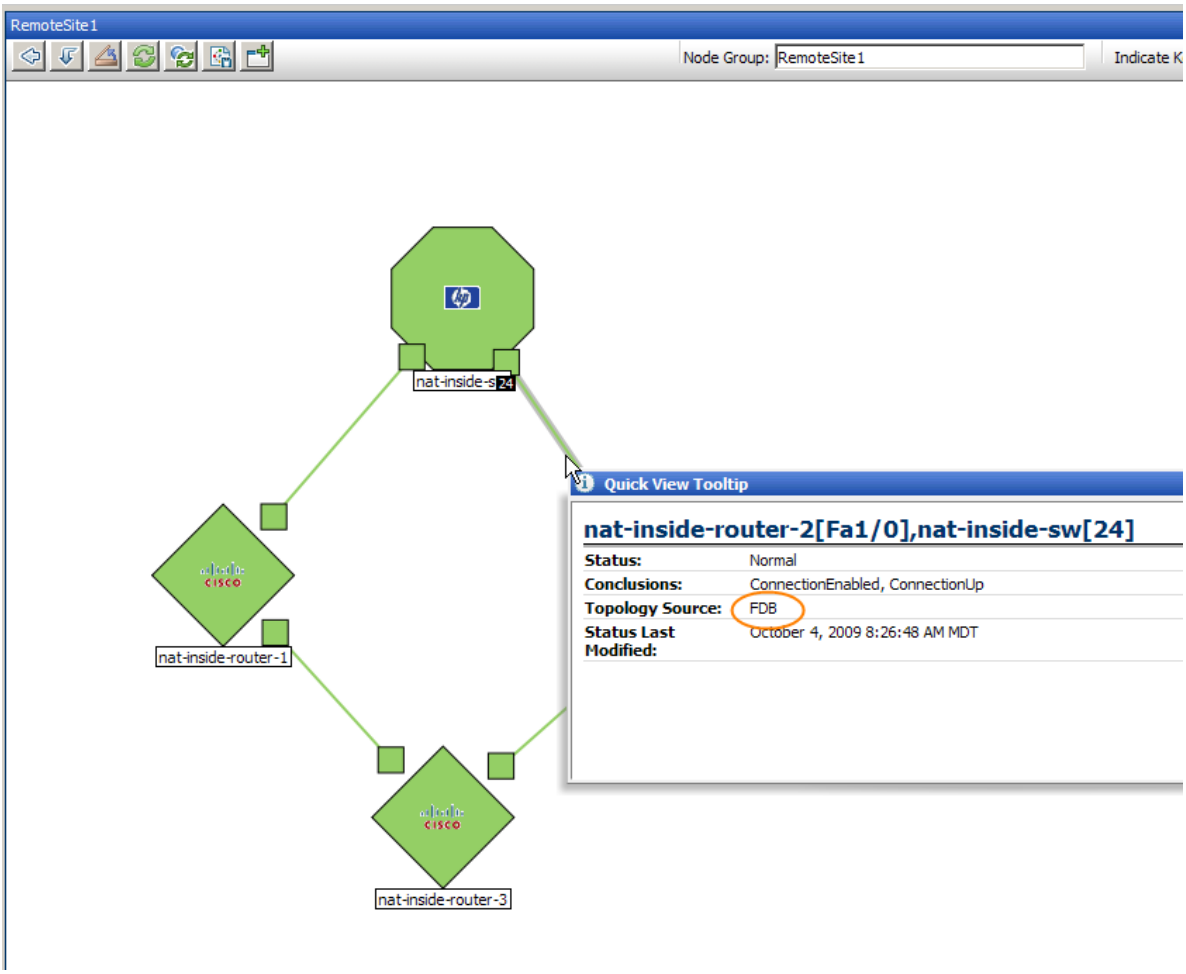


Figure 10: Properly Connected Nodes (FDB)

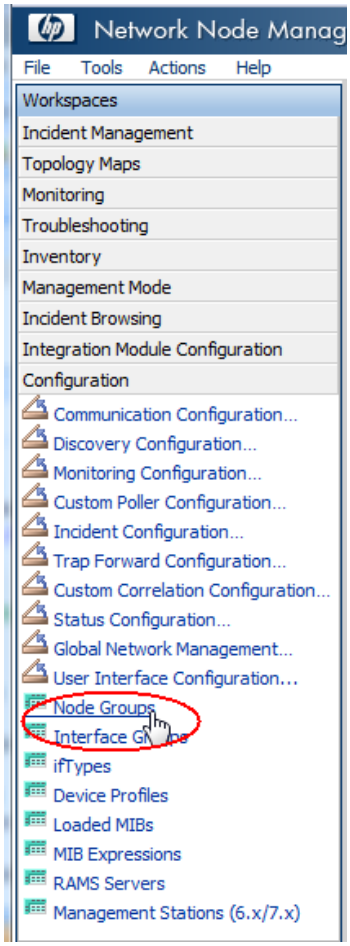


Create a Node Group and Node Group Maps for the Nodes on this Site.

The next step is to create a node group for the nodes at this site so that NNMi can better filter these views. If you want to have both a Layer 2 and a Layer 3 Node Group map, you will need to create two node groups (one for each map). You can use the same filtering for each Node Group.
To create a node group, do the following:

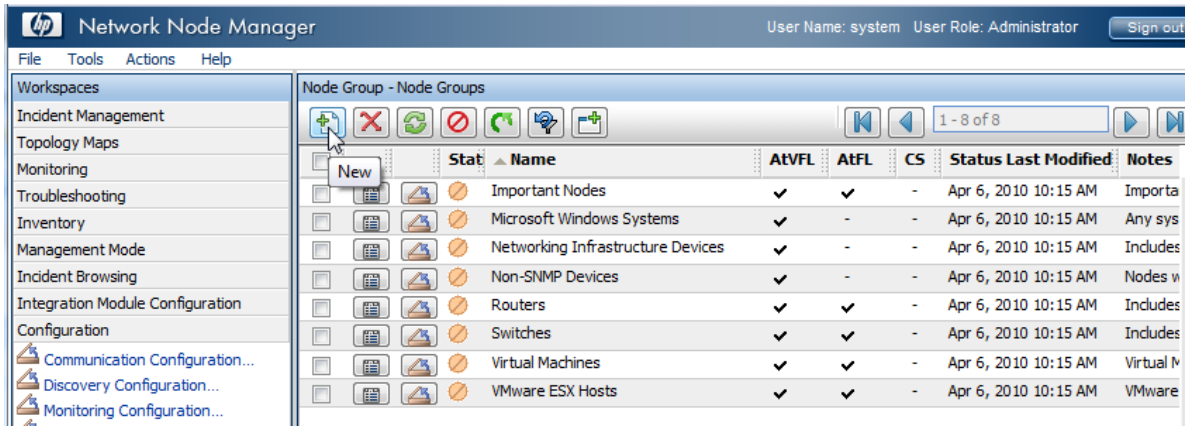
Go to the `Configuration` workspace; then select **Node Groups**.

Figure 11: Select Node Groups



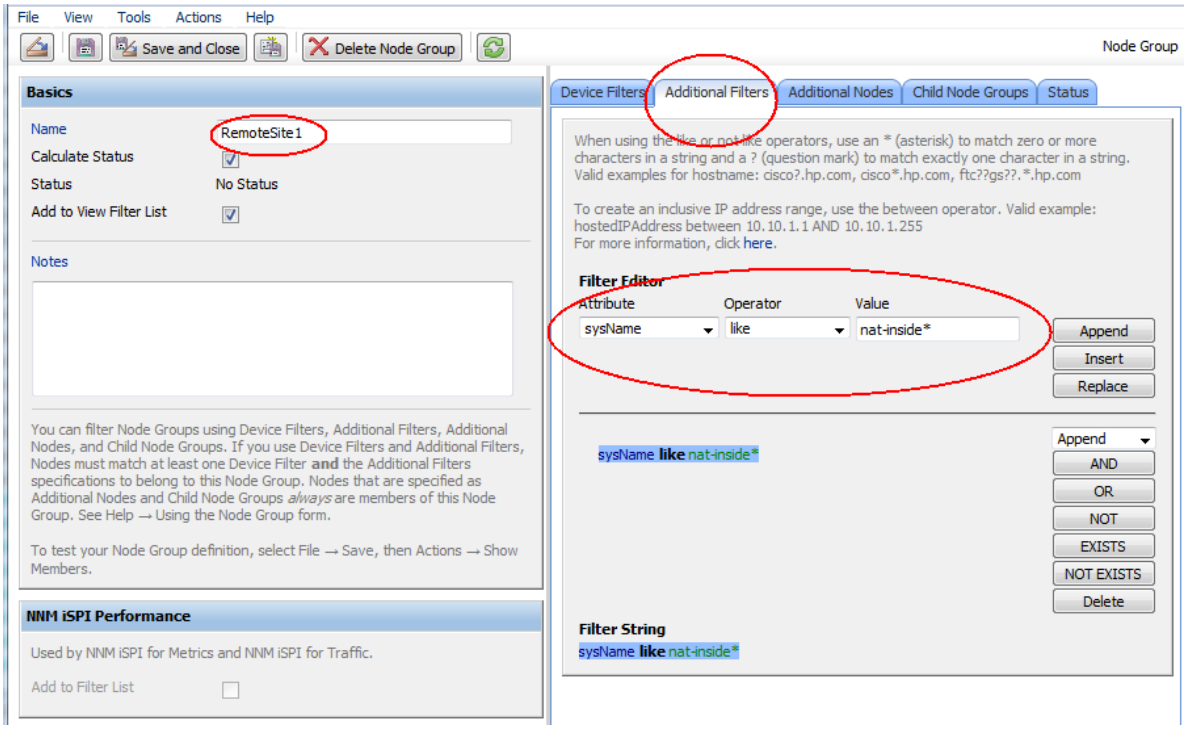
Click the **New** icon.

Figure 12: Click the New Icon to Create a New Node Group



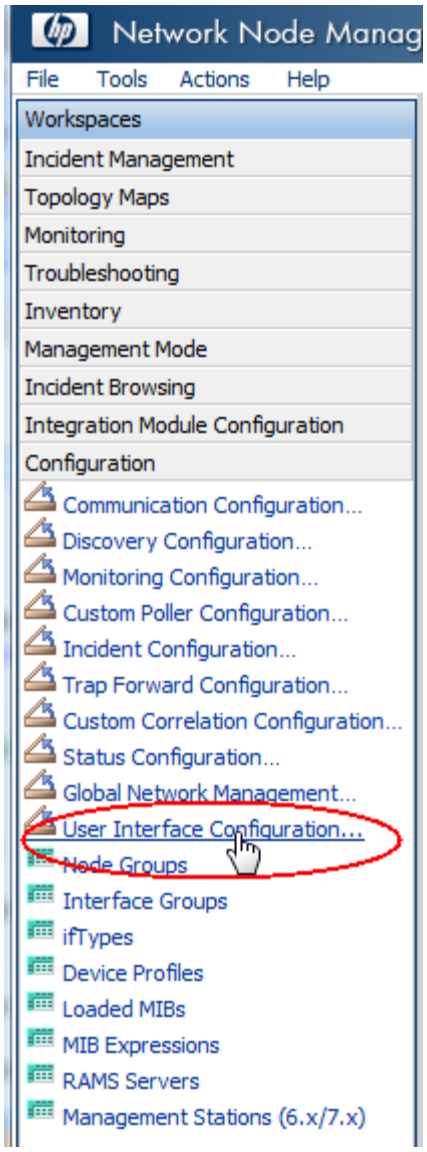
Create a filter that identifies all the nodes at this particular site.

Figure 13: Create a Filter

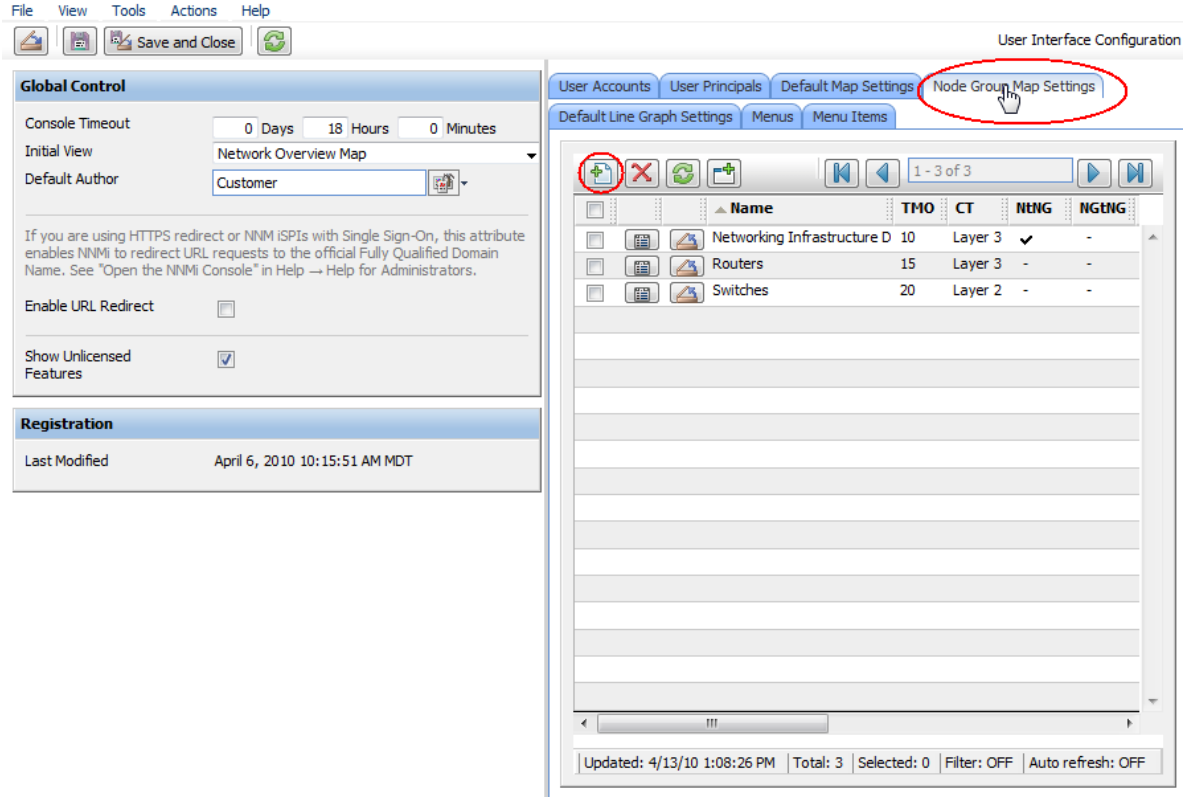

File View Tools Actions Help
Save and Close
Delete Node Group
Node Group
Basics
Name
RemoteSite1
Calculate Status
Status
No Status
Add to View Filter List
Notes
You can filter Node Groups using Device Filters, Additional Filters, Additional Nodes, and Child Node Groups. If you use Device Filters and Additional Filters, Nodes must match at least one Device Filter and the Additional Filters specifications to belong to this Node Group. Nodes that are specified as Additional Nodes and Child Node Groups always are members of this Node Group. See Help → Using the Node Group form.
To test your Node Group definition, select File → Save, then Actions → Show Members.
NNM iSPI Performance
Used by NNM iSPI for Metrics and NNM iSPI for Traffic.
Add to Filter List
Device Filters
Additional Filters
Additional Nodes
Child Node Groups
Status
When using the like or not like operators, use an * (asterisk) to match zero or more characters in a string and a ? (question mark) to match exactly one character in a string.
Valid examples for hostname: cisco?.hp.com, cisco*.hp.com, ftc??gs??.*.hp.com
To create an inclusive IP address range, use the between operator. Valid example:
hostedIPAddress between 10.10.1.1 AND 10.10.1.255
For more information, click here.
Filter Editor
Attribute
Operator
Value
sysName
like
nat-inside*
Append
Insert
Replace
sysName like nat-inside*
Append
AND
OR
NOT
EXISTS
NOT EXISTS
Delete
Filter String
sysName like nat-inside*

To create a node group map for these nodes, click **User Interface Configuration**.

Figure 14: User Interface Configuration

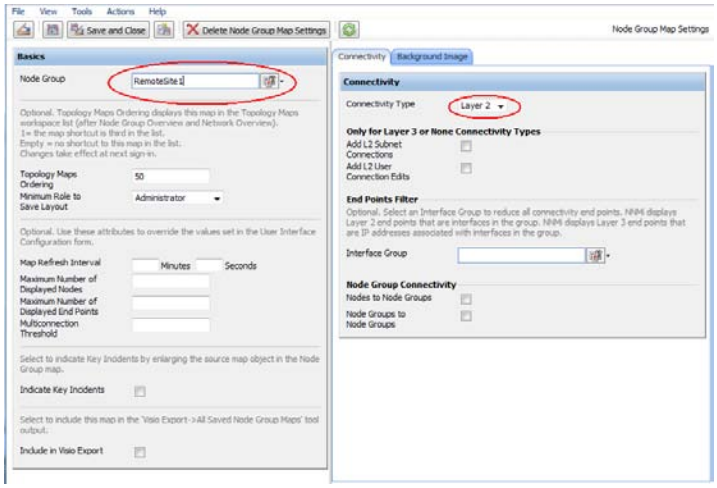


Click the **Node Group Map Settings** tab; then click **New**.



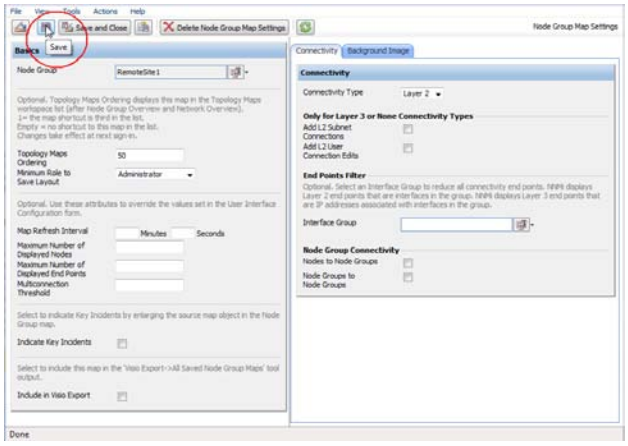
Select the `Node Group`; then choose the `Connectivity Type`.

Figure 15: Select the Node Group and Connectivity Type



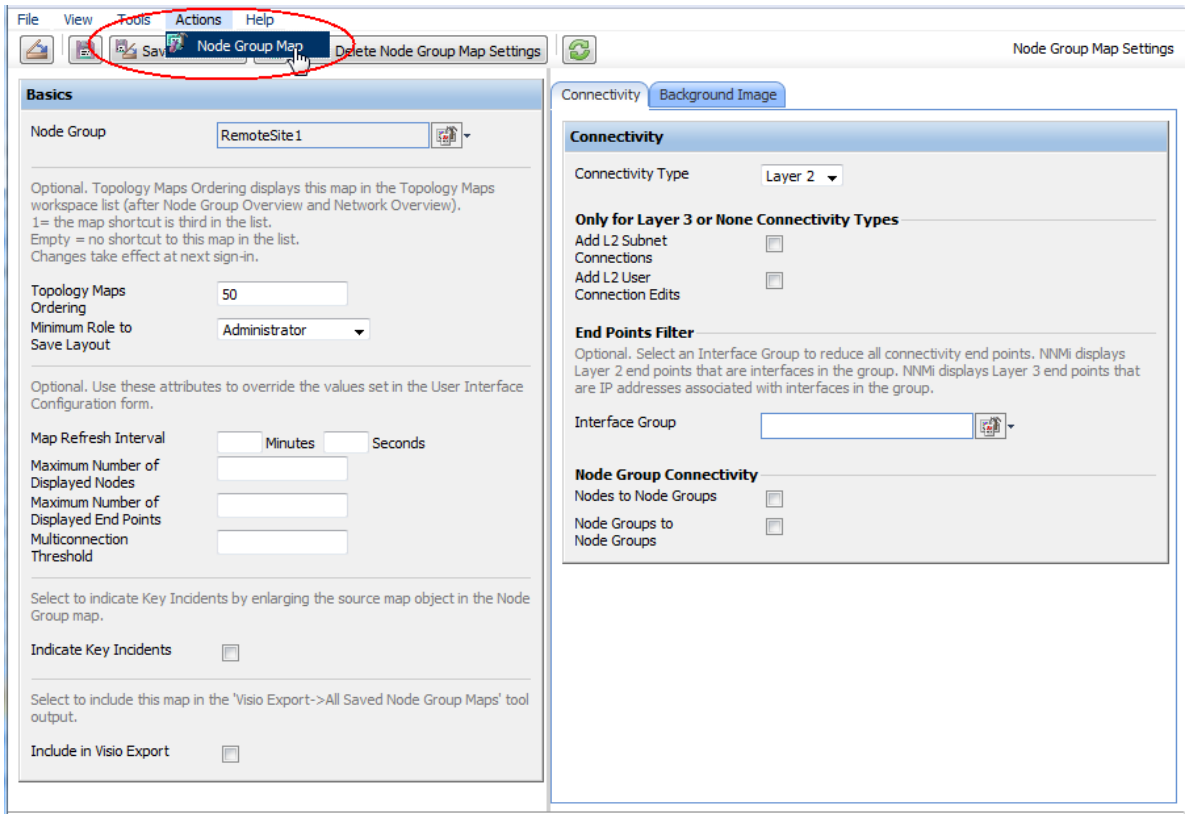
Click **Save**.

Figure 16: Save the Node Group Map Settings



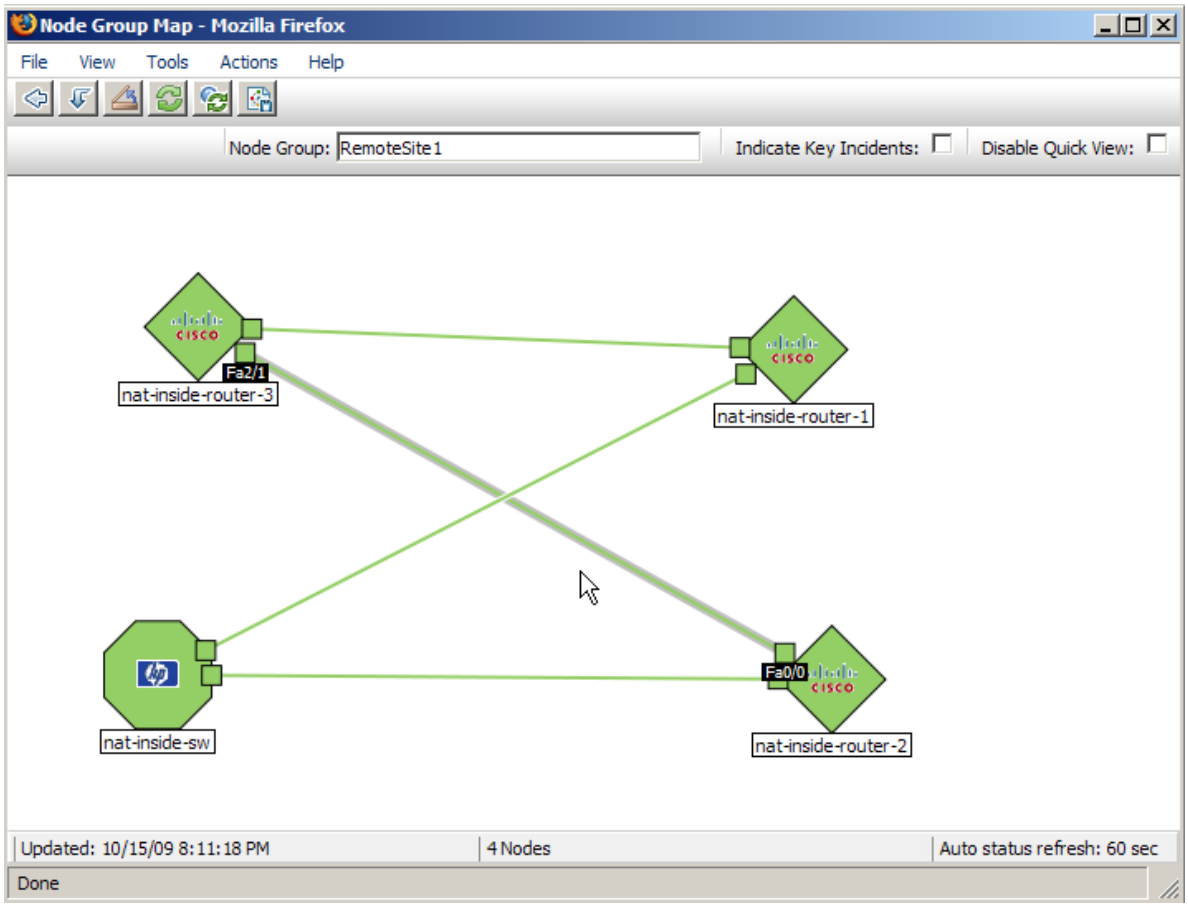
Open the **Node Group Map** from the **Actions** menu.

Figure 17: Open the Node Group Map



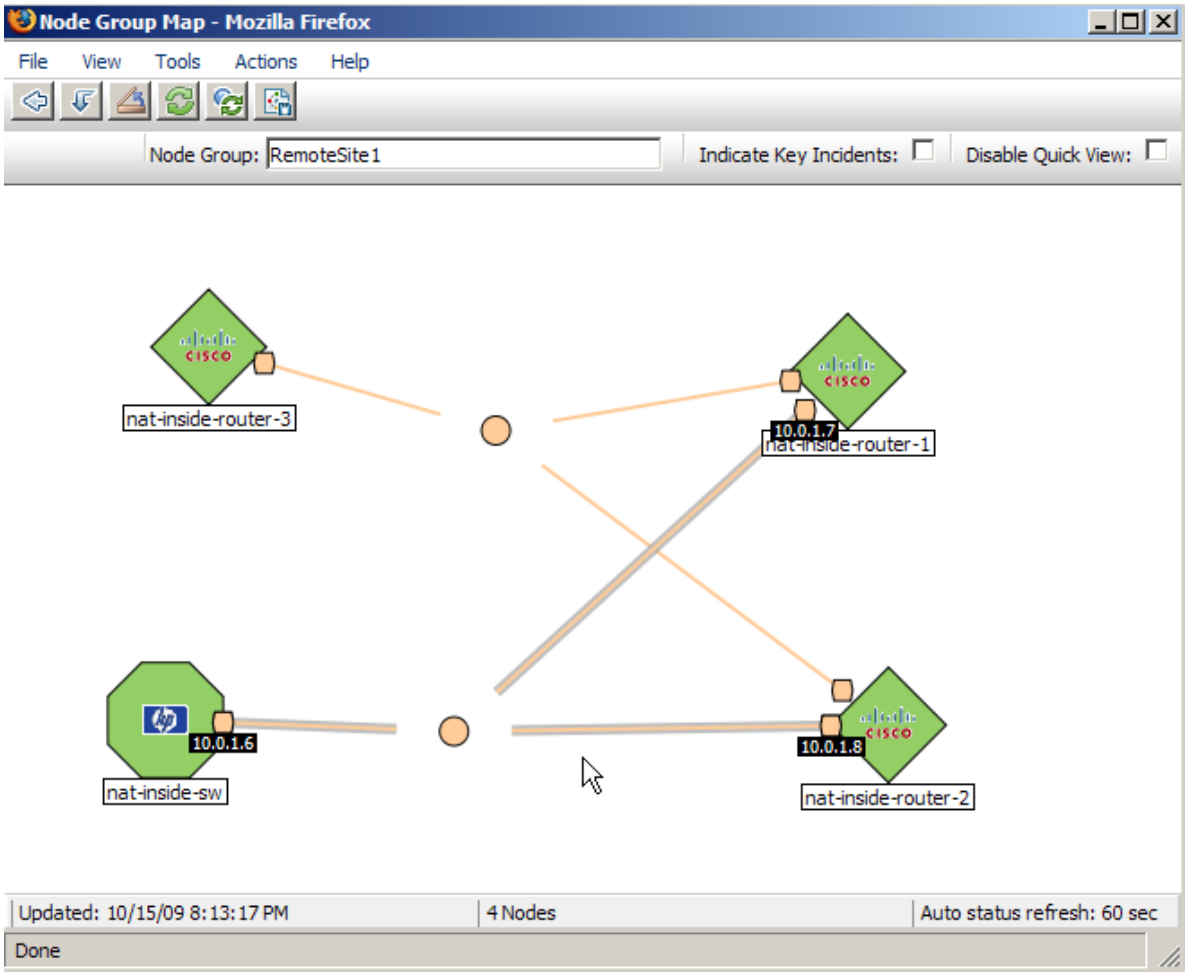
You can see the Layer 2 Node Group Map.

Figure 18: Observe the Layer 2 Node Group Map



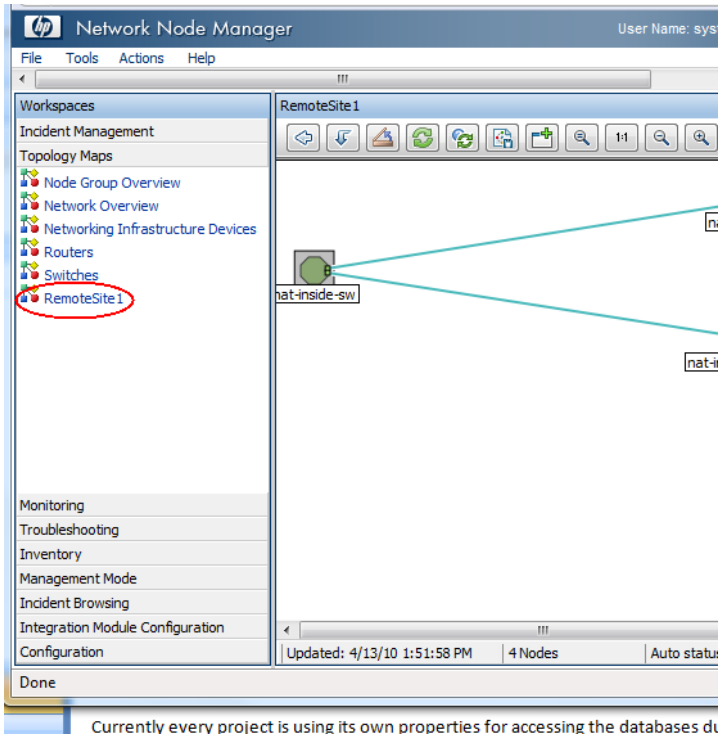
A layer 3 map, shown in figure 17, is similar. Notice that NNMi bases these connections on IP Addresses.

Figure 19: Layer 3 Map



You may need to log out, and then log back in again to see the `Node Group Map` in the `Topology Maps` workspace.

Figure 20: Node Group Map Shown in the Topology Maps Workspace



Neighbor View tips

When working with nodes behind NAT gateways, be cautious with some of the maps. When selecting a node, a layer 2 neighbor view works well, however a layer 3 neighbor view does not give accurate results. This is due to multiple nodes sharing the overlapping IP addresses. NNMi shows these overlapping IP addresses *connected together* in the layer 3 neighbor view, however they are not connected when they sit behind different NAT gateways.

This will not affect any monitoring or fault analysis because NNMi does not base that analysis on a layer 3 neighbor relationship. You will still have good monitoring and analysis.

SNMP Traps

In order for the NNMi management server to receive SNMP traps from nodes behind the NAT gateway, you must make changes to the managed nodes. This example covers two types of SNMP traps: SNMPv2c traps and SNMPv1 traps. This example also shows changes specific to Cisco devices. Other vendors may require similar changes. This example does not provide specific IOS commands.

Challenge with traps

The challenge with traps usually comes down to source address resolution. NNMi must unambiguously resolve the source address of traps that it receives. This problem manifests itself differently depending on the SNMP version (v1 or v2c).

SNMPv2c traps

Dealing with SNMPv2c traps is relatively easy. Table 1 shows the format of an SNMPv2c trap, with the IP Header forming the top section of the table and the SNMP Trap PDU forming the lower section of the table.

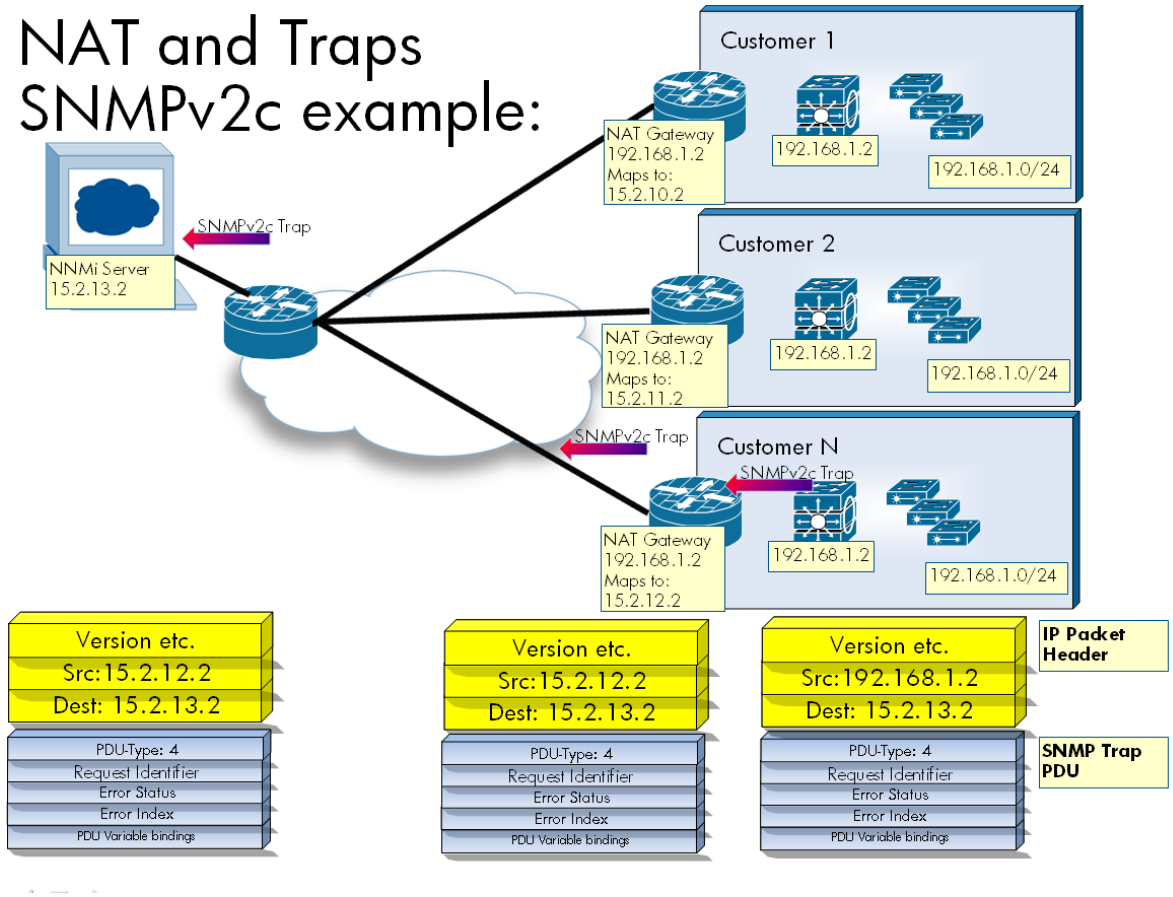
Version etc.
Source Address
Destination Address
PDU-Type: 4
Request Identifier
Error Status
Error Index
PDU Variable bindings

Table 1 SNMPv2c Trap Format

Since SNMPv2c traps do not have an `Agent Address` field in the PDU (protocol data unit), the only source field of the trap is within the IP Packet Header. NAT routers properly translate the source field. Only one step is required on the source node: make sure the interface associated with the private inside IP address sources all traps from devices behind the NAT router.
This IP address must be on a device that is statistically mapped to a public address on the NAT gateway. This allows the NAT gateway to translate the trap to the correct public address.

Figure 19 shows an example of this correct translation from the NAT gateway. You can see that the NAT gateway properly translates a trap that begins with the source address of 192.168.1.2 to address 15.2.13.2. Then the NNMi server correctly resolves this address.

Figure 21: NAT Gateway Correct Translation



SNMPv1 traps

SNMPv1 traps are more complex because they embed the `Agent Address` inside the SNMP Trap PDU. Table 2 shows he format of an SNMPv1 trap, with the IP Header forming the top section of the table and the SNMP Trap PDU forming the lower section of the table. You can see the `Agent Address` highlighted in the PDU.

Version etc.
Source Address
Destination Address
PDU-Type: 4
Enterprise
Agent Address
Generic Trap Code
Specific Trap Code
Timestamp
PDU Variable Bindings

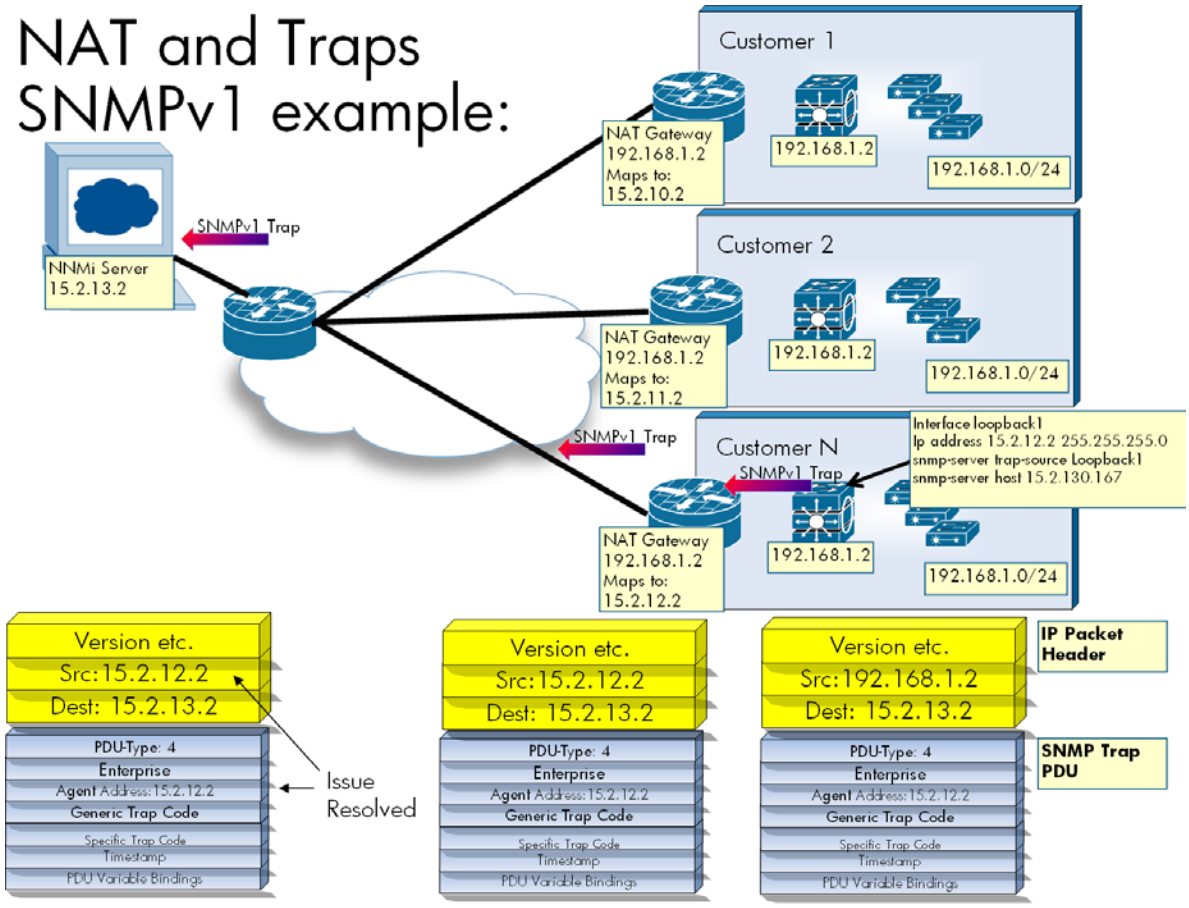
Table 2: SNMPv1 Trap Format

Because the `Agent Address` is embedded in the PDU, the NAT router will not translate this value since it is part of the payload rather than the header.

A Cisco-specific solution for this problem is to create an addition loopback address on the device behind the NAT gateway that corresponds to the routable public address. Then make this the source address for traps sent from the device. By making this change, NNMi can correctly resolve SNMPv1 traps.

See the example in Figure 20. Without adding another loopback address to the node, the NAT gateway translates the source address in the IP header to 15.2.12.2, based on the mapping in the static NAT pool. However, the PDU is not translated, so the `agent-addr` 192.168.1.2 remains. Then NNMi receives the SNMPv1 trap and uses the `agent-addr` to resolve the trap to the source object. This source object is ambiguous since it may be the 192.168.1.2 in customer 1, customer 2, or customer N.

Figure 22: SNMPv1 Example



By adding another loopback address to the node that contains the IP address of the static mapping for private management address on this device, this scenario can now succeed. In this example, the address is 15.2.12.2. The SNMP server is configured to send traps from this address. When 192.168.1.2 generates an SNMPv1 trap, the IP header source is 192.168.1.2. Since the source of this trap has been set to the loopback1, the trap takes the `agent-addr` from that interface's IP address. The trap gets an `agent-addr` of 15.2.12.2. The NAT gateway translates the source in the IP header to 15.2.12.2 based on the mapping in the static NAT pool. The PDU is not translated, so the `agent-addr` of 15.2.12.2 remains intact.

NNMi receives the SNMPv1 trap and uses the `agent-addr` to resolve the trap to the source object. Since the 15.2.12.2 maps uniquely to the device the trap came from, the trap is resolved to the correct source objects. If traps are generated by the other 192.168.1.2 devices, and they are configured to source traps from their static NAT address, the traps will be resolved to the correct object.

Conclusion

By following the steps presented in this paper to configure NNMi to monitor devices located behind the NAT gateway and to configure managed nodes to provide NNMi with sufficient information to determine trap sources, you can more effectively monitor networks that contain devices using static NAT.

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

Acrobat® 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.

Java™ is a US trademark of Sun Microsystems, Inc.

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

Oracle is a registered trademark of Oracle Corporation 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 USA, 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)

This product includes software developed by The Legion Of The Bouncy Castle. (http://www.bouncycastle.org)

This product contains software developed by Trantor Standard Systems Inc. (http://www.trantor.ca)