OPTIMIZE

MERCURY BUSINESS AVAILABILITY CENTER™

Discovery Manager Tutorial

MERCURY™

BUSINESS TECHNOLOGY OPTIMIZATION

Mercury Business Availability Center

Discovery Manager Tutorial

Version 6.1

Document Release Date: March 5, 2006

MERCURY™

Mercury Business Availability Center, Version 6.1
Discovery Manager Tutorial

This manual, and the accompanying software and other documentation, is protected by U.S. and international copyright laws, and may be used only in accordance with the accompanying license agreement. Features of the software, and of other products and services of Mercury Interactive Corporation, may be covered by one or more of the following patents: United States: 5,511,185; 5,657,438; 5,701,139; 5,870,559; 5,958,008; 5,974,572; 6,137,782; 6,138,157; 6,144,962; 6,205,122; 6,237,006; 6,341,310; 6,360,332, 6,449,739; 6,470,383; 6,477,483; 6,549,944; 6,560,564; 6,564,342; 6,587,969; 6,631,408; 6,631,411; 6,633,912; 6,694,288; 6,738,813; 6,738,933; 6,754,701; 6,792,460 and 6,810,494. Australia: 763468 and 762554. Other patents pending. All rights reserved.

Mercury, Mercury Interactive, the Mercury logo, the Mercury Interactive logo, LoadRunner, WinRunner, SiteScope and TestDirector are trademarks of Mercury Interactive Corporation and may be registered in certain jurisdictions. The absence of a trademark from this list does not constitute a waiver of Mercury's intellectual property rights concerning that trademark.

All other company, brand and product names may be trademarks or registered trademarks of their respective holders. Mercury disclaims any responsibility for specifying which marks are owned by which companies or which organizations.

Mercury provides links to external third-party Web sites to help you find supplemental information. Site content and availability may change without notice. Mercury makes no representations or warranties whatsoever as to site content or availability.

Mercury Interactive Corporation
379 North Whisman Road
Mountain View, CA 94043
Tel: (650) 603-5200
Toll Free: (800) TEST-911
Customer Support: (877) TEST-HLP
Fax: (650) 603-5300

© 2005-2006 Mercury Interactive Corporation, All rights reserved

If you have any comments or suggestions regarding this document, please send them by e-mail to documentation@mercury.com.

AMLIB_DMTUT6.1/01

Table of Contents

Welcome to the Discovery Manager Tutorial ..... **v**
Before You Begin v
Using This Tutorial vi
Who Should Read This Guide vii
Getting More Information viii

Lesson 1: Introducing the Discovery Process 1
What is a Discovery Pattern? 1
What is the Configuration Item Type Model? 2
Running the Discovery Process 3
Launching Mercury Business Availability Center 5

Lesson 2: Defining the Seed Network 7
Inserting a CI Manually 8
Defining the Discovery Scope 11

Lesson 3: Discovering Network CIs 17
Activating the ICMP_NET_Dis_IpC Discovery Pattern 18
What Happens When You Activate the ICMP_NET_Dis_IpC Pattern? 19

Lesson 4: Creating a TQL Query 21
Defining a TQL Query to View the Discovered Network CIs 21

Lesson 5: Performing an Advanced Network Discovery 27
Defining the SNMP Connection Data 28
Activating the SNMP_NET_Dis_Connection Discovery Pattern 32
Defining a TQL to View the Discovered CIs 35

Lesson 6: Expanding the Network Discovery 39
Activating Patterns that Expand the Network Discovery 40
Viewing the Discovered CIs 43

Lesson 7: Discovering Database Instances and Oracle Resources.....45
Adding the SQL Protocol.....46
Activating the SQL_NET_Dis_Connection Pattern.....49
Activating the SQL_APP_Dis_Oracle Discovery Pattern.....51

Lesson 8: Discovering WebLogic Instances and Components.....55
Defining the WebLogic Protocol.....56
Discovering WebLogic Instances.....59
Discovering WebLogic Components.....61

Lesson 9: Discovering Host Resources.....63
Defining the WMI Protocol.....64
Discovering WMI Components.....68

Lesson 10: Troubleshooting Pattern Activation Issues.....71

Appendix A: Optional Variables in Discovery Patterns.....73

Appendix B: Discovery Patterns.....75
Security Privileges for Patterns.....143
Specific Pattern Data Source.....144

Welcome to the Discovery Manager Tutorial

Welcome to the Mercury Business Availability Center Discovery Process Tutorial, a self-paced guide that teaches you how to run the discovery process in Mercury Business Availability Center.

This tutorial instructs you on how to discover the IT resources in your system. It takes you through a gradual discovery process, from the most basic network discovery to more in-depth discoveries such as applications, databases and servers.

Before You Begin

To do this tutorial, you must have Mercury Business Availability Center operational. The Discovery Probe must be installed and running. You must also have access to the Discovery Manager user interface to activate the discovery patterns.

Using This Tutorial

This tutorial contains the following lessons:

Lesson 1 Introducing the Discovery Process

Introduces you to the Mercury Application Mapping discovery process, discovery patterns and the Configuration Item Type Model.

Lesson 2 Defining the Seed Network

Shows you how to define the seed network from which to start the discovery process.

Lesson 3 Discovering Network CIs

Shows you how to activate the discovery pattern **ICMP_NET_Dis_IpC**, which is designed to discover the networks that fall within the defined IP address range.

Lesson 4 Creating a TQL Query

Shows you how to define a TQL query that retrieves specified network CIs from the Mercury Universal CMDB.

Lesson 5 Performing an Advanced Network Discovery

Shows you how to activate a task whose job it is to discover SNMP connection data of the new IPs discovered in your IT infrastructure.

Lesson 6 Expanding the Network Discovery

Shows you how to expand the network discovery to include the discovery of other network resources such as ARP tables and TCP connections.

Lesson 7 Discovering Database Instances and Oracle Resources

Shows you how to discover the database instances and Oracle resources in your IT infrastructure.

Lesson 8 Discovering WebLogic Instances and Components

Shows you how to uncover WebLogic instances and WebLogic components in your IT infrastructure.

Lesson 9 Discovering Host Resources

Shows you how to activate a number of patterns that discover WMI-based resources, such as disks, CPU, memory, or files.

Appendix A Optional Variables in Discovery Patterns

Describes how to add the attribute **optional="true"** to a variable tag in a discovery pattern.

Appendix B Discovery Patterns

Lists the available discovery patterns.

Who Should Read This Guide

This guide is intended for the following users of Mercury Business Availability Center:

- Mercury Business Availability Center administrators
- Mercury Business Availability Center platform administrators
- Mercury Business Availability Center application administrators
- Mercury Business Availability Center data collector administrators

Readers of this guide should be knowledgeable about enterprise system administration, have familiarity with ITIL concepts, and be knowledgeable about Mercury Business Availability Center in general and Mercury Application Mapping technology specifically.

Getting More Information

For information on using and updating the Mercury Business Availability Center, reference information on additional documentation resources, typographical conventions used in the Documentation Library, and quick reference information on deploying, administering, and using Mercury Business Availability Center, refer to *Getting Started with Mercury Business Availability Center*.

1

Introducing the Discovery Process

The Mercury Business Availability Center discovery process is the mechanism that enables you to collect data about your system by discovering the IT infrastructure resources and their interdependencies. It can discover such resources as applications, databases, network devices, different types of servers, and so on. Each discovered IT resource is then delivered and stored in the Mercury Universal CMDB where it is represented as a managed CI.

The Mercury Business Availability Center discovery process is run by activating discovery.

What is a Discovery Pattern?

A discovery pattern is an XML file that defines a discovery task. It contains a description of the CIs and relationships that are created with each specific pattern. The definitions are taken from the Configuration Item Type Model, which contains the definitions of all CIT and relationship types. When the discovery pattern is activated, it discovers CITs and relationships of the types that are described in each pattern, and places them in the CMDB.

What is the Configuration Item Type Model?

By default, the Configuration Item Type Model (as seen in the CI Type Manager tab in Mercury Business Availability Center) is divided into two logical groups.

- CI Types
- Relationships

The Configuration Item Type Model contains the definitions of all the CITs defined in the system and the relationships that define the connection between them. Each CIT has its own attributes, as well as the attributes inherited from its parent CIT. The discovery process uncovers CIs and relationships according to the attributes defined in the Configuration Item Type Model.

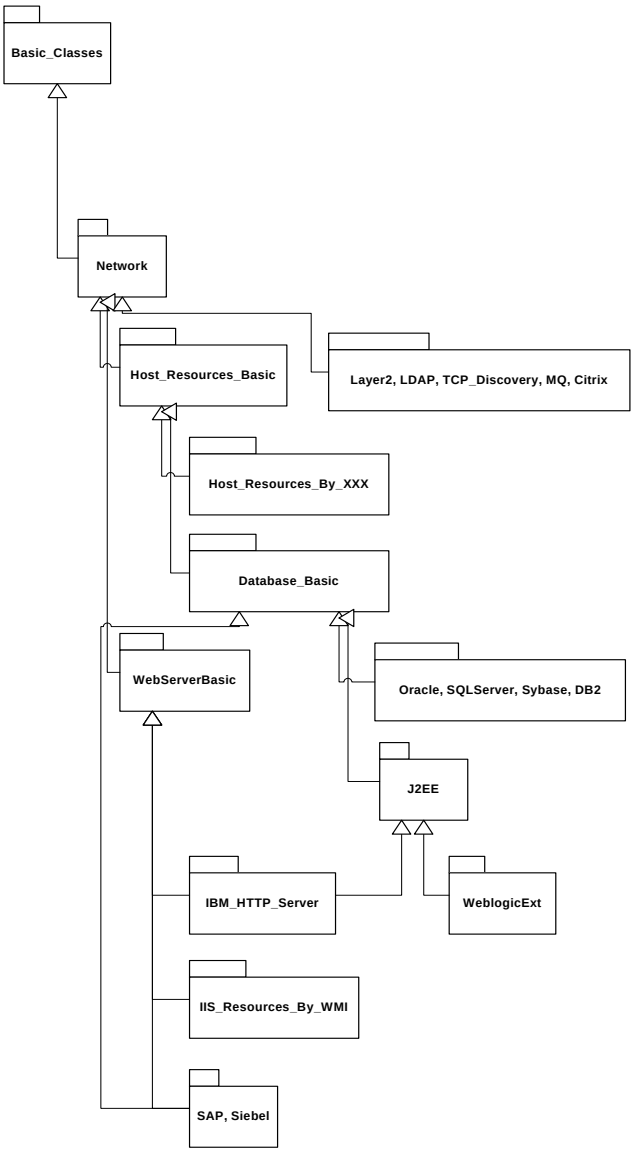
Note: The CIT definitions that appear in the Configuration Item Type Model depend on which packages were deployed. For information on packages, refer to "Customized Package Creation" in *Discovery Manager Administration*.

Running the Discovery Process

The discovery process is a gradual uncovering of the elements in your system. Discovery is first done at the most basic level, and then at more in-depth ones.

After you have installed Mercury Business Availability Center (see "Launching Mercury Business Availability Center" on page 5), the network in which the Discovery Probe is located, the Host on which the Discovery Probe resides, and the Host's IP address are automatically discovered. These discovered CIs are then placed in the CMDB. They act as triggers that activate a discovery pattern. Every time a discovery pattern is activated, it discovers more CIs, which in turn are used as triggers for other discovery patterns. This process continues until your entire IT infrastructure is discovered and mapped.

The following diagram displays the relationship between the deployed packages.



Launching Mercury Business Availability Center

This section describes how to launch Mercury Business Availability Center.

To launch Mercury Business Availability Center:

1. Launch your Web browser and navigate to Mercury Business Availability Center. You log in to Mercury Business Availability Center using the URL provided to you by your Mercury Business Availability Center administrator.

   The default URL is **http://<hostname>/MercuryAM**, where **hostname** is the name of the machine on which the Centers Server is installed.

   The Mercury Business Availability Center login page loads.

2. Enter your assigned username and password.

3. Click **Log In**. The Mercury Business Availability Center default page is displayed.

2

Defining the Seed Network

Once the Mercury Business Availability Center server and the Discovery Probe are connected to a new IT environment, Mercury Business Availability Center automatically identifies the following CIs:

- The network in which the Discovery Probe is located
- The host on which the Discovery Probe resides
- The host's IP address

These CIs are then placed into the CMDB.

Before you can begin the discovery process, you must define the seed network from which to start the process. You can either perform the discovery process by using the seed network that is already in the CMDB by default after installing Mercury Business Availability Center, or define one by manually adding a CI to the CMDB.

If you want to begin the discovery process with the network that is defined in the Discovery Probe, you can use the default seed network. If you want to begin the discovery process with another network, you can define your own seed network. For information on how to define another seed network, see "Inserting a CI Manually" on page 8.

In this lesson, you will learn about:

- "Inserting a CI Manually" on page 8
- "Defining the Discovery Scope" on page 11
- "Verifying that the Changes Have Been Made to the Discovery Probe" on page 15

Inserting a CI Manually

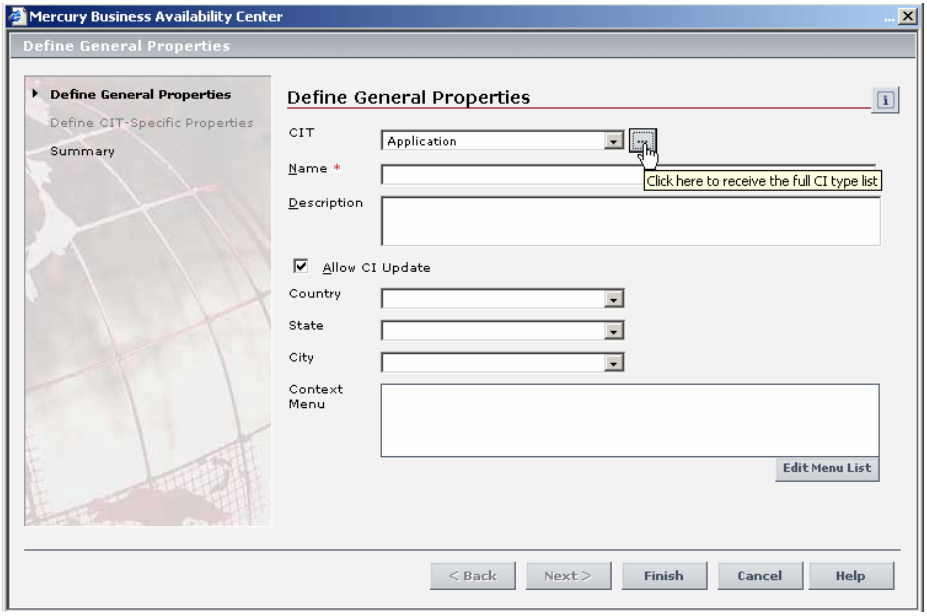
To perform a discovery, you need to choose a seed network from which to perform the discovery process. You can either use the default seed network or define a new one manually.

In this exercise, you will define the seed network manually and then configure its attributes.

To define a seed network:

1 Select **Admin** > **CMDB** and click the **IT Universe** tab.

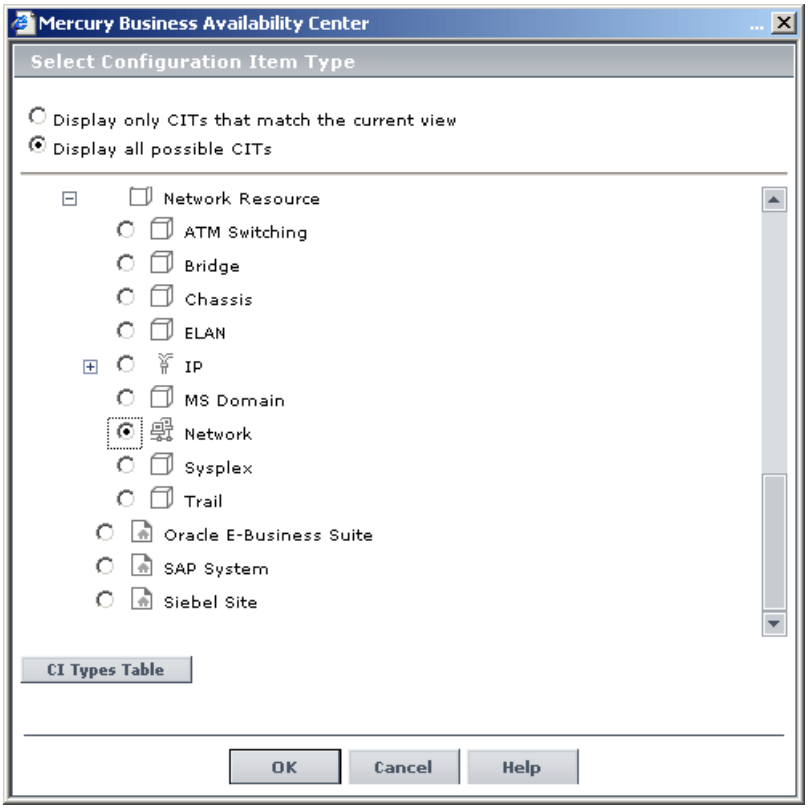
2 Click **Create new CIs** to open the Define General Properties dialog box.



Note: It is not necessary to define the General Properties for this tutorial.



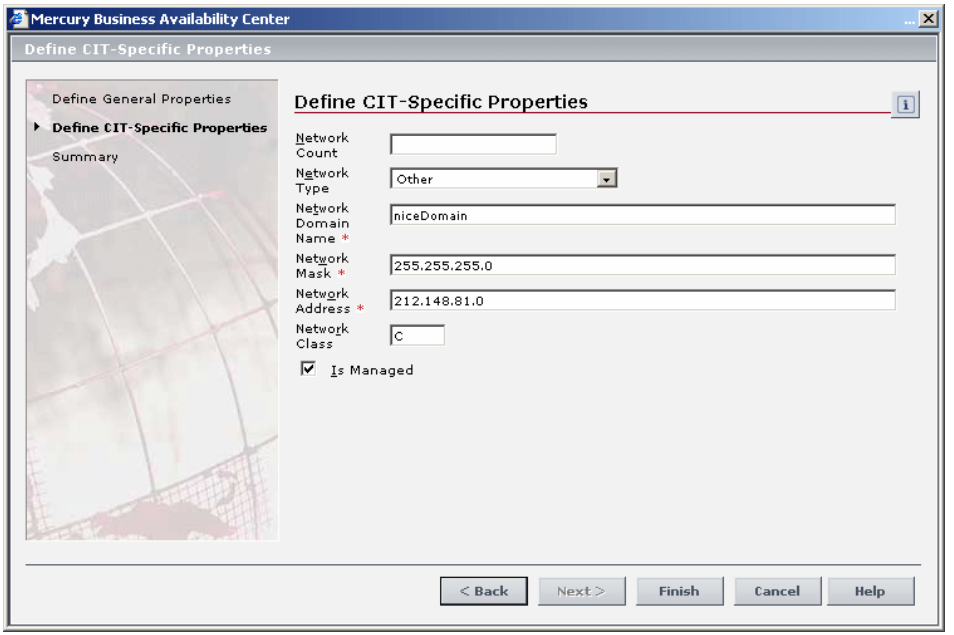
3 Click the button to the right of the **CIT** box to open the Select Configuration Item Type dialog box.



4 At the top, select **Display all possible CITs**. and then select **Network**.

5 Click **OK** to return to the Define General Properties page.

6 Click **Next** to open the Define CIT-Specific properties page.



7 In the **Network Domain Name** box, type the name of the domain as you defined it during installation. For this exercise, type niceDomain.

8 In the **network_netmask** box, and type the net mask of the network for which you want to do the discovery. For example, 255.255.255.0.

9 In the **network_netaddr** box, and type the IP address of the seed network from which you want to start the discovery. For example, 212.148.81.0.

10 In the **Network Class** check box, type C.

11 Ensure that the **Is Managed** check box is selected.

12 Click **Finish** to display the Summary page. The Summary page displays the CIs that were created and added to the view.

13 Click **Close** to close the Summary page and save the network attributes you have defined.

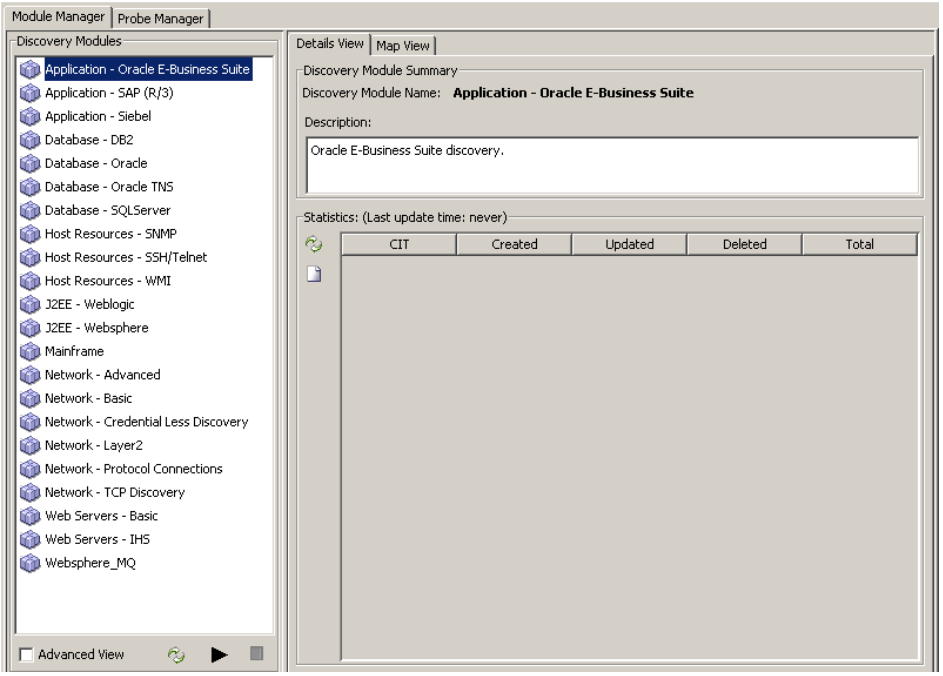
Defining the Discovery Scope

Before you activate the discovery patterns to start collecting data about your network, you need to add a Discovery Probe. For each Discovery Probe, you need to define the discovery scope that defines the range of the IP addresses to be discovered as well as configure the connection data for each protocol included in the discovery process.

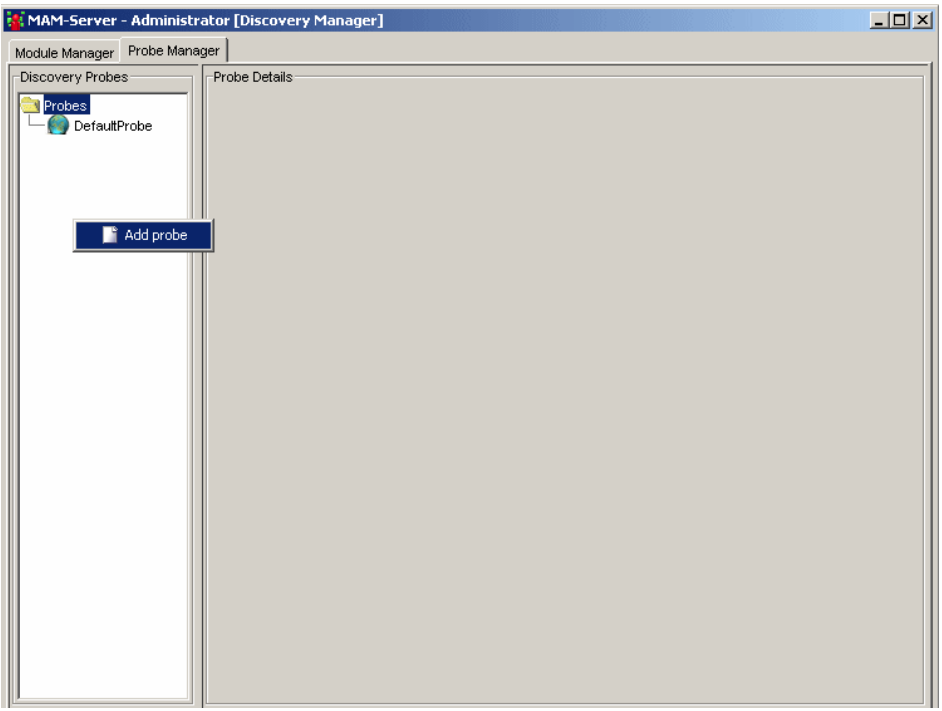
The discovery process can encompass several Discovery Probes. You need to define a separate range for each Discovery Probe. Anything discovered by the discovery patterns outside of the defined range is not included in the discovery process.

To configure the discovery scope:

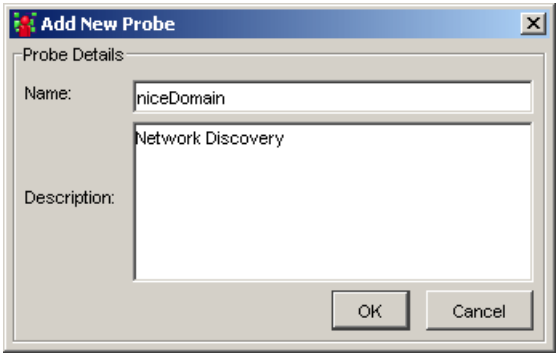
1 Select **Admin** > **CMDB** and click the **Discovery Manager** tab.



2 To add a new Discovery Probe, click the **Probe Manager** tab.



3 In the Discovery Probes pane, right-click the **Probes** folder or any empty area and select **Add probe** to open the Add New Probe dialog box.



4 In the **Name** box, type the name of the Discovery Probe as defined during the installation. For example, niceDomain.

5 In the **Description** box, type description for the Discovery Probe. For this exercise, write Network discovery.

6 Click **OK** to save your changes.



7 Click the **Add IP range** button to open the Add Range dialog box.



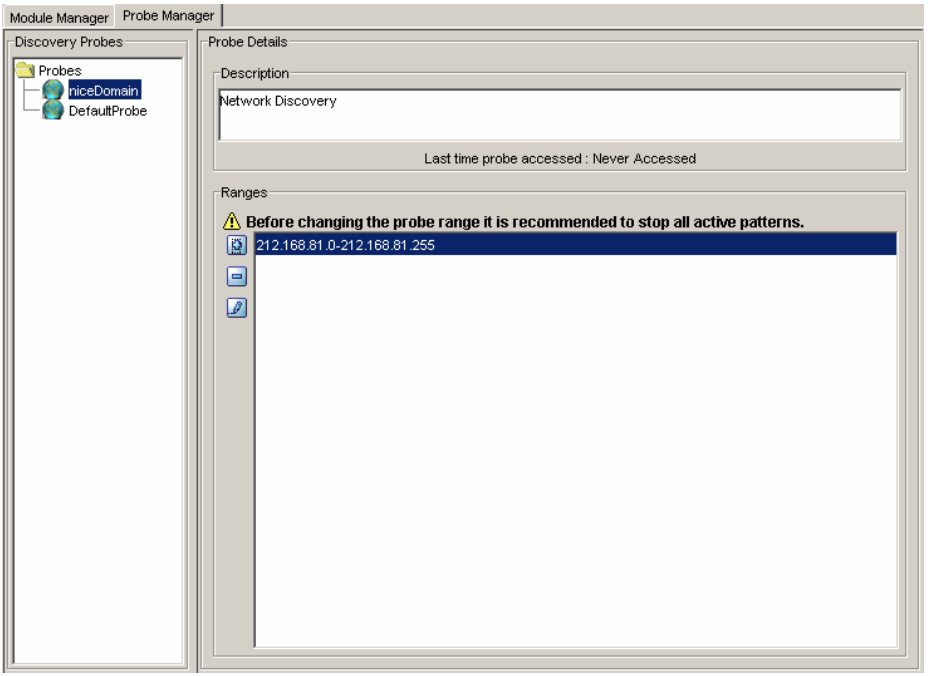
8 Enter an IP address range using the following format:

start_ip_address - end_ip_address

Note: The IP address range can include a wild card character (*) in the lower bound IP address of the IP range pattern. The asterisk represents any number in the range of 0-255. If you use an asterisk, you do not need to enter a second IP address. For example, 10.0.48.* covers the whole range from 10.0.48.0 to 10.0.48.255.

---

9 Click **OK**. The full IP address range appears in the **Ranges** pane, as seen below.

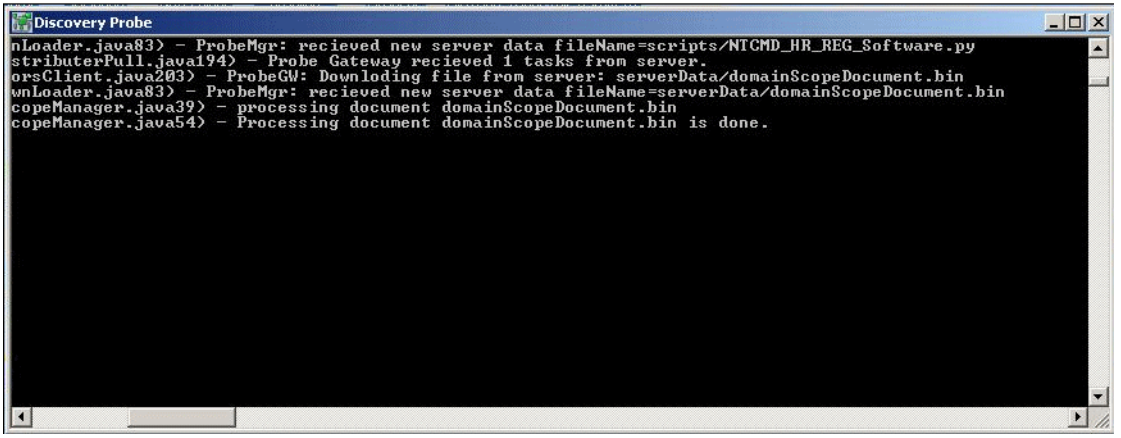


10 Click **Apply** to save the changes you made in the discovery scope configurations in the CMDB.

Verifying that the Changes Have Been Made to the Discovery Probe

The changes you made in the discovery scope configurations are delivered to and stored in the CMDB. From there, the changes are sent to the Discovery Probe. Verification that the changes have been sent to the Discovery Probe is seen in the following message displayed in the **wrapperProbe log** file, that is located in \<**Mercury Business Availability Center root directory**>**root****logs**\.

processing document domainScopeDocument.bin
Processing document domainScopeDocument.bin is done.



In the next lesson, you will activate the discovery pattern that discovers the networks contained within the range defined in this lesson.

3

Discovering Network CIs

The network CIs that were discovered in the previous lesson (see "Defining the Seed Network" on page 7) act as triggers for the continued discovery of other resources. This applies regardless of whether the default seed network was used to start the discovery or one was defined manually.

In order for the discovered network CIs in the CMDB to act as triggers for discovering other resources, the relevant discovery patterns must be activated.

In this lesson, you will activate the discovery pattern **ICMP_NET_Dis_IpC**, which is designed to discover the network IPs that fall within the IP address range as defined in the Discovery Manager dialog box in "Defining the Discovery Scope" on page 11.

In this lesson, you will learn about:

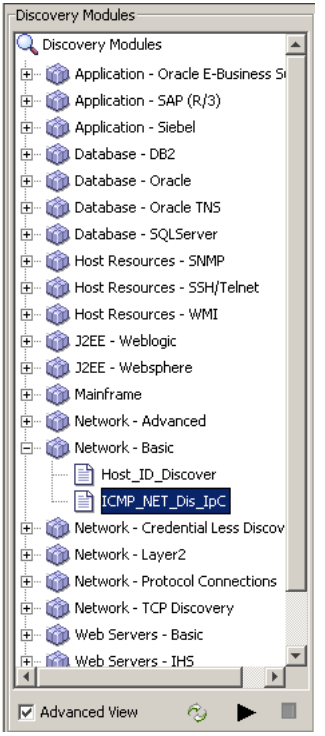
- "Activating the ICMP_NET_Dis_IpC Discovery Pattern" on page 18
- "What Happens When You Activate the ICMP_NET_Dis_IpC Pattern?" on page 19

Activating the ICMP_NET_Dis_IpC Discovery Pattern

In this section, you will activate the **ICMP_NET_Dis_IpC** pattern. To activate discovery patterns, you must select the relevant patterns from the Discovery Manager.

To activate the ICMP_NET_Dis_IpC pattern:

1. Select **Admin** > **CMDB** and click the **IT Universe** tab to open the Discovery Manager.
2. Click the **Module Manager** tab and select the **Advanced View** check box.
3. Click the **Expand** button to the left of the **Network - Basic** module.



4. Right-click **ICMP_NET_Dis_IpC** and click **Activate** to activate the pattern.

What Happens When You Activate the ICMP_NET_Dis_IpC Pattern?

For every network in the CMDB, Mercury Business Availability Center takes the network address and the network mask and calculates the range of the IP addresses you want to discover.

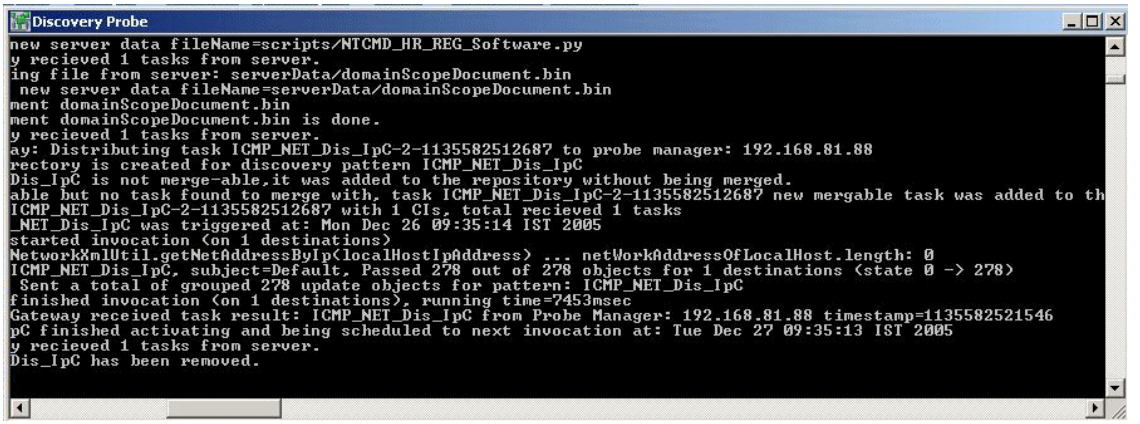
This pattern then activates a task whose job is to ping all the IP addresses that were calculated. For every IP address that answers the ping request, Mercury Business Availability Center creates a CI in the CMDB.

Note: Only the IP addresses that are considered to be inside the scope defined in the Discovery Manager dialog box will be pinged.

By identifying the network's IPs, new IPs are discovered on the network. All IP addresses that respond to the ping request are the newly discovered CIs that are added to the CMDB which, in turn, act as triggers to activate other discovery patterns.

Verifying the Discovery Results

CIs that are discovered are delivered to and stored in the CMDB. Verification that the CIs have been sent to the CMDB can be seen here.



The Discovery Probe indicates that the name of the pattern that was activated and the number of network CIs discovered.

This example shows that the **ICMP_NET_Dis_IpC** discovery pattern was activated and 278 CIs were discovered.

In the next lesson, you will define a TQL that retrieves the network CIs from the CMDB so you can see the results of the discovery.

4

Creating a TQL Query

In the previous lesson, you activated the discovery pattern that discovered the networks that fell within the IP address range you defined in Lesson 2. To see the discovered network CIs, you need to define a TQL query that retrieves the specified network CIs from the CMDB.

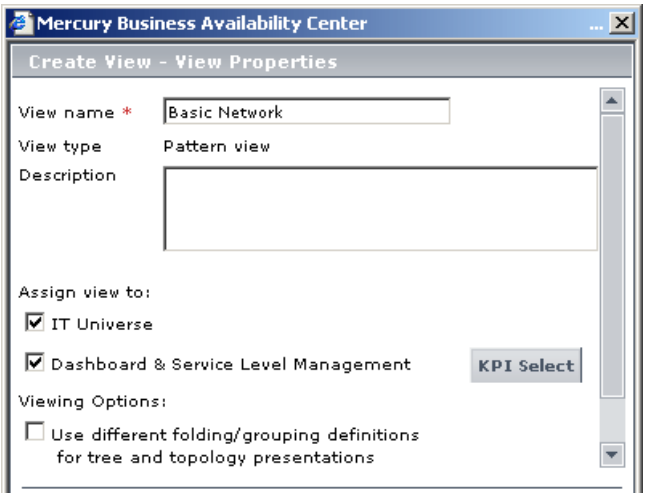
Defining a TQL Query to View the Discovered Network CIs

In this section, you will create the pattern view in which you want to define the TQL query. Then you will add the TQL nodes to the pattern view and define the relationship between them.

To define a TQL query to view the discovered network CIs:

1 In the **View Manager** tab, select the folder in the View Explorer in which you want to place the view.

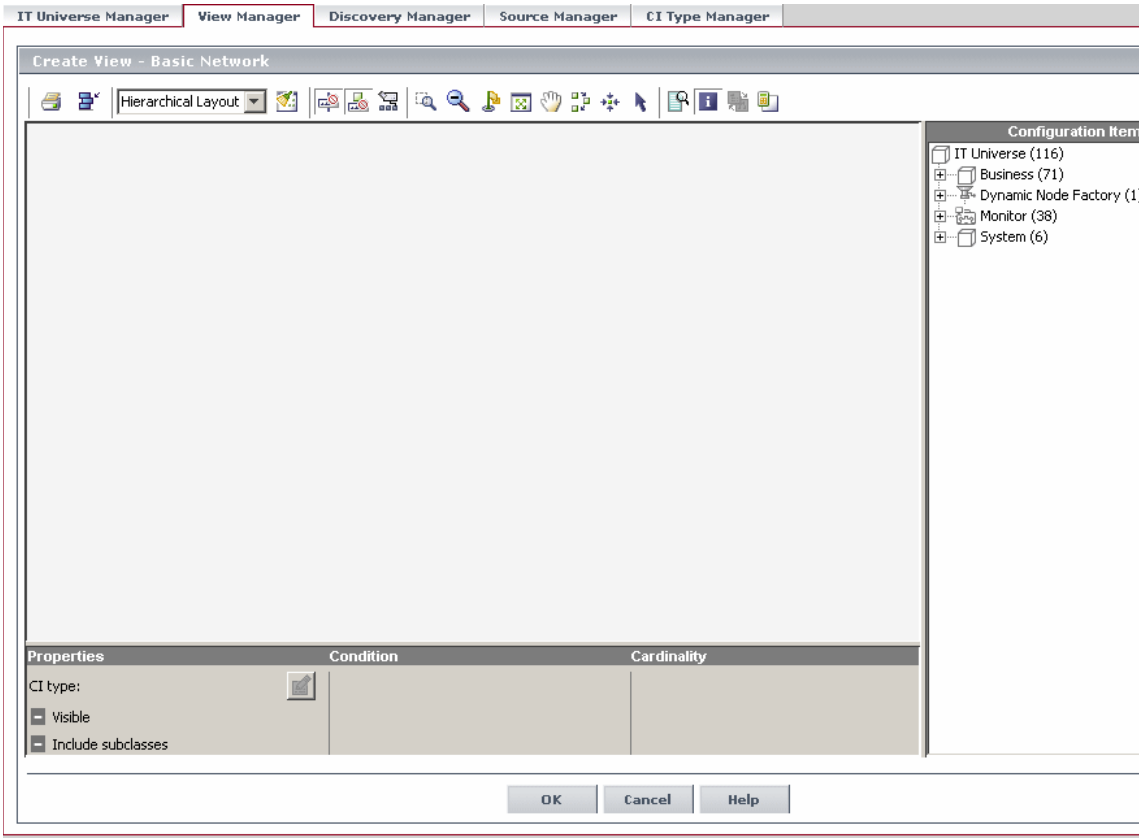
2 Click **New Pattern View** open the Create View - View Properties dialog box.



3 In the **View name** box, type Basic Network.

4 Select the **IT Universe** and **Dashboard & Service Level Management** check boxes.

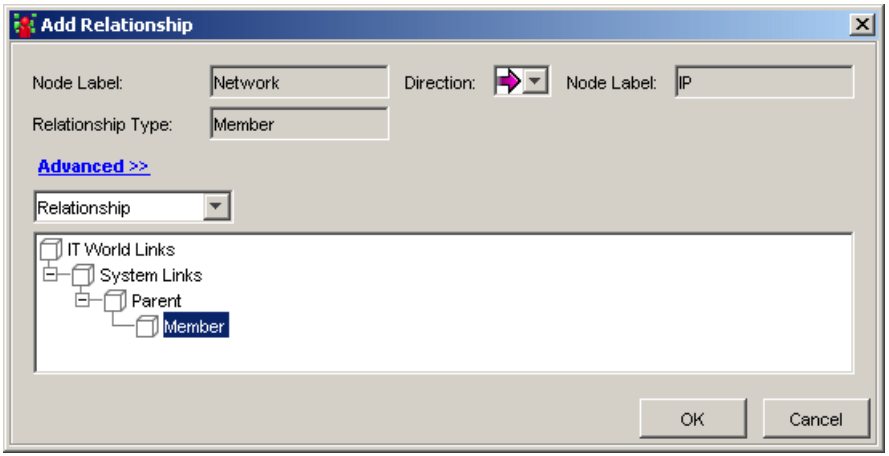
5 Click **OK** to open the Create View window.



6 From the tree displayed in the Configuration Item Type Model, click and drag the following TQL nodes to the Topology map:

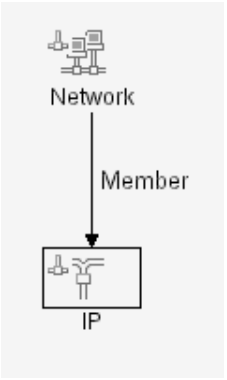
- Network
- IP

7 Select the two nodes, right-click and then click **Add Relationship** to open the Add Relationship dialog box.



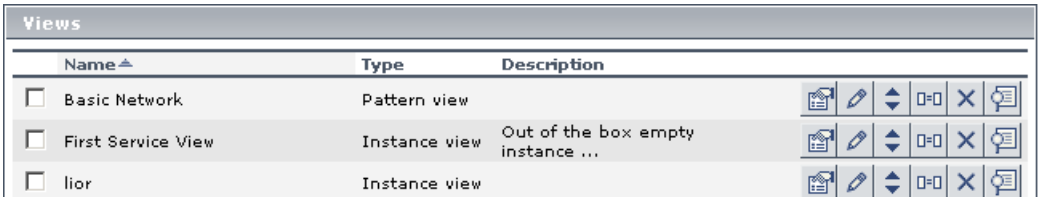
8 To link the **Network** and **IP** TQL nodes, click **Advanced** and select **Member**. **Member** appears in the **Relationship Type** box.

9 Click **OK**. The TQL query you have created is displayed below.



10 Click **OK** to save the TQL definitions in the CMDB.

The pattern view you have created appears in the Views pane.



5

Performing an Advanced Network Discovery

In Lesson 3, you activated the **ICMP_NET_Dis_IpC** discovery pattern, which identified all the network IPs. After these IP addresses are added to the CMDB, they act as triggers for the **SNMP_NET_Dis_Connection** discovery pattern. This pattern activates a task whose job it is to discover SNMP connection data of the new IPs discovered in your IT infrastructure. The task results add a host to each IP together with its SNMP connection data to the CMDB.

In this lesson, you will define the SNMP connection data and activate the **SNMP_NET_Dis_Connection** pattern that discovers hosts that use the SNMP protocol.

In this lesson, you will learn about:

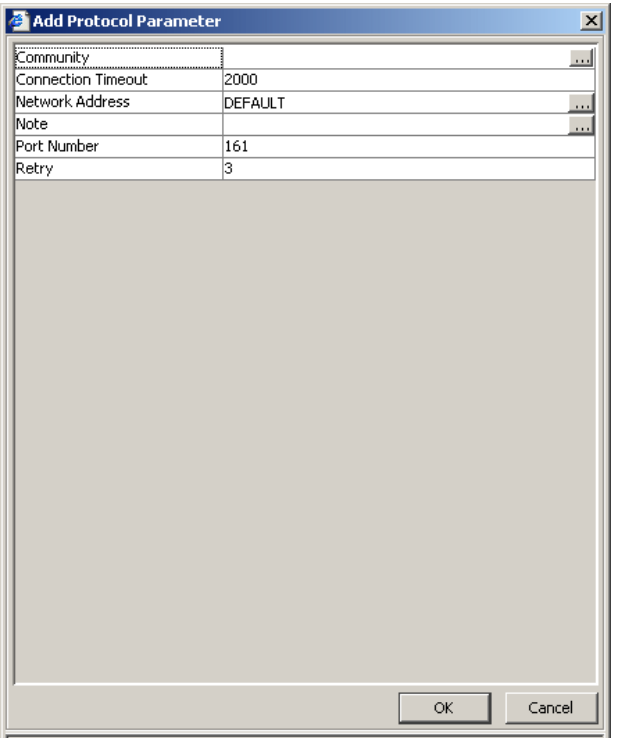
- "Defining the SNMP Connection Data" on page 28
- "Activating the SNMP_NET_Dis_Connection Discovery Pattern" on page 32
- "Defining a TQL to View the Discovered CIs" on page 35
- "Adding TQL Nodes and Relationships to the Query" on page 36
- "Creating a New Pattern View" on page 35
- "Adding TQL Nodes and Relationships to the Query" on page 36

Defining the SNMP Connection Data

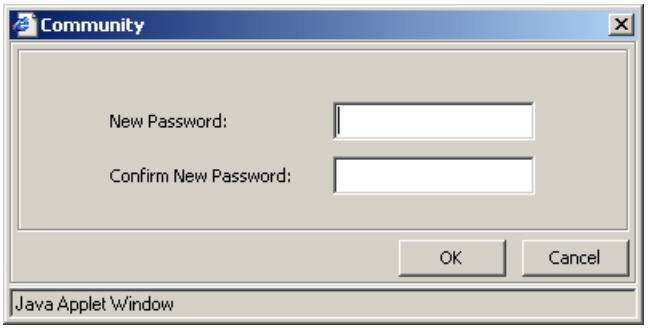
In this section, you will define the SNMP protocol through which the data will be collected.

To define the SNMP connection data:

1. Select **Admin** > **CMDB** and click the **IT Universe** tab to open the Discovery Manager.
2. Click the **Probe Manager** tab.
3. In the Discovery Probes pane, select **niceDomain**.
4. Click the **Expand** button to the left of **niceDomain** and then select **SNMP Protocol**.
5. Click the **Add new connection details for the selected protocol type** button in the Protocol entries pane to open the Add Protocol Parameter dialog box.



6 Click the button at the right end of the **Community** box to open the Community dialog box.



7 Ask your system administrator what the Community string is and type it in the **New Password** box.

8 Type the Community string again in the **Confirm New Password** box and click **OK**.

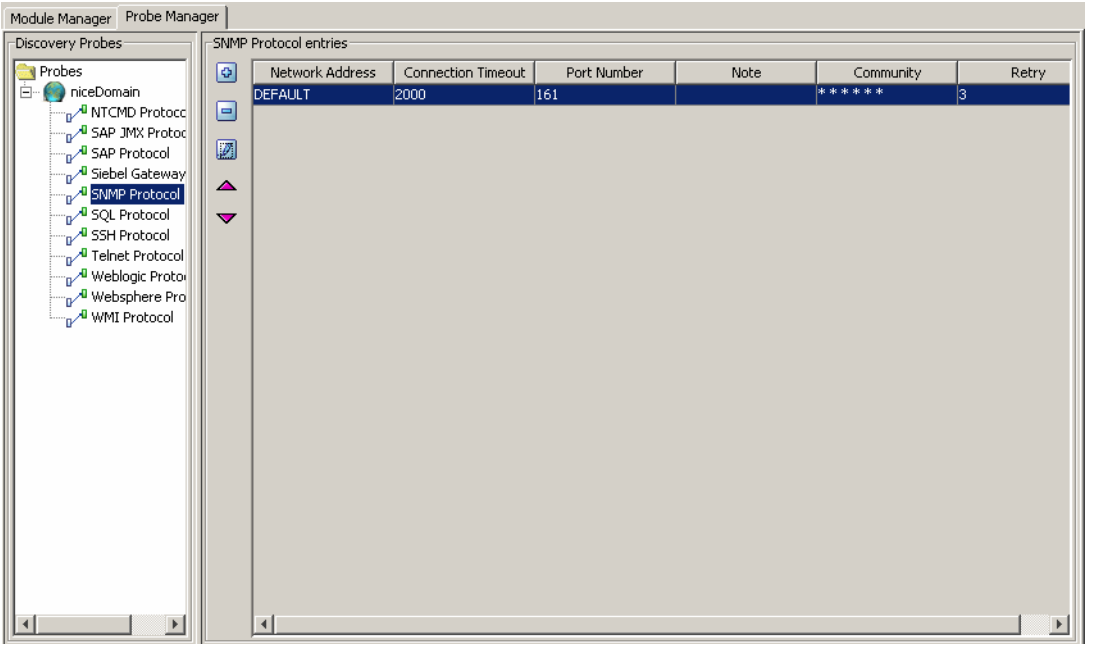
9 In the **Connection Timeout** box, leave the default value 2000.

10 In the **Network Address** box, leave the default value DEFAULT.

11 In the **Port Number** box, ask your system administrator for the required port number.

12 In the **Retry** box, leave the default value 3.

13 Click **OK**. The parameter values you have defined appear in the **Protocol Entries** section, as seen below.

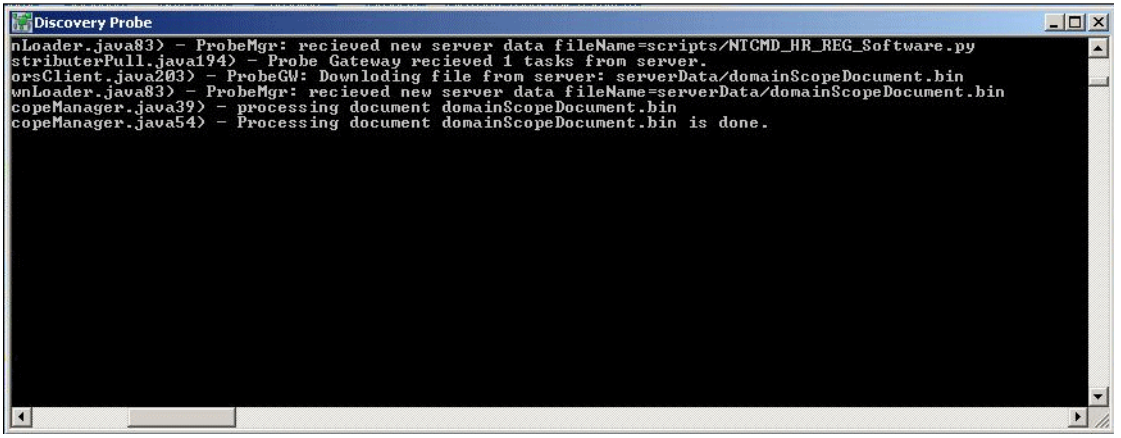


14 Click **Apply** to save the changes you have made in the CMDB.

Verifying That the Changes Have Been Made to the Discovery Probe

Each change you make in the Discovery Manager dialog box is delivered to and stored in the CMDB. From there, the changes are sent to the Discovery Probe. Verification that the changes have been sent to the Discovery Probe is seen in the following message displayed in the **wrapperProbe log** file, that is located in \<**Mercury Business Availability Center root directory**>**root****logs**\.

processing document domainScopeDocument.bin
Processing document domainScopeDocument.bin is done.



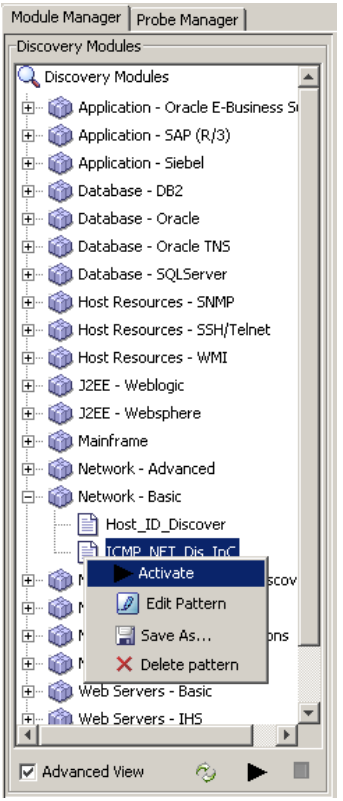
Activating the SNMP_NET_Dis_Connection Discovery Pattern

In this section, you will activate the **SNMP_NET_Dis_Connection** discovery pattern to discover hosts that use the SNMP protocol.

To activate the SNMP_NET_Dis_Connection pattern:

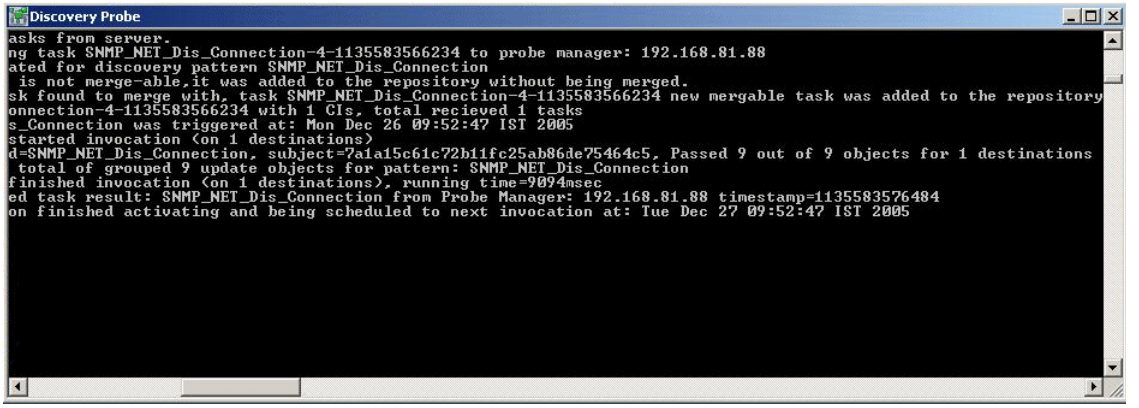
1. Select the **Module Manager** tab.
2. Select the **Advanced View** check box.
3. In the Discovery Modules pane, click the **Expand** button to the left of the **Network- Basic** module.

4 Right-click **SNMP_NET_Dis_Connection** and click the **Activate** button, or select **SNMP_NET_Dis_Connection** and click the **Activate** button in the bottom- right corner of the Discovery Modules pane.



Verifying the Discovery Results

Check the discovery results in the Discovery Probe. The following example shows that the **SNMP_NET_Dis_Connection** pattern was activated and displays the number of CIs discovered.



Defining a TQL to View the Discovered CIs

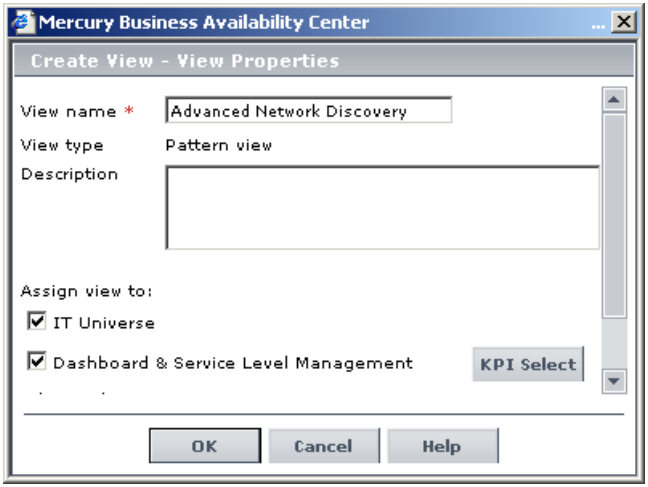
To view the discovered network CIs, you will be:

- "Creating a New Pattern View" on page 35
- "Adding TQL Nodes and Relationships to the Query" on page 36

Creating a New Pattern View

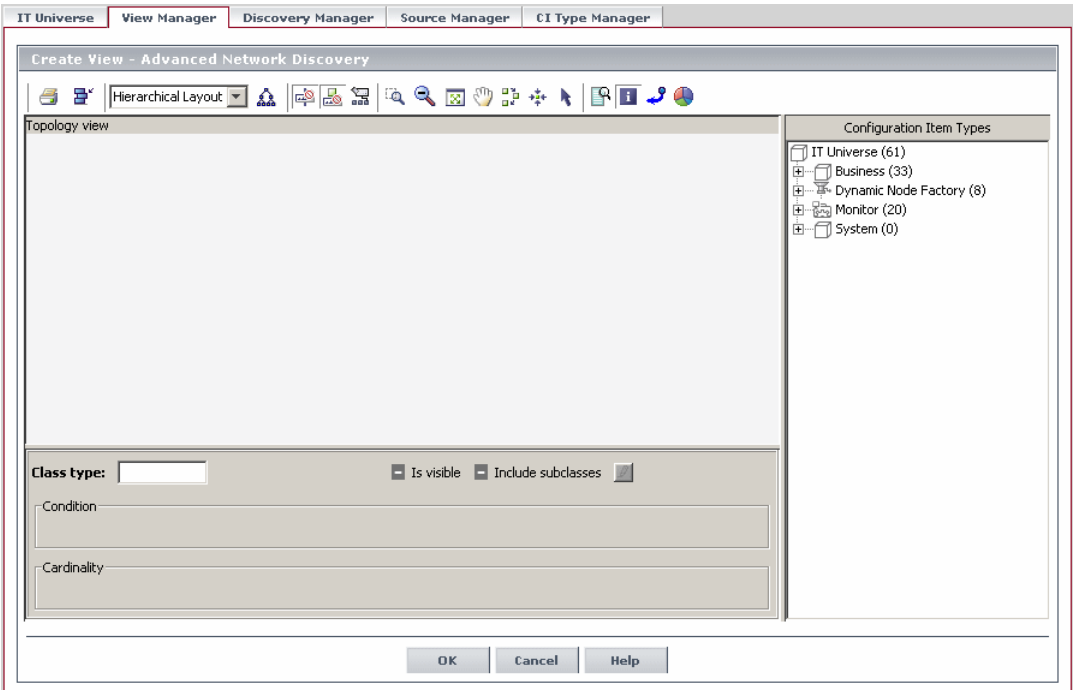
In this section, you will create a pattern view whose topology map displays the results of the TQL query.

1. Select **Admin** > **CMDB** and then click the **View Manager** tab.
2. In the View Explorer, select the folder in which you want to place the view.
3. Click **New Pattern View** to open the Create View - View Properties dialog box.



4. In the **View name** box, type a unique name for the pattern view. For this exercise, type Advanced Network Discovery.
5. Select the check boxes of the applications in which you want the view you are creating to appear:
   - IT Universe
   - Dashboard & Service Level Management

6 Click **OK** to display the Create View window.



Adding TQL Nodes and Relationships to the Query

After you define the TQL, you must add the required CIs and define the relationship between them.

To add nodes and relationships to the TQL:

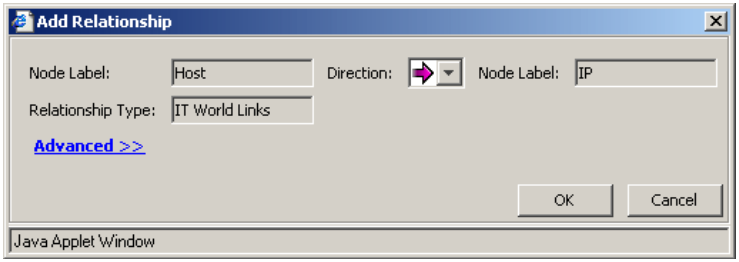
1 From the tree displayed in the Configuration Item Type Model, click and drag the following CITs to the topology map.

- Network
- IP
- Host
- SNMP

2 Link the nodes according to the following table.

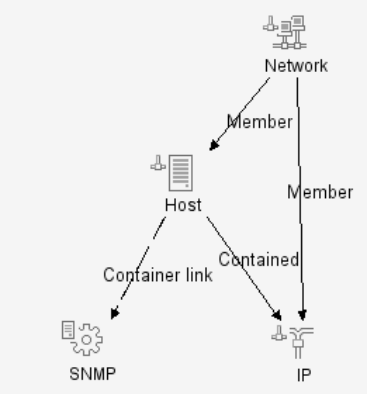
Link this node	To this node	With this relationship
IP	Network	Member
Host	Network	Member
SNMP	Host	Container Link
IP	Host	Contained

Simultaneously select the two nodes in each row, right-click and then click **Add Relationship**. The Add Relationship dialog box opens.



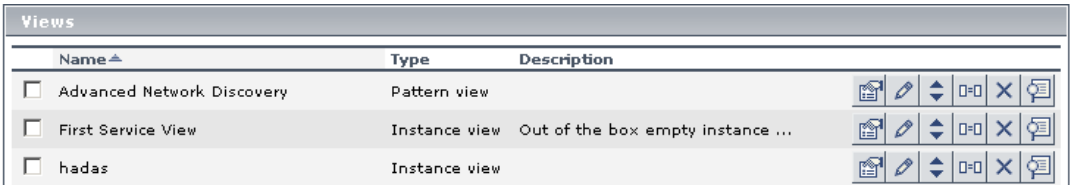
3 Select the relationships according to the table above, and click **OK**.

The TQL you have created is displayed below.



4 Click **OK** to save the TQL definitions in the CMDB.

The pattern view you have created appears in the Views pane.



In the following lesson, you will expand the discovery to include other network resources.

6

Expanding the Network Discovery

In this lesson, you will expand the network discovery to include the discovery of other network resources.

In this lesson, you will learn about:

- "Activating Patterns that Expand the Network Discovery" on page 40
- "Viewing the Discovered CIs" on page 43

Activating Patterns that Expand the Network Discovery

The following table contains a list of discovery patterns that activate tasks whose job is to discover other network components needed for building the network infrastructure, such as relationships, ARP tables and port numbers.

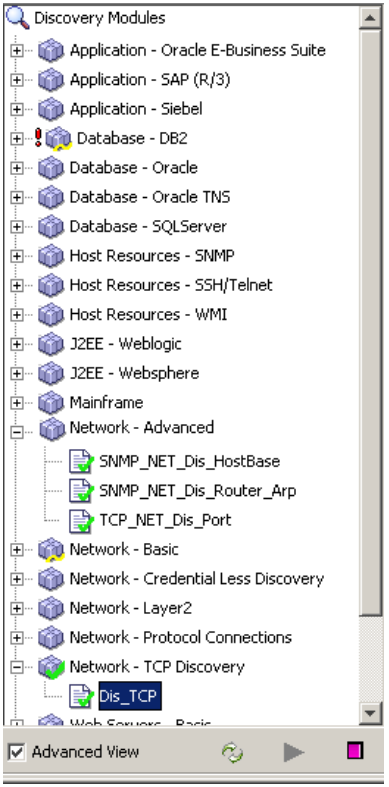
Discovery Pattern	Definition
SNMP_NET_Dis_HostBase	Activates a task whose job is to discover all the routing relationships between the hosts in your system.
SNMP_NET_Dis_Router_Arp	Activates a task whose job is to discover the ARP tables containing the IP addresses of the machines with whom the server is communicating.
SNMP_NET_Dis_TCP	Activates a task whose job is to discover all the TCP connections between the different machines in your system.
TCP_NET_Dis_Port	Activates a task whose job is to discover all the port numbers in the **portNumberToPortName.xml** file, which is located in: **<Mercury Business Availability Center root directory>\root\lib\collectors\discoveryManager\serverData** **Note**: This file is provided with your **Mercury Business Availability Center** package. You can edit the file if required. For details, see Discovery Process Configuration Files in *Discovery Manager Administration*. The results of this discovery become the trigger CIs for discovering applications.

To activate the discovery patterns:

1. Select **Admin** > **CMDB** and select the **Discovery Manager** tab to open the Discovery Management.
2. Select the **Module Manager** tab.
3. Select the **Advanced View** check box.
4. Click the **Expand** button to the left of the **Network - Advanced** module.
5. Select the following patterns:
   - **SNMP_NET_Dis_HostBase**
   - **SNMP_NET_Dis_Router_Arp**
   - **TCP_NET_Dis_Port**
6. Click the **Expand** button to the left of the **Network - TCP Discovery** module.

7 Select the following pattern:

➤ **Dis_TCP**.



8 The activated patterns, as seen here, is marked with a green check.

Viewing the Discovered CIs

Mercury Business Availability Center provides predefined views for certain discovery results. You can view the following discovered CIs in the following predefined views:

View these CIs	In this predefined view
All the TCP connections between the different machines in your system	Client_Server_Connections
All the routing relationships between the hosts in the network.	Route
All the ARP tables containing the IP addresses of the machines with whom your computer is communicating.	Network

---

Note: Mercury Business Availability Center does not provide a predefined view for the port numbers in the **portNumberToPortName.xml** file.

7

Discovering Database Instances and Oracle Resources

The CMDB now contains networks, host CIs with SNMP connection data and other network resources. In this lesson, you will uncover the database instances and Oracle resources in your IT infrastructure.

The **SQL_NET_Dis_Connection** pattern discovers the following database types:

- Oracle
- DB2
- Sybase
- SQLServer

The CIs discovered in the **TCP_NET_Dis_Port** pattern (see See "TCP_NET_Dis_Port" on page 40.) act as a trigger for the **SQL_NET_Dis_Connection** pattern, which activates a task whose job is to discover database instances.

In this lesson, you will learn about:

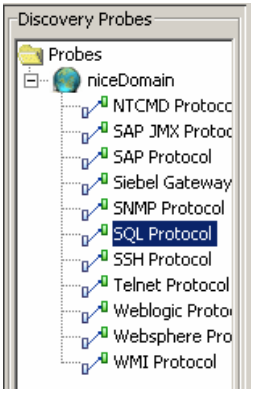
- "Adding the SQL Protocol" on page 46
- "Activating the SQL_NET_Dis_Connection Pattern" on page 49
- "Activating the SQL_APP_Dis_Oracle Discovery Pattern" on page 51

Adding the SQL Protocol

You need to add the Oracle protocol to discover all the Oracle resources.

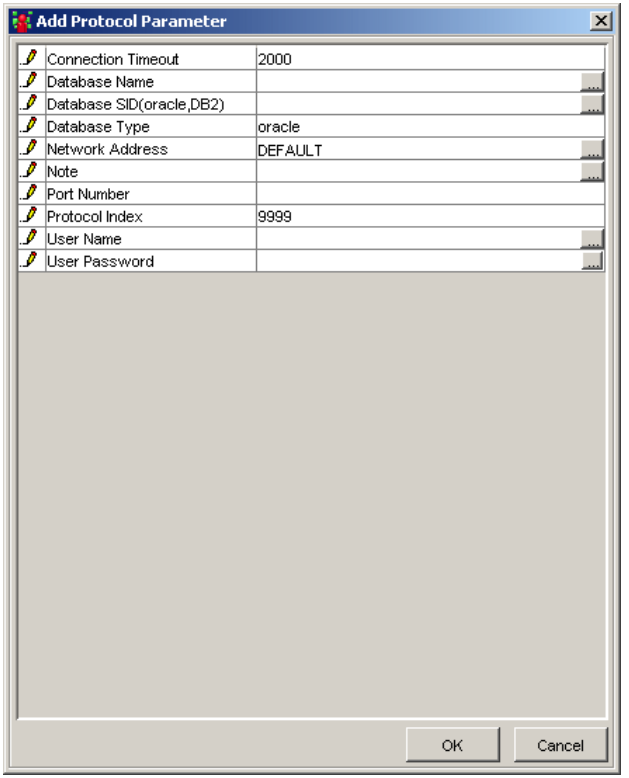
To add the Oracle protocol:

1. Select **Admin** > **CMDB** and select the **Discovery Manager** tab to open the Discovery Manager.
2. Click the **Probe Manager** tab.
3. In the Discovery Probes pane, select **niceDomain**.
4. Click the **Expand** button to the left of **niceDomain** and then select **SQL Protocol**.



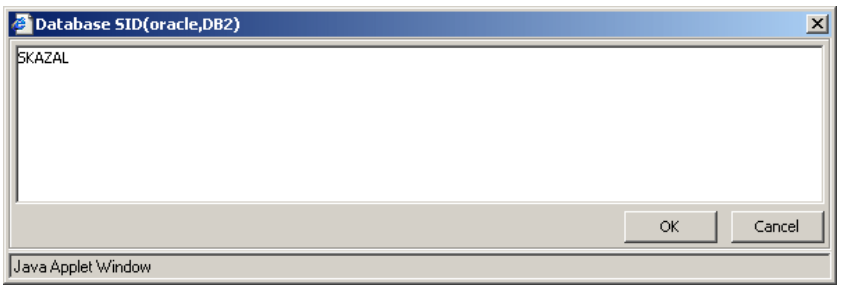


5 Click the **Add new connection details for the selected protocol type** button in the Protocol entries pane to open the Add Protocol Parameter dialog box.



6 In the **Connection Timeout** box, leave the default as 2000.

7 Click the button at the right of the **Database SID(oracle, DB2)** box. In the dialog box that opens, type the name of your database SID. For example, SKAZAL.



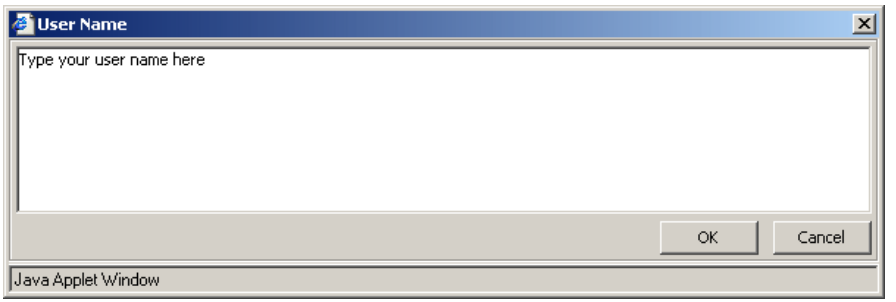
8 Click **OK** to save your changes.

9 In the **Database Type** box, leave the default value oracle

10 In the **Network Address** box, leave the default value DEFAULT.

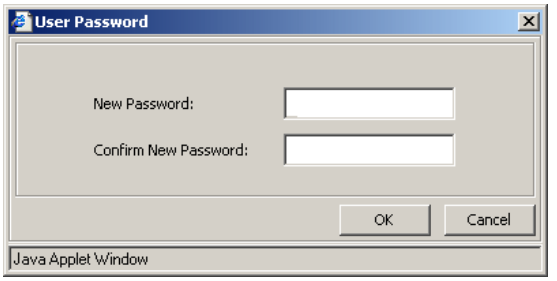
11 In the **Port Number** box, type the port number on which the database listens.

12 Click the button at the right end of the **User Name** box. In the dialog box that opens, type your username.



13 Click **OK** to save your changes.

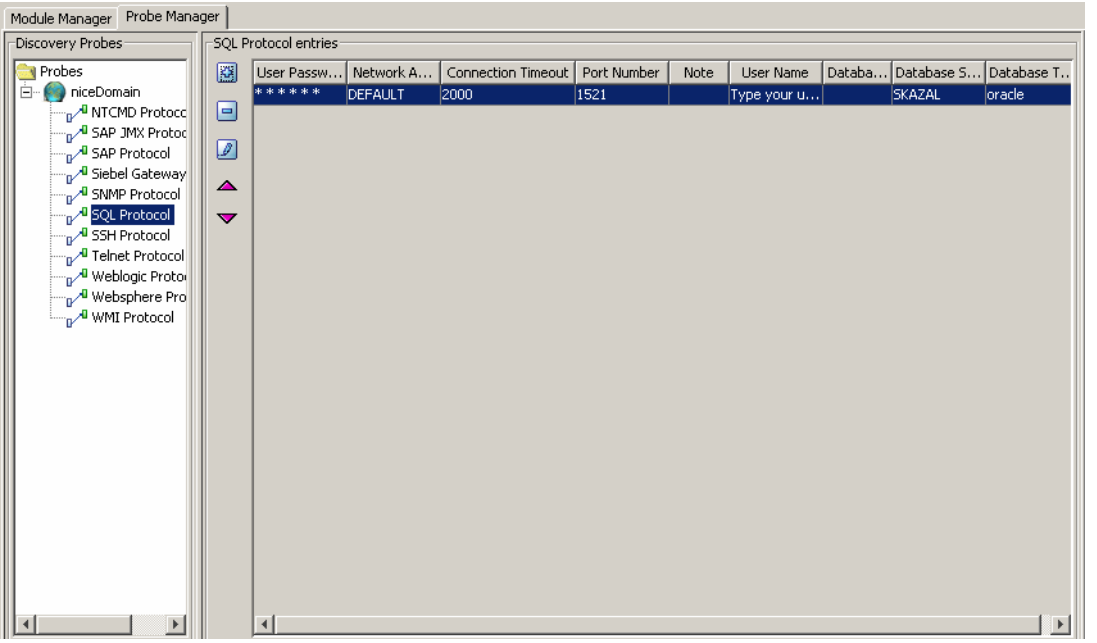
14 Click the button at the right end of the **User Password** box to open the User Password dialog box.



15 In the **New Password** box, type your password.

16 Type your password again in the **Confirm New Password** box and click **OK** to save the password information and close the User Password dialog box.

17 Click **OK** to save the protocol definitions you have set. The protocol definitions appear in the **Protocol Entries** section.



18 Click **Apply** again to save the changes in the CMDB.

To verify that the CMDB has been updated with the changes you made in the network protocol configurations, check that the following notification appears in the Discovery Probe:
Processing document domainScopeDocument.bin is done

Activating the SQL_NET_Dis_Connection Pattern

In this section, you will activate the pattern that discovers database instances in your IT infrastructure.

To activate the SQL_NET_Dis_Connection pattern:

1 Select the **Module Manager** tab.

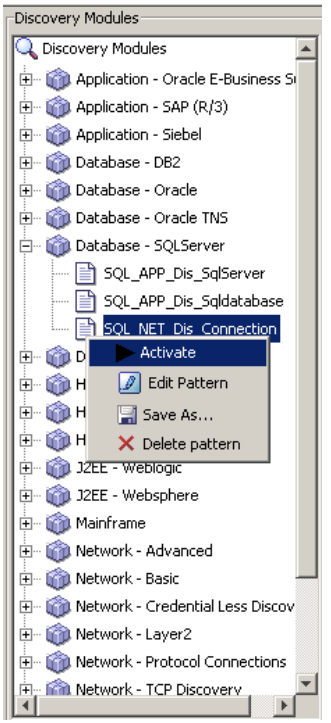
2 Select the **Advanced View** check box.



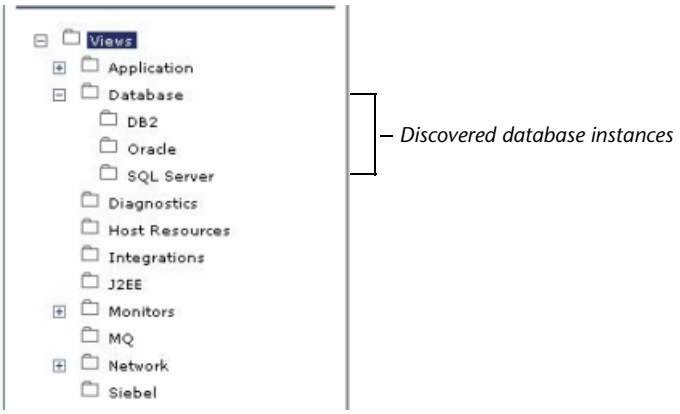
3 In the Discovery Modules pane, click the **Expand** button to the left of the **Database - SQLServer** module.



4 Right-click S**QL_NET_Dis_Connection** and click the **Activate** button, or select S**QL_NET_Dis_Connection** and click the **Activate** button in the bottom- right corner of the Discovery Modules pane.



The pattern uncovers all Oracle, DB2, Sybase and SQLServer database instances that exist in your IT infrastructure. They appear in a predefined folder provided by the system called **Database**.



Activating the SQL_APP_Dis_Oracle Discovery Pattern

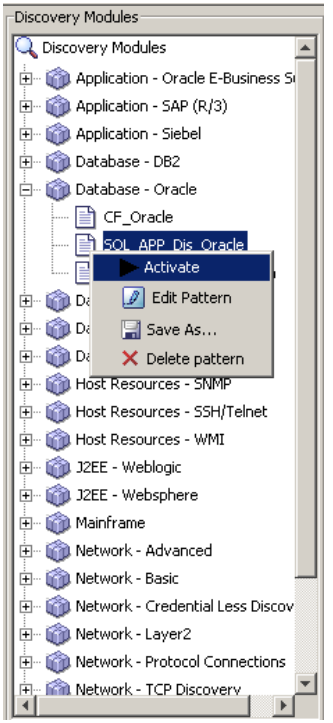
Now that you have discovered all the instances of Oracle, DB2, Sybase and SQLServer databases, you will perform a more in-depth discovery that uncovers all the existing Oracle resources. To do this, you need to Activate the **SQL_APP_Dis_Oracle** pattern

In this section, you will activate the discovery pattern **SQL_APP_Dis_Oracle** whose task is to discover Oracle resources.

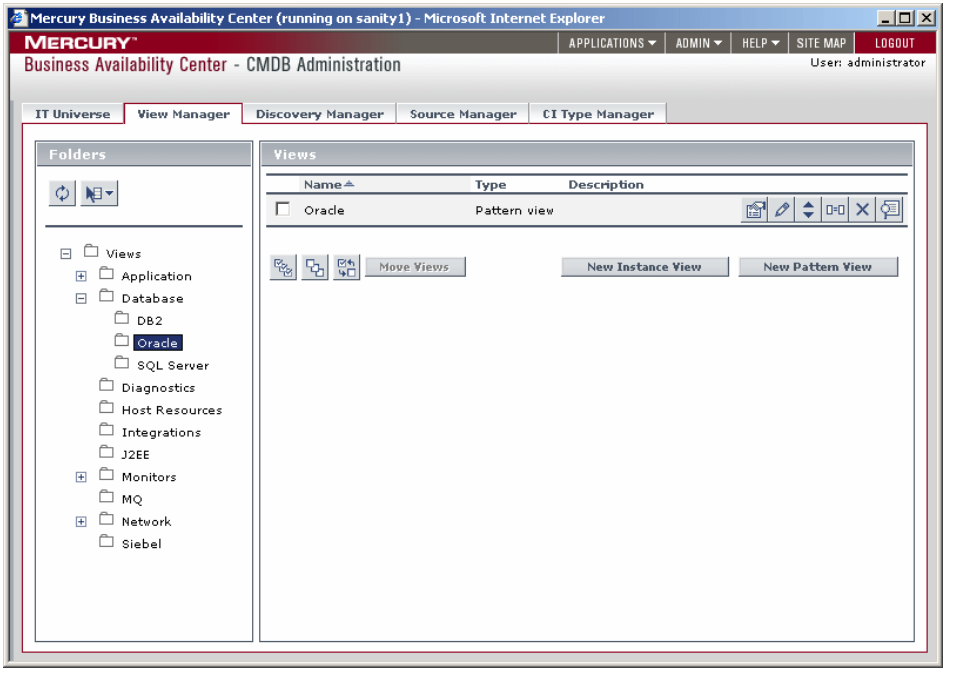
To discover Oracle resources:

1. Select the **Module Manager** tab.
2. Select the **Advanced View** check box.
3. Click the **Expand** button to the left of the **Database - Oracle** module.

4 Right-click **SQL_APP_Dis_Oracle** and click the **Activate** button, or select **SQL_APP_Dis_Oracle** and click the **Activate** button in the bottom- right corner of the Discovery Modules pane.



The pattern uncovers all the Oracle resources, such as users, tables and tablespaces for each database instance. The discovered resources appear in a predefined view called **Oracle**.



All the existing Oracle resources that are discovered can be viewed in the topology map in IT Universe (for details, see *IT Universe Administration*.

8

Discovering WebLogic Instances and Components

In the previous lesson, you discovered database instances and the Oracle resources in your IT infrastructure. In this lesson, you will uncover WebLogic instances and WebLogic components in your IT infrastructure.

- You activate the **J2EE_JMX_Weblogic_Connection** pattern to discover the WebLogic instances.

  The CIs discovered in the **TCP_NET_Dis_Port** pattern (see "Activating Patterns that Expand the Network Discovery" on page 40), act as a trigger for the **J2EE_JMX_Weblogic_Connection** pattern, which activates the task whose job is to discover all instances of WebLogic.

- You activate the **J2EE_JMX_Weblogic** pattern to discover the WebLogic components.

  The CIs discovered in the **J2EE_JMX_Weblogic_Connection** pattern act as a trigger for the **J2EE_JMX_Weblogic** pattern, which activates the task whose job is to discover all WebLogic components.

In this lesson, you will learn about:

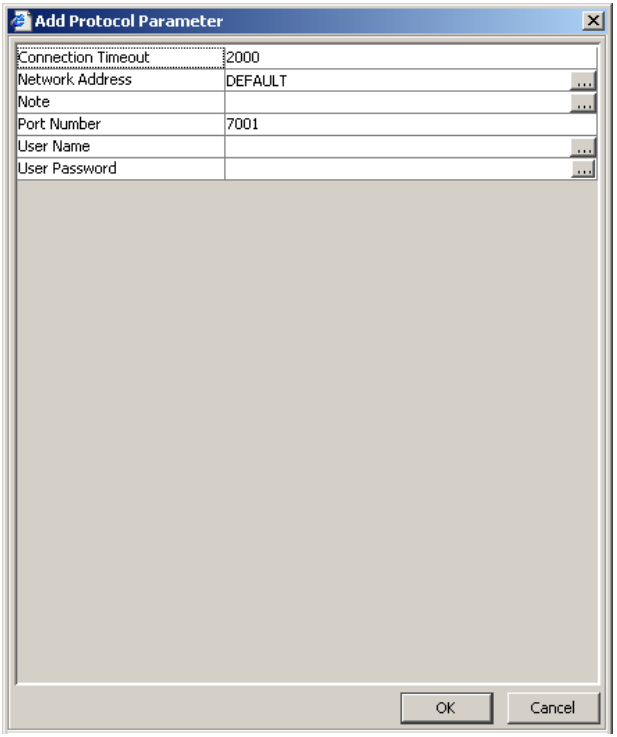
- "Defining the WebLogic Protocol" on page 56
- "Discovering WebLogic Instances" on page 59
- "Discovering WebLogic Components" on page 61

Defining the WebLogic Protocol

In this section, you will add the WebLogic protocol and define its connection data.

To define the WebLogic Protocol:

1. Select **Admin** > **CMDB** and click the **IT Universe** tab to open the Discovery Manager.
2. Click the **Probe Manager** tab.
3. In the Discovery Probes pane, select **niceDomain**.
4. Click the **Expand** button to the left of **niceDomain** and then select **Weblogic Protocol**.
5. Click the **Add new connection details for the selected protocol type** button in the Protocol entries pane to open the Add Protocol Parameter dialog box.

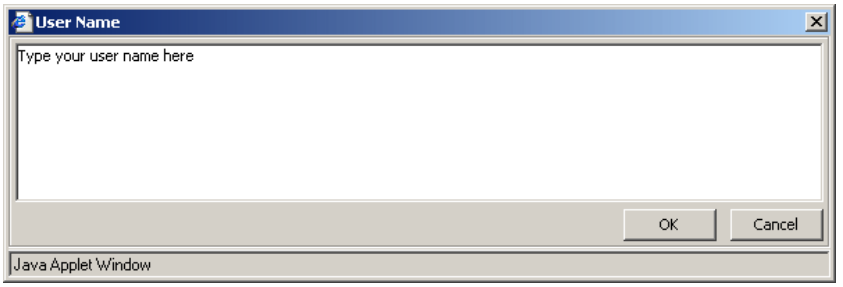


6 In the **Connection Timeout** box, leave the default value as 2000.

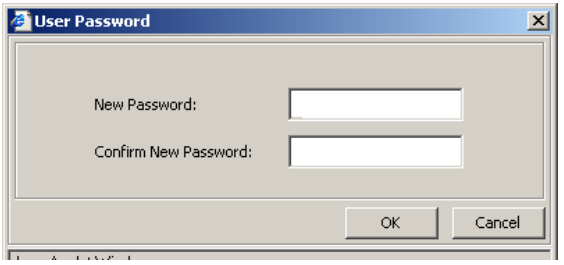
7 In the **Network Address** box, leave the default value as DEFAULT.

8 In the **Port Number** box, type the port number on which the Weblogic server listens.

9 Click the button at the right end of the **User Name** box. In the dialog box that opens, type your user name and click **OK**.



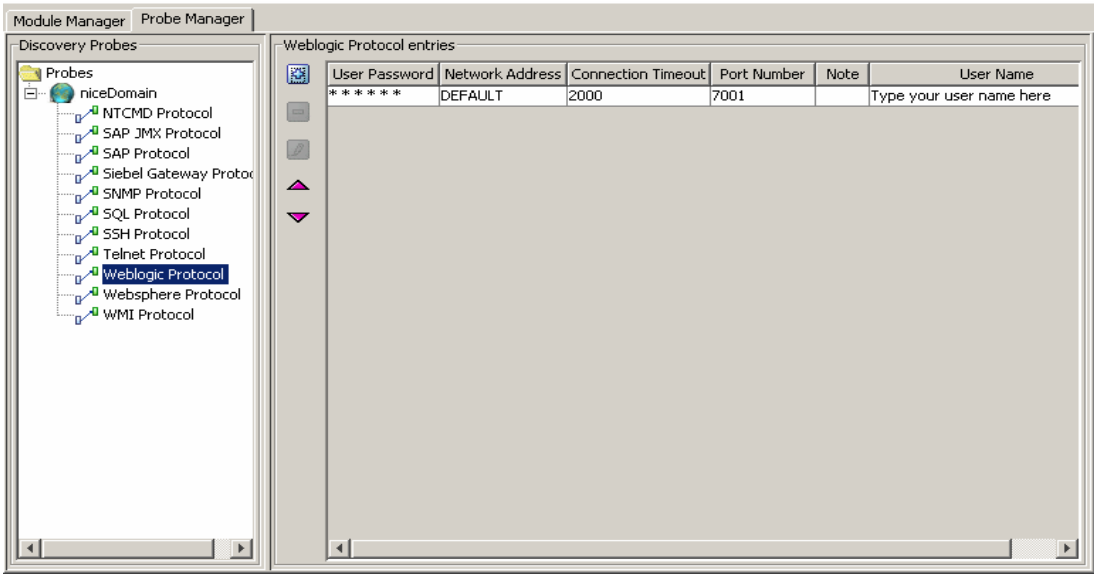
10 Click the button at the right end of the **User Password** box to open the User Password dialog box.



11 In the **New Password** box, type your password.

12 Type your password again in the **Confirm New Password** box and click **OK** to save your changes and close the User Password dialog box dialog box.

13 Click **OK** to save the protocol definitions you have set. The protocol definitions appear in the Protocol entries pane, as seen below.



14 Click **Apply** to save the changes in the CMDB.

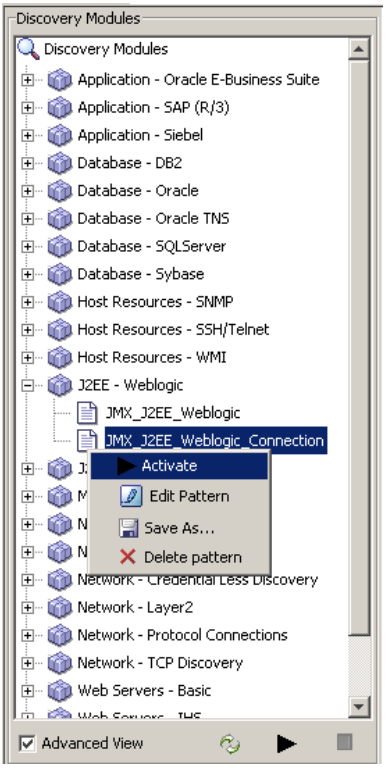
To verify that the CMDB has been updated with the changes you made in the Discovery Probe configurations, check that the following notification appears in the Discovery Probe:
Processing document domainScopeDocument.bin is done

Discovering WebLogic Instances

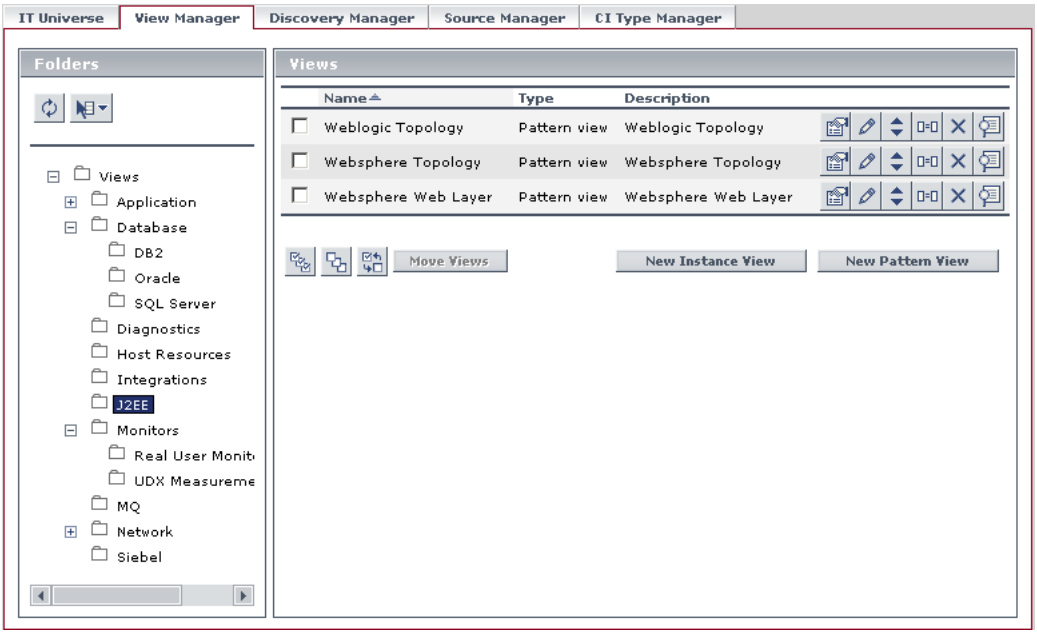
You need to activate the **JMX_J2EE_Weblogic_Connection** pattern in order to discover all the instances of WebLogic.

To discover all instances of WebLogic:

1. Click the **Module Manager** tab.
2. Select the **Advanced View** check box.
3. In the Discovery Modules pane, click the **Expand** button to the left of the **J2EE - Weblogic**module.
4. Select **JMX_J2EE_Weblogic_Connection**.
5. Right-click **JMX_J2EE_Weblogic_Connection** and click the **Activate** button, or select **JMX_J2EE_Weblogic_Connection** and click the **Activate** button in the bottom- right corner of the Discovery Modules pane.



The pattern discovers all the Weblogic instances in your system. The discovered CIs appear in a predefined view called J2EE.



The **J2EE_JMX_Weblogic_Connection** discovery pattern discovers WebLogic instances, which can be viewed in the topology map in IT Universe (for details, see *IT Universe Administration*.

Discovering WebLogic Components

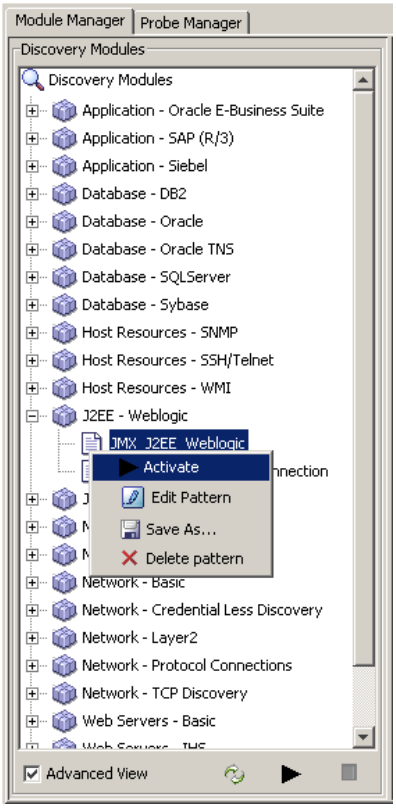
In this section, you are going to activate the pattern that discovers Weblogic components.

To discover WebLogic components:

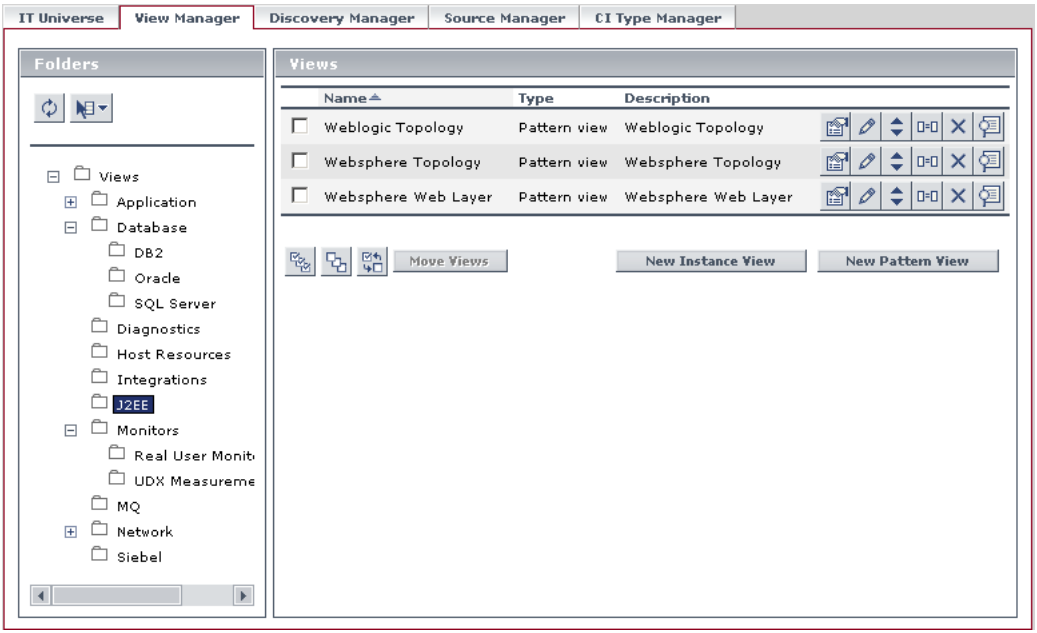
1 Click the **Module Manager** tab.

2 Click the **Expand** button to the left of the **J2EE-Weblogic** module.

3 Right-click **JMX_J2EE_Weblogic** and click the **Activate** button, or select **JMX_J2EE_Weblogic** and click the **Activate** button in the bottom- right corner of the Discovery Modules pane.



The pattern discovers all the Weblogic resources in your system.



The **J2EE_JMX_Weblogic_Connection** discovery pattern discovers Weblogic instances, which can be viewed in the topology map in IT Universe (for details, see *IT Universe Administration*.

9

Discovering Host Resources

In the previous lesson, you discovered the WebLogic instances and WebLogic components in your system.

In this lesson, you will activate a number of patterns that discover WMI-based resources, such as disks, CPU, memory, or files.

- WMI_HR_CPU_Dynamic
- WMI_HR_Disk_Dynamic
- WMI_HR_Process_Dynamic
- WMI_HR_Service_Dynamic

In this lesson you will learn about:

- "Defining the WMI Protocol" on page 64
- "Discovering WMI Components" on page 68

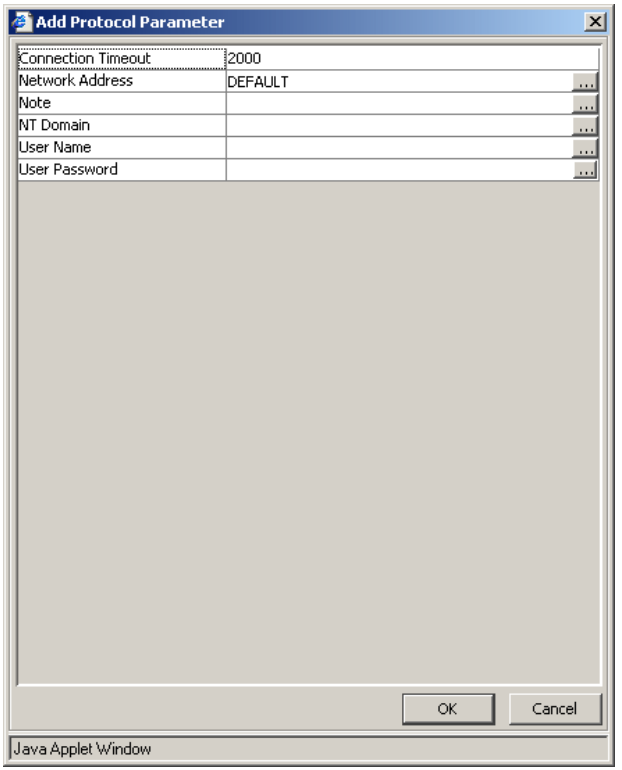
Defining the WMI Protocol

In this section, you will add the WMI protocol and define its connection data.

To define the WMI protocol:

1. Select **Admin** > **CMDB** and click the **IT Universe** tab to open the Discovery Manager.
2. Click the **Probe Manager** tab.
3. In the Discovery Probes pane, select **niceDomain**.
4. Click the **Expand** button to the left of **niceDomain** and then select **WMI Protocol**.
5. Click the **Add new connection details for the selected protocol type** button in the Protocol entries pane to open the Add Protocol Parameter dialog box.

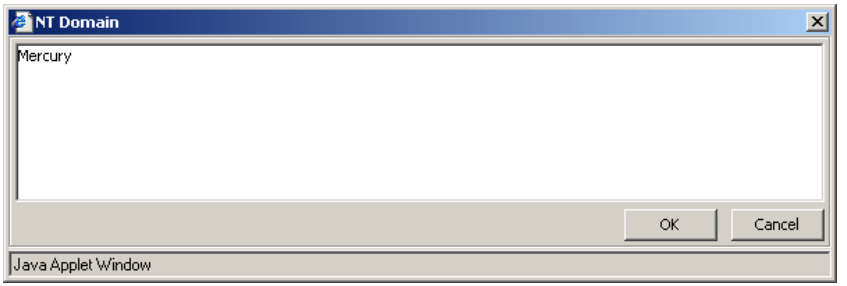
6 From the **Add Protocol** list, select **WMI Protocol**. The Add Protocol Parameter dialog box opens.



7 In the **Connection Timeout** box, leave the default as 2000.

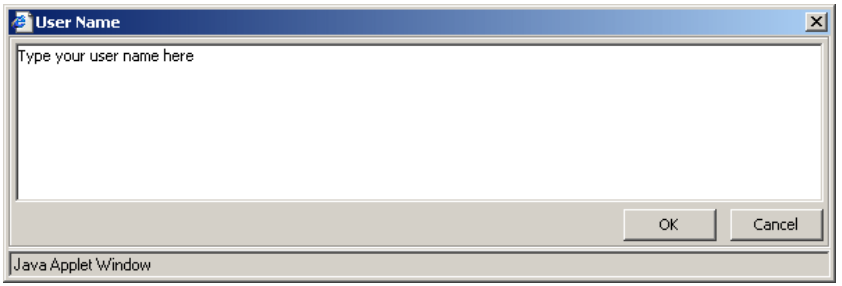
8 In the **Network Address** box, leave the default as DEFAULT.

9 Click the button at the right end of the **NT Domain** box. In the dialog box that opens, type your domain. For example, Mercury.

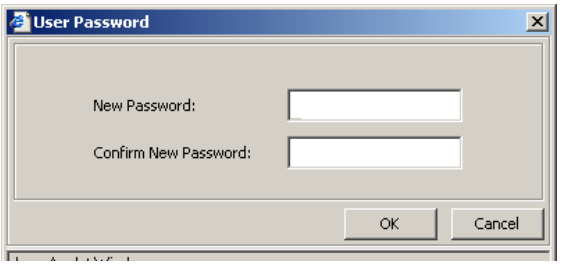


10 Click **OK**.

11 Click the button at the right end of the **User Name** box. In the dialog box that opens, type your user name and click **OK**.



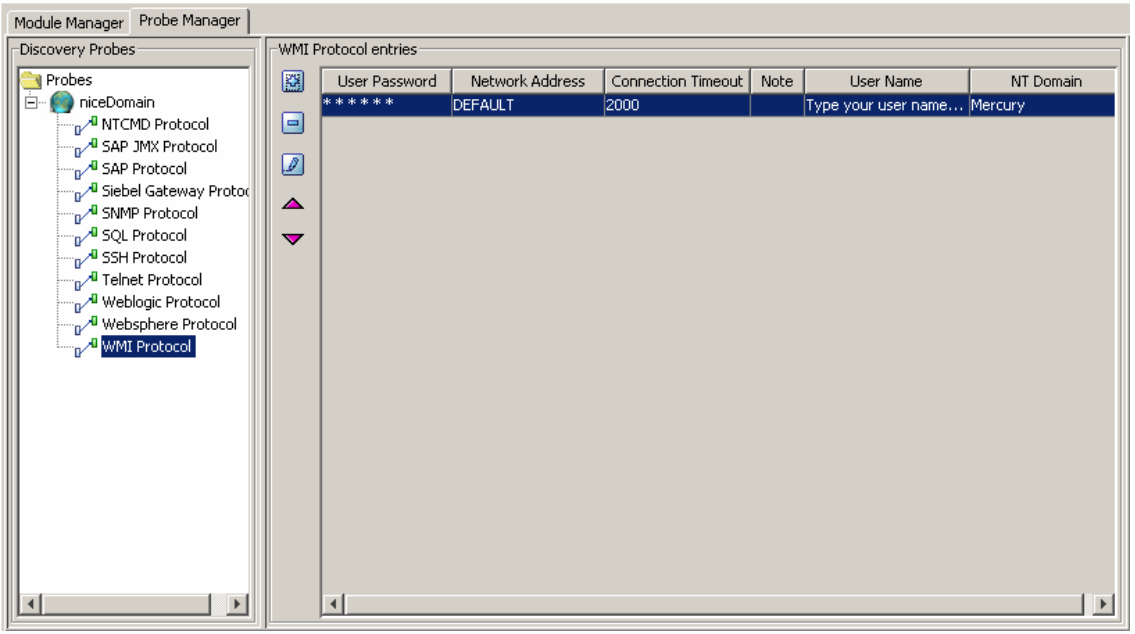
12 Click the button at the right end of the **User Password** box to open the User Password dialog box.



13 In the **New Password** box, type your password.

14 Type your password again in the **Confirm New Password** box and click **OK** to save your changes and close the User Password dialog box dialog box.

15 Click **OK** to save the protocol definitions you have set. The protocol definitions appear, as seen below.



16 Click **Apply** again to save the changes in the CMDB.

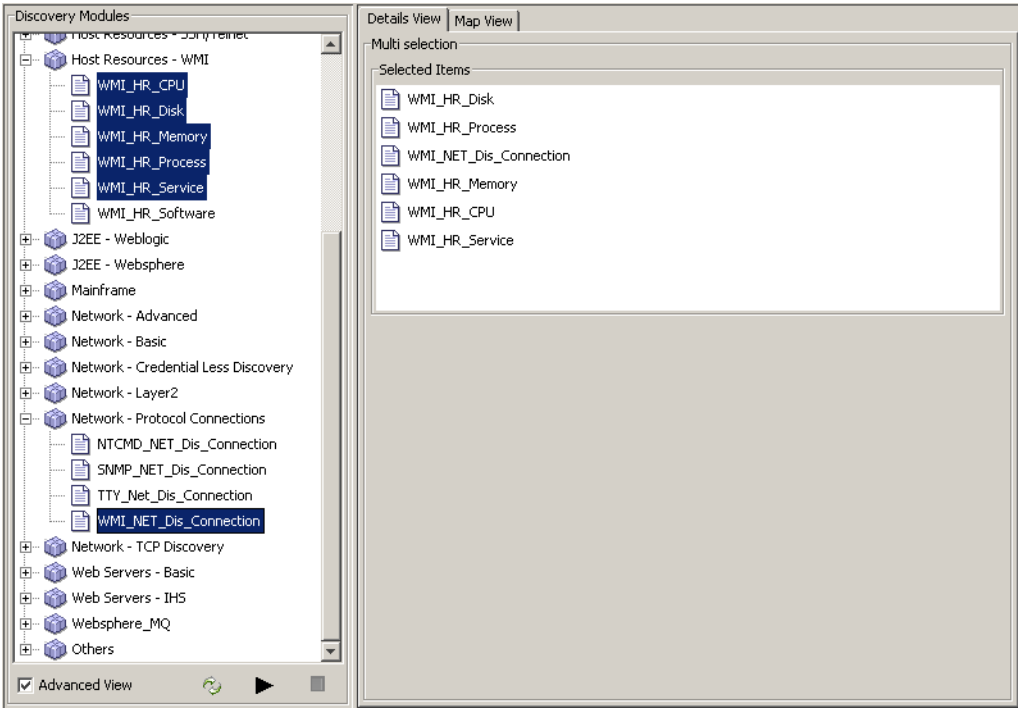
Discovering WMI Components

In this section, you are going to activate the pattern that discovers WMI-based resources.

To discover WMI components:

1. Click the **Module Manager** tab.
2. Select the **Advanced View** check box.
3. Click the **Expand** button to the left of the **Host_resources - WMI** module and select the following:
   - **WMI_HR_Dis_Memory**
   - **WMI_HR_Dis_CPU**
   - **WMI_HR_Dis_Disk**
   - **WMI_HR_Process**
   - **WMI_HR_Service**
4. Click the **Expand** button to the left of the **Network - Protocol Connections** module and select **WMI_NET_Dis_Connection**.

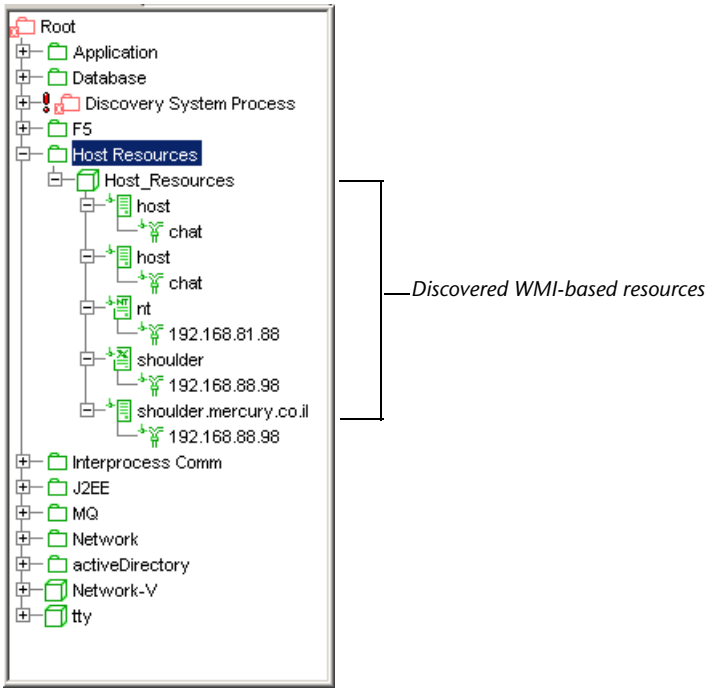
Note: You can connect several patterns simultaneously by holding down the CTRL key and selecting the required patterns, as seen below.





5 Click the **Activate** button in the bottom- right corner of the Discovery Modules pane.

The patterns uncover all the WMI-based resources, which are located under **Host Resources**.



10

Troubleshooting Pattern Activation Issues

When activating a discovery pattern, you may get an error message. You can troubleshoot the problem by checking:

- The Discovery System Process view. This view contains all the patterns that are running. Each pattern is represented by a CI in this view. If an errors occurs, an event is sent to that CI describing the error that has occurred.
- The following log files that are located in **\Mercury\<Mercury Business Availability Center root directory>\root\logs:**
  - **probemanager-error.log.** Contains all error messages that occurred in the Probe Manager.
  - **probemanager-infra.log.** Contains infrastructure messages for the Probe Manager.
  - **probemanager-services.log** Contains logs for services. This is the most relevant log file in which to look for errors, warnings, or information regarding a specific discovery pattern.
  - **probegateway.log.** Contains logs for the Probe Gateway.
  - **collectorsManagement.log**. Contains logs of all the discovery processes that occur in the Mercury Business Availability Center server.
  - **wrapperServer.log.** Contains general logs for the Mercury Business Availability Center server.
  - **world.log.** Contains logs that tell you if there were problems inserting CIs into the CMBD.

A

Optional Variables in Discovery Patterns

Mercury Business Availability Center version 2.4.FP 1 allows you to add the attribute **optional="true"** to a variable tag in a discovery pattern, as shown below.

```
<variables>
        <variable name="SNMP" getObjectBy="condition" optional="true">
                <condition>
                        <object id="-1" class="snmp" subsystem="6" container_name="host_applicationlist">
                                <attribute name="snmp_port" type="java.lang.Integer" list="false"
operator="EQ">161</attribute>
                                <attribute name="root_container" type="host" list="false" operator="EQ">
                                        <object id="${HOST.root_id}" class="host" subsystem="6" />
                                </attribute>
                        </object>
                </condition>
        </variable>
</variables>
```

Adding this attribute to the discovery pattern ensures that the discovery pattern does not fail if the specified variable is missing.

If the **optional="true"** attribute is added, every reference to the missing variable will be replaced by either:

- An empty string " " as a value
- A default value if it is provided for the attribute

Use the following syntax to define the default value. The default value appears in bold:

<destinationData name="retry">${SNMP.snmp_retry:**5**}</destinationData>

In the above example, the **optional="true"**attribute was added to the variable tag and $SNMP was not found. The retry value will be replaced by **5** since the value that is put after the **:** (colon) becomes the default value if the attribute does not exist.

Note: Discovery patterns that do not use optional variables will fail if the specified variables are not found.

B

Discovery Patterns

Application Discovery

Pattern		
_XCMD_Invoke_InstallAgent.xml	**Package**	Network
	Parse Method	Java
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Relationships	
	Monitored Objects/Relationships	
	TQL	"NO PQL
	Schedule	"once
	Notes	invokes an application, not in use

Pattern		
IHS_Dis_Admin_Connection	**Package**	J2EE
	Parse Method	Jython
	Protocol	Telnet
	Discovered Objects/Relationships	httpcontext,
	Monitored Objects/Relationships	
	TQL	host_ihs_telnet
	Schedule	
	Notes	
IHS_NET_Dis_Connection	**Package**	J2EE
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Relationships	ibmhttpserver
	Monitored Objects/Relationships	
	TQL	host
	Schedule	
	Notes	

Pattern		
jboss_components.xml	**Package**	J2EE
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Relationships	jmsdestination, jmsserver, ejbcomponent, webapplication, servlet, connectionpool, j2eecluster
	Monitored Objects/Relationships	
	TQL	jboss
	Schedule	1800
	Notes	
jboss_connection.xml	**Package**	J2EE
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Relationships	jboss
	Monitored Objects/Relationships	
	TQL	host
	Schedule	Once
	Notes	

Pattern		
JMX_APP_DIS_WEBSPHERE.xml	**Package**	J2EE
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Relationships	j2eeserver, jmsprovider, jdbcprovider, jdbcdatasource, j2eeapplication, ejbmodule, webapplication, servlet, ejbstateless, ejbstateful, ejbentity ejbmessagedriven,
	Monitored Objects/Relationships	
	TQL	websphere
	Schedule	1800
	Notes	
JMX_APP_DIS_WEBSPHERE_connection.xml	**Package**	J2EE
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Relationships	websphere
	Monitored Objects/Relationships	
	TQL	host
	Schedule	Once
	Notes	

Pattern		
JMX_APP_Dis_WL.xml	**Package**	J2EE
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Relationships	j2eeserver, jmsprovider, jdbcprovider, jdbcdatasource, j2eeapplication, ejbmodule, webapplication, servlet, ejbstateless, ejbstateful, ejbentity ejbmessagedriven,
	Monitored Objects/Relationships	
	TQL	weblogic
	Schedule	86400
	Notes	
JMX_APP_Dis_WL_Connection.xml	**Package**	J2EE
	Parse Method	Jython
	Protocol	Java
	Discovered Objects/Relationships	weblogic
	Monitored Objects/Relationships	
	TQL	host
	Schedule	86400
	Notes	

Pattern		
JMX_J2EE_Weblogic.xml	**Package**	J2EE-JSR77
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Relationships	j2eeserver, jmsprovider, jdbcprovider, jdbcdatasource, j2eeapplication, ejbmodule, webapplication, servlet, ejbstateless, ejbstateful, ejbentity ejbmessagedriven,
	Monitored Objects/Relationships	
	TQL	weblogic
	Schedule	86400
	Notes	
JMX_J2EE_Weblogic_Connection.xml	**Package**	J2EE-JSR77
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Relationships	weblogic
	Monitored Objects/Relationships	
	TQL	host
	Schedule	86400
	Notes	

Pattern		
JMX_J2EE_Weblogic61.xml	**Package**	J2EE-JSR77
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Relationships	j2eeserver, jmsprovider, jdbcprovider, jdbcdatasource, j2eeapplication, ejbmodule, webapplication, servlet, ejbstateless, ejbstateful, ejbentity ejbmessagedriven,
	Monitored Objects/Relationships	
	TQL	weblogic
	Schedule	86400
	Notes	
JMX_J2EE_Weblogic61_Connection.xml	**Package**	J2EE-JSR77
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Relationships	weblogic
	Monitored Objects/Relationships	
	TQL	host
	Schedule	86400
	Notes	

Pattern		
NTCMD_MQ_Topology.xml	**Package**	Websphere_MQ
	Parse Method	Jython
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Relationships	mqqueuemanager, mqcluster, mqrepository, mqxmitq, mqqueuelocal, mqqueueremote, mqaliasq, mqqueue, mqalias, mqchsdr, mqchsvr, mqchannel, mqchannelof, mqchrqstr, mqchclntconn, mqchclusrcvr, mqchclussdr, webspheremq
	Monitored Objects/Relationships	
	TQL	ntcmd
	Schedule	86400
	Notes	it uses mq utils

Pattern		
SAP_Dis_Site.xml	**Package**	SAP
	Parse Method	Jython
	Protocol	BAPI
	Discovered Objects/Relationships	sapserver, sapsite, sapservice, sapsupportpackage, sapcomponent,
	Monitored Objects/Relationships	
	TQL	host
	Schedule	
	Notes	discovers all entries in the CCMS
SIEBEL_DIS_APP_SERVERS.xml	**Package**	Siebel
	Parse Method	Jython
	Protocol	Siebel protocol
	Discovered Objects/Relationships	siebelappserver, siebelcompgrp, siebelcomponent, siebelgateway,
	Monitored Objects/Relationships	
	TQL	siebel_gtwy_connected
	Schedule	
	Notes	SIEBEL Discovery through srvrmgr application provider by Siebel

Pattern		
SIEBEL_DIS_DB_NT.xml	**Package**	Siebel
	Parse Method	Jython
	Protocol	WMI
	Discovered Objects/Relationships	database, dbconnector, siebelappserver,
	Monitored Objects/Relationships	
	TQL	siebel_appserver_nt
	Schedule	
	Notes	
SIEBEL_DIS_DB_UNIX.xml	**Package**	Siebel
	Parse Method	Jython
	Protocol	Telnet
	Discovered Objects/Relationships	database, siebelappserver
	Monitored Objects/Relationships	
	TQL	siebel_appserver_unix
	Schedule	
	Notes	

Pattern		
SIEBEL_DIS_DB_UNIX_SSH.xml	**Package**	Siebel
	Parse Method	Jython
	Protocol	SSH
	Discovered Objects/Relationships	database, siebelappserver
	Monitored Objects/Relationships	
	TQL	siebel_appserver_unix_ssh
	Schedule	
	Notes	
SIEBEL_DIS_GATEWAY_CONNECTION_(GTWY).xml	**Package**	Siebel
	Parse Method	Jython
	Protocol	Siebel protocol
	Discovered Objects/Links	siebelgateway, siebelsite,
	Monitored Objects/Links	
	TQL	siebel_gtwy
	Schedule	
	Notes	SIEBEL Discovery through srvrmgr application provider by Siebel

Pattern		
SIEBEL_DIS_GATEWAY_CONNECTION_(GTWY_HOST).xml	**Package**	Siebel
	Parse Method	Jython
	Protocol	Siebel protocol
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	siebel_gtwy_host
	Schedule	
	Notes	SIEBEL Discovery through srvrmgr application provider by Siebel
SIEBEL_DIS_WEBAPPS_NT.xml	**Package**	Siebel
	Parse Method	Jython
	Protocol	NTCMD
	Discovered Objects/Links	siebelwse, siebelgateway, siebelsite, siebelwebapp, webserver
	Monitored Objects/Links	
	TQL	siebel_webserver_nt
	Schedule	
	Notes	

Pattern		
SIEBEL_DIS_WEBAPPS_UNIX.xml	**Package**	Siebel
	Parse Method	Jython
	Protocol	Telnet
	Discovered Objects/Links	siebelwse, siebelgateway, siebelsite, siebelwebapp, webserver
	Monitored Objects/Links	
	TQL	siebel_webserver_unix
	Schedule	
	Notes	
SIEBEL_DIS_WEBAPPS_UNIX_SSH.xml	**Package**	Siebel
	Parse Method	Jython
	Protocol	SSH
	Discovered Objects/Links	siebelwse, siebelgateway, siebelsite, siebelwebapp, webserver
	Monitored Objects/Links	
	TQL	siebel_webserver_unix
	Schedule	
	Notes	

Pattern		
SNMP_NET_Mon_CitrixServer.xml	**Package**	Citrix
	Parse Method	Jython
	Protocol	SNMP
	Discovered Objects/Links	citrixserver, citrixfarm
	Monitored Objects/Links	
	TQL	citrix_port_on_snmp_host
	Schedule	Once
	Notes	
SNMP_NET_Mon_CitrixSession.xml	**Package**	Citrix
	Parse Method	Jython
	Protocol	SNMP
	Discovered Objects/Links	citrixsession, citrixclient
	Monitored Objects/Links	
	TQL	host_with_citrixserver
	Schedule	Once
	Notes	
TCP_PORT_EXPECTED.xml	**Package**	J2EE
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Links	application
	Monitored Objects/Links	
	TQL	http_ports
	Schedule	86400
	Notes	should create specific webserver, eg. IIS, Tomcat, Apache etc.

Pattern		
TELNET_MQ_Topology.xml	**Package**	Websphere_MQ
	Parse Method	Jython
	Protocol	Telnet
	Discovered Objects/Links	mqqueuemanager, mqcluster, mqrepository, mqxmitq, mqqueuelocal, mqqueueremote, mqaliasq, mqqueue, mqalias, mqchsdr, mqchsvr, mqchannel, mqchannelof, mqchrqstr, mqchclntconn, mqchclusrcvr, mqchclussdr, webspheremq
	Monitored Objects/Links	
	TQL	telnet
	Schedule	86400
	Notes	it uses mq utils
WMI_APP_Lis_IIS_Down.xml	**Package**	IIS_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	iis
	TQL	iis_server
	Schedule	Once
	Notes	destination oriented DCOM listener, pulling

Pattern		
WMI_APP_Lis_IIS_Up.xml	**Package**	IIS_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	iis
	TQL	iis_server
	Schedule	Once
	Notes	destination oriented DCOM listener, pulling
WMI_APP_Mon_IIS.xml	**Package**	IIS_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	iis
	TQL	wmi
	Schedule	30
	Notes	
WMI_Exchange_Dis_Server.xml	**Package**	Exchange_Resources_By_ WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	exchangeserver
	Monitored Objects/Links	
	TQL	wmi
	Schedule	Once
	Notes	

Pattern		
WMI_Exchange_Dis_Server_Resources.xml	**Package**	Exchange_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	exchangesite, exchangeroutinggroup, exchangeconnector, exchangelink, exchangequeue
	Monitored Objects/Links	
	TQL	wmi
	Schedule	600
	Notes	
WMI_Exchange_Lis_ServerState_Changes.xml	**Package**	Exchange_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	exchangeserver
	TQL	"NO PQL
	Schedule	"Once
	Notes	

BMC Patrol Integration

Pattern		
PATROL_HR_Lis.xml	**Package**	Host_Resources_By_Patrol
	Parse Method	Dynamic
	Protocol	TCP
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	patrol_agent
	Schedule	Once (Listener)
	Notes	it listens to patrol events

Database

Pattern		
SQL_APP_Dis_Db2.xml	**Package**	DB2
	Parse Method	Dynamic
	Protocol	JDBC
	Discovered Objects/Links	db2user
	Monitored Objects/Links	
	TQL	db2
	Schedule	1800
	Notes	

Pattern		
SQL_APP_Dis_Oracle.xml	**Package**	Oracle
	Parse Method	Dynamic
	Protocol	JDBC
	Discovered Objects/Links	dbtablespace, dbuser, owner, program
	Monitored Objects/Links	
	TQL	oracle_database
	Schedule	1800
	Notes	
SQL_APP_Dis_Sqldatabase.xml	**Package**	SQL_Server
	Parse Method	Dynamic
	Protocol	JDBC
	Discovered Objects/Links	sqlfile, disk
	Monitored Objects/Links	
	TQL	sql_server_sqldatabase
	Schedule	86400
	Notes	

Pattern		
SQL_APP_Dis_SqlServer.xml	**Package**	SQL_Server
	Parse Method	Dynamic
	Protocol	JDBC
	Discovered Objects/Links	sqldatabase, sqlbackup, sqlalert, sqljob, sqljobstep, sqlperformancemonitor, sqlprocesses, program, dbclient
	Monitored Objects/Links	
	TQL	sql_server
	Schedule	1800
	Notes	
SQL_APP_Dis_Sybase.xml	**Package**	Sybase
	Parse Method	Dynamic
	Protocol	JDBC
	Discovered Objects/Links	sybasedb
	Monitored Objects/Links	
	TQL	sybase
	Schedule	1800
	Notes	

Pattern		
SQL_NET_Dis_Connection.xml	**Package**	Database_Basic
	Parse Method	Java
	Protocol	JDBC
	Discovered Objects/Links	oracle, db2, sybase, sqlserver
	Monitored Objects/Links	
	TQL	db_ports
	Schedule	604800
	Notes	

FTP

Pattern		
FTP_NET_Dis_Connection.xml	**Package**	FTP
	Parse Method	Java
	Protocol	FTP
	Discovered Objects/Links	ftp
	Monitored Objects/Links	
	TQL	ftp_ports
	Schedule	Once
	Notes	
FTP_NET_Dis_Files.xml	**Package**	FTP
	Parse Method	Dynamic
	Protocol	FTP
	Discovered Objects/Links	
	Monitored Objects/Links	file
	TQL	"NO PQL
	Schedule	"300
	Notes	
FTP_NET_Mon.xml	**Package**	FTP
	Parse Method	Dynamic
	Protocol	FTP
	Discovered Objects/Links	
	Monitored Objects/Links	file
	TQL	host_ftp_username
	Schedule	1800
	Notes	

Host Resource

Pattern		
BB_HR_Lis_Agents_Heartbeat_Dynamic.xml	**Package**	Host_Resources_By_Big_Brother
	Parse Method	Java
	Protocol	TCP
	Discovered Objects/Links	
	Monitored Objects/Links	bbagent
	TQL	new_probe
	Schedule	Once (Listener)
	Notes	read a buffer that was sent via TCP from BB agent
BB_HR_Lis_CPU_Dynamic.xml	**Package**	Host_Resources_By_Big_Brother
	Parse Method	Dynamic
	Protocol	TCP
	Discovered Objects/Links	bbagent, cpu, memory, program
	Monitored Objects/Links	cpu, memory, program
	TQL	new_probe
	Schedule	Once (Listener)
	Notes	read a buffer that was sent via TCP from BB agent

Pattern		
BB_HR_Lis_Disks_Dynamic.xml	**Package**	Host_Resources_By_Big_Brother
	Parse Method	Dynamic
	Protocol	TCP
	Discovered Objects/Links	bbagent, disk
	Monitored Objects/Links	disk
	TQL	new_probe
	Schedule	Once (Listener)
	Notes	read a buffer that was sent via TCP from BB agent
BB_HR_Lis_NTEventLog.xml	**Package**	Host_Resources_By_Big_Brother
	Parse Method	Dynamic
	Protocol	TCP
	Discovered Objects/Links	bbagent, nt, eventlog
	Monitored Objects/Links	eventlog
	TQL	new_probe
	Schedule	Once (Listener)
	Notes	read a buffer that was sent via TCP from BB agent

Pattern		
BB_HR_Lis_Processes_Dynamic.xml	**Package**	Host_Resources_By_Big_Brother
	Parse Method	Dynamic
	Protocol	TCP
	Discovered Objects/Links	bbagent, program
	Monitored Objects/Links	program
	TQL	new_probe
	Schedule	Once (Listener)
	Notes	read a buffer that was sent via TCP from BB agent
BB_HR_Lis_Services_Dynamic.xml	**Package**	Host_Resources_By_Big_Brother
	Parse Method	Dynamic
	Protocol	TCP
	Discovered Objects/Links	bbagent, service
	Monitored Objects/Links	bbagent, service
	TQL	new_probe
	Schedule	Once (Listener)
	Notes	read a buffer that was sent via TCP from BB agent

Pattern		
NTCMD_HR_Dis_Nt_Dynamic.xml	**Package**	Host_Resources_By_NTCMD
	Parse Method	Dynamic
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	nt
	Monitored Objects/Links	
	TQL	ntcmd_data
	Schedule	Once
	Notes	psinfo.exe
NTCMD_HR_Dis_Service_Dynamic.xml	**Package**	Host_Resources_By_NTCMD
	Parse Method	Dynamic
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	service
	Monitored Objects/Links	service
	TQL	ntcmd_data
	Schedule	Once
	Notes	psservice.exe

Pattern		
NTCMD_HR_Dis_ServiceSpecific.xml	**Package**	Host_Resources_By_NTCMD
	Parse Method	Dynamic
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	service
	Monitored Objects/Links	
	TQL	new_service
	Schedule	Once
	Notes	psservice.exe
NTCMD_HR_Dis_Software_Dynamic.xml	**Package**	Host_Resources_By_NTCMD
	Parse Method	Dynamic
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	softwares
	Monitored Objects/Links	
	TQL	ntcmd_data
	Schedule	Once
	Notes	psinfo.exe -s

Pattern		
NTCMD_HR_Mon_Disks_Dynamic.xml	**Package**	Host_Resources_By_NTCMD
	Parse Method	Dynamic
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	disk
	Monitored Objects/Links	disk
	TQL	ntcmd_data
	Schedule	300
	Notes	psinfo,exe -d
NTCMD_HR_Mon_Process_Dynamic.xml	**Package**	Host_Resources_By_NTCMD
	Parse Method	Dynamic
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	program
	Monitored Objects/Links	program
	TQL	ntcmd_data
	Schedule	300
	Notes	pslist.exe

Pattern		
NTCMD_HR_Mon_Service_Dynamic.xml	**Package**	Host_Resources_By_NTCMD
	Parse Method	Dynamic
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	service
	Monitored Objects/Links	service
	TQL	ntcmd_data
	Schedule	300
	Notes	psservice.exe
SNMP_HR_Mon_Disks_Dynamic.xml	**Package**	Host_Resources_By_SNMP
	Parse Method	Dynamic
	Protocol	SNMP
	Discovered Objects/Links	disk
	Monitored Objects/Links	disk
	TQL	snmp_agent_of_a_server
	Schedule	60 seconds
	Notes	start oid: 1.3.6.1.2.1.25.2.3.1.3

Pattern		
SNMP_HR_Mon_Printq_Dynamic.xml	**Package**	Host_Resources_By_SNMP
	Parse Method	Dynamic
	Protocol	SNMP
	Discovered Objects/Links	printqs
	Monitored Objects/Links	printqs
	TQL	snmp_agent_of_a_server
	Schedule	60 seconds
	Notes	start oid: 1.3.6.1.4.1.77.1.2.29.1.2
SNMP_HR_Mon_Process_Dynamic.xml	**Package**	Host_Resources_By_SNMP
	Parse Method	Dynamic
	Protocol	SNMP
	Discovered Objects/Links	program
	Monitored Objects/Links	program
	TQL	snmp_agent_of_a_server
	Schedule	60 seconds
	Notes	start oid: 1.3.6.1.2.1.25.4.2.1.1

Pattern		
SNMP_HR_Mon_Service_Dynamic.xml	**Package**	Host_Resources_By_SNMP
	Parse Method	Dynamic
	Protocol	SNMP
	Discovered Objects/Links	service
	Monitored Objects/Links	service
	TQL	snmp_agent_of_a_server
	Schedule	60 seconds
	Notes	start oid: 1.3.6.1.4.1.77.1.2.3.1.1
SNMP_HR_Mon_Software_Dynamic.xml	**Package**	Host_Resources_By_SNMP
	Parse Method	Dynamic
	Protocol	SNMP
	Discovered Objects/Links	software
	Monitored Objects/Links	software
	TQL	snmp_agent_of_a_server
	Schedule	60 seconds
	Notes	start oid: 1.3.6.1.2.1.25.6.3.1.2

Pattern		
SNMP_NET_Dis_Users.xml	**Package**	Host_Resources_By_SNMP
	Parse Method	Dynamic
	Protocol	SNMP
	Discovered Objects/Links	osuser
	Monitored Objects/Links	
	TQL	snmp_agent_port_161
	Schedule	Once
	Notes	start oid: 1.3.6.1.4.1.77.1.2.25.1.1
TELNET_HR_Dis_Disks.xml	**Package**	Host_Resources_By_Telnet
	Parse Method	Dynamic
	Protocol	Telnet
	Discovered Objects/Links	disk
	Monitored Objects/Links	disk
	TQL	telnet
	Schedule	300
	Notes	df -k
TELNET_HR_Dis_HpDaemons.xml	**Package**	Host_Resources_By_Telnet
	Parse Method	Dynamic
	Protocol	Telnet
	Discovered Objects/Links	daemon
	Monitored Objects/Links	
	TQL	telnet_of_hp_device
	Schedule	Once
	Notes	ps -ef | grep '?'

Pattern		
TELNET_HR_Dis_HpSoftware.xml	**Package**	Host_Resources_By_Telnet
	Parse Method	Dynamic
	Protocol	Telnet
	Discovered Objects/Links	software
	Monitored Objects/Links	
	TQL	telnet_of_hp_device
	Schedule	Once
	Notes	swlist
TELNET_HR_Dis_Processes.xml	**Package**	Host_Resources_By_Telnet
	Parse Method	Dynamic
	Protocol	Telnet
	Discovered Objects/Links	program
	Monitored Objects/Links	program
	TQL	telnet
	Schedule	300
	Notes	ps -ef
TELNET_HR_Dis_SunDaemons.xml	**Package**	Host_Resources_By_Telnet
	Parse Method	Dynamic
	Protocol	Telnet
	Discovered Objects/Links	daemon
	Monitored Objects/Links	
	TQL	telnet_of_sun_device
	Schedule	Once
	Notes	ps -ef

Pattern		
TELNET_HR_Dis_SunSoftware.xml	**Package**	Host_Resources_By_Telnet
	Parse Method	Dynamic
	Protocol	Telnet
	Discovered Objects/Links	software
	Monitored Objects/Links	
	TQL	telnet_of_sun_device_no_snmp
	Schedule	Once
	Notes	pkginfo
WMI_HR_Dis_CPU_Dynamic.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	cpu
	Monitored Objects/Links	
	TQL	wmi
	Schedule	Once
	Notes	
WMI_HR_Dis_Disk_Dynamic.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	disk
	Monitored Objects/Links	
	TQL	wmi
	Schedule	Once
	Notes	

Pattern		
WMI_HR_Dis_Memory_Dynamic.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	memory
	Monitored Objects/Links	
	TQL	wmi
	Schedule	Once
	Notes	
WMI_HR_Dis_Process_Dynamic.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	program
	Monitored Objects/Links	
	TQL	wmi
	Schedule	Once
	Notes	
WMI_HR_Dis_Service_Dynamic.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	service
	Monitored Objects/Links	
	TQL	wmi
	Schedule	Once
	Notes	

Pattern		
WMI_HR_Lis_CPUBusy.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	cpu
	TQL	wmi
	Schedule	Once (Listener)
	Notes	destination oriented DCOM listener, pulling
WMI_HR_Lis_CPUNormal.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	cpu
	TQL	wmi
	Schedule	Once (Listener)
	Notes	destination oriented DCOM listener, pulling

Pattern		
WMI_HR_Lis_MemoryLow.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	memory
	TQL	wmi
	Schedule	Once (Listener)
	Notes	destination oriented DCOM listener, pulling
WMI_HR_Lis_MemoryNormal.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	memory
	TQL	wmi
	Schedule	Once (Listener)
	Notes	destination oriented DCOM listener, pulling

Pattern		
WMI_HR_Lis_NTEventLog.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	eventlog
	TQL	wmi
	Schedule	Once (Listener)
	Notes	destination oriented DCOM listener, pulling
WMI_HR_Lis_Process_Started.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	program
	TQL	wmi
	Schedule	Once (Listener)
	Notes	destination oriented DCOM listener, pulling

Pattern		
WMI_HR_Lis_Process_Stopped.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	program
	TQL	wmi
	Schedule	Once (Listener)
	Notes	destination oriented DCOM listener, pulling
WMI_HR_Lis_Service_Started.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	service
	TQL	wmi
	Schedule	Once (Listener)
	Notes	destination oriented DCOM listener, pulling

Pattern		
WMI_HR_Lis_Service_Stopped.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	service
	TQL	wmi
	Schedule	Once (Listener)
	Notes	destination oriented DCOM listener, pulling
WMI_HR_Mon_Disks_Dynamic.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	disk
	TQL	wmi
	Schedule	600
	Notes	
WMI_HR_Mon_Processes_Dynamic.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	program
	TQL	wmi
	Schedule	600
	Notes	

Pattern		
WMI_HR_Mon_Services_Dynamic.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	service
	TQL	wmi
	Schedule	30
	Notes	

Internal

Pattern		
SynchronizeByIp.xml	**Package**	Network
	Parse Method	Jython
	Protocol	
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	complete_host
	Schedule	once
	Notes	synchronizes hosts in the system

Invoke Operation

Pattern		
_NTCMD_HR_Invoke_StartService.xml	**Package**	Host_Resources_By_NTCMD
	Parse Method	Dynamic
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	
	Monitored Objects/Links	service
	TQL	"NO PQL
	Schedule	"Once
	Notes	dispatched on a service, 'psservice.exe start'

Pattern		
_NTCMD_HR_Invoke_StopProgram.xml	**Package**	Host_Resources_By_NTCMD
	Parse Method	Dynamic
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	"NO PQL
	Schedule	"Once
	Notes	dispatched on a program, 'pskill.exe'
_NTCMD_HR_Invoke_StopService.xml	**Package**	Host_Resources_By_NTCMD
	Parse Method	Dynamic
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	
	Monitored Objects/Links	service
	TQL	"NO PQL
	Schedule	"Once
	Notes	dispatched on a service, 'psservice.exe stop'

Pattern		
WMI_Invoke_Create_Process.xml	**Package**	Host_Resources_By_WMI
	Parse Method	N/A
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	"NO PQL
	Schedule	"Once
	Notes	dispatched on a specific object
WMI_Invoke_Start_Service.xml	**Package**	Host_Resources_By_WMI
	Parse Method	N/A
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	"NO PQL
	Schedule	"Once
	Notes	dispatched on a specific object

Pattern		
WMI_Invoke_Stop_Service.xml	**Package**	Host_Resources_By_WMI
	Parse Method	N/A
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	"NO PQL
	Schedule	"Once
	Notes	dispatched on a specific object
WMI_Invoke_Terminate_Process.xml	**Package**	Host_Resources_By_WMI
	Parse Method	N/A
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	"NO PQL
	Schedule	"Once
	Notes	dispatched on a specific object

Pattern		
WMI_REG_Invoke_Get_Key.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	wmi
	Schedule	Once
	Notes	skeleton for retriving data from registry
WMI_REG_Invoke_Get_Values.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	wmi
	Schedule	Once
	Notes	skeleton for retriving data from registry

Pattern		
WMI_REG_Invoke_GetBBValues.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	bbagent
	Monitored Objects/Links	
	TQL	wmi
	Schedule	Once
	Notes	discover Big Brother agent + configuration
WMI_REG_Invoke_GetInstalledSoftware.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	software
	Monitored Objects/Links	
	TQL	wmi
	Schedule	Once
	Notes	like "Control Panel" -> "Add Remove Program"

Pattern		
WMI_REG_Invoke_Set_Value.xml	**Package**	Host_Resources_By_WMI
	Parse Method	N/A
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	wmi
	Schedule	Once
	Notes	skeleton for setting data to registry
WMI_REG_Invoke_SetBBValues.xml	**Package**	Host_Resources_By_WMI
	Parse Method	N/A
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	"NO PQL
	Schedule	"Once
	Notes	configure Big Brother agent

Layer 2

Pattern		
SNMP_Dis_L2_Bridge.xml	**Package**	Layer2
	Parse Method	Jython
	Protocol	SNMP
	Discovered Objects/Links	concentrator, port
	Monitored Objects/Links	
	TQL	bridge_no_vlan
	Schedule	86400
	Notes	layer 2 picture is completed w/ Server Logic rules which create layertwo & backbone links
SNMP_Dis_L2_Vlan.xml	**Package**	Layer2
	Parse Method	Jython
	Protocol	SNMP
	Discovered Objects/Links	concentrator, port
	Monitored Objects/Links	
	TQL	catalyst_vlan
	Schedule	86400
	Notes	layer 2 picture is completed w/ Server Logic rules which create layertwo & backbone links

LDAP

Pattern		
ldapPingTest.xml	**Package**	LDAP
	Parse Method	Jython
	Protocol	LDAP
	Discovered Objects/Links	activedirectory
	Monitored Objects/Links	
	TQL	host
	Schedule	
	Notes	

Network

Pattern		
CMD_NET_Mon_DnsServer.xml	**Package**	Network
	Parse Method	Dynamic
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	dnsserver_ports
	Schedule	1800
	Notes	uses nslookup to resolve dnsnames, not in use

Pattern		
DNS_NET_Dis_DnsName.xml	**Package**	Network
	Parse Method	Java
	Protocol	
	Discovered Objects/Links	host_dnsname
	Monitored Objects/Links	
	TQL	known_ip_no_DNS
	Schedule	once
	Notes	uses java InetAddress class
Host_ID_Discover_Python.xml	**Package**	Network
	Parse Method	Jython
	Protocol	
	Discovered Objects/Links	host, ip
	Monitored Objects/Links	
	TQL	new_ip_not_unknown
	Schedule	86400
	Notes	tries all 4 major protocols: snmp, telnet, xcmd & wmi
ICMP_NET_Dis_IpB.xml	**Package**	Network
	Parse Method	Java
	Protocol	ICMP
	Discovered Objects/Links	ip
	Monitored Objects/Links	
	TQL	network_B_not_unknown
	Schedule	60
	Notes	uses IcmpPingUtilDll.dll

Pattern		
ICMP_NET_Dis_IpC.xml	**Package**	Network
	Parse Method	Java
	Protocol	ICMP
	Discovered Objects/Links	ip
	Monitored Objects/Links	
	TQL	network_C_not_unknown
	Schedule	86400
	Notes	uses IcmpPingUtilDll.dll
ICMP_NET_Mon_Ip.xml	**Package**	Network
	Parse Method	Java
	Protocol	ICMP
	Discovered Objects/Links	
	Monitored Objects/Links	ip
	TQL	new_ip_not_unknown
	Schedule	600
	Notes	uses IcmpPingUtilDll.dll
MS_NET_Dis_Domain.xml	**Package**	Network
	Parse Method	Java
	Protocol	WIN API
	Discovered Objects/Links	msdomain, nt
	Monitored Objects/Links	
	TQL	new_local
	Schedule	3600
	Notes	uses win api via dll to fetch list of MS domain/workgroup

Pattern		
NTCMD_NET_Dis_Connection.xml	**Package**	Network
	Parse Method	Java
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	host, ip, interface, ntcmd, network
	Monitored Objects/Links	
	TQL	host
	Schedule	Once
	Notes	
NTCMD_NET_Dis_TCP.xml	**Package**	Network
	Parse Method	Java
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	iport, clientserver (link)
	Monitored Objects/Links	
	TQL	ntcmd
	Schedule	once
	Notes	
NTCMD_NET_Mon_ClientServer.xml	**Package**	Network
	Parse Method	
	Protocol	XCMD (NetBIOS)
	Discovered Objects/Links	
	Monitored Objects/Links	iport, clientserver (link)
	TQL	ntcmd_data
	Schedule	300
	Notes	

Pattern		
SNMP_NET_Dis_Arp.xml	**Package**	Network
	Parse Method	Java
	Protocol	SNMP
	Discovered Objects/Links	host, ip, interface, network
	Monitored Objects/Links	
	TQL	snmp_agent_port_161
	Schedule	86400
	Notes	
SNMP_NET_Dis_Bridge.xml	**Package**	Network
	Parse Method	Java
	Protocol	SNMP
	Discovered Objects/Links	bridge, port, layertwo, backbone
	Monitored Objects/Links	
	TQL	bridge_of_non_enterasys_switch
	Schedule	Once
	Notes	

Pattern		
SNMP_NET_Dis_Connection.xml	**Package**	Network
	Parse Method	Java
	Protocol	SNMP
	Discovered Objects/Links	host, ip, interface, snmp, network
	Monitored Objects/Links	
	TQL	new_ip_not_unknown
	Schedule	86400
	Notes	update the correct host class (nt, unix, router etc.) according to OID
SNMP_NET_Dis_HostBase.xml	**Package**	Network
	Parse Method	Java
	Protocol	SNMP
	Discovered Objects/Links	host, ip, interface, network, rout, bridge
	Monitored Objects/Links	
	TQL	snmp_agent_port_161
	Schedule	86400
	Notes	

Pattern		
SNMP_NET_Dis_Router_Arp.xml	**Package**	Network
	Parse Method	Java
	Protocol	SNMP
	Discovered Objects/Links	host, ip, interface, network
	Monitored Objects/Links	
	TQL	snmp_agent_of_a_router
	Schedule	86400
	Notes	
SNMP_NET_Dis_TCP.xml	**Package**	Network
	Parse Method	Java
	Protocol	SNMP
	Discovered Objects/Links	ipserver, ipclient, tcp, clientserver
	Monitored Objects/Links	
	TQL	snmp_agent_of_a_server
	Schedule	86400
	Notes	
SSH_Net_Dis_Connection_jython.xml	**Package**	Network
	Parse Method	Jython
	Protocol	SSH
	Discovered Objects/Links	host, ip, interface, telnet, network
	Monitored Objects/Links	
	TQL	ip_possibly_ssh
	Schedule	Once
	Notes	

Pattern		
TCP_NET_Dis_Port.xml	**Package**	Network
	Parse Method	Java
	Protocol	TCP
	Discovered Objects/Links	ipserver, use
	Monitored Objects/Links	
	TQL	host
	Schedule	86400
	Notes	try to open Java socket
TCP_NET_Mon_Port.xml	**Package**	Network
	Parse Method	Java
	Protocol	TCP
	Discovered Objects/Links	
	Monitored Objects/Links	ipserver, use
	TQL	port_of_a_host_in_domain
	Schedule	3600
	Notes	try to open Java socket
TELNET_NET_Dis_Connection.xml	**Package**	Network
	Parse Method	Java
	Protocol	Telnet
	Discovered Objects/Links	host, ip, interface, telnet, network
	Monitored Objects/Links	
	TQL	telnet_port
	Schedule	Once
	Notes	

Pattern		
Telnet_Net_Dis_Connection_jython.xml	**Package**	Network
	Parse Method	Jython
	Protocol	Telnet
	Discovered Objects/Links	host, ip, interface, telnet, network
	Monitored Objects/Links	
	TQL	ip_possibly_telnet
	Schedule	Once
	Notes	
WMI_NET_Dis_Connection.xml	**Package**	Network
	Parse Method	Java
	Protocol	WMI (DCOM)
	Discovered Objects/Links	host, ip, interface, wmi, network
	Monitored Objects/Links	
	TQL	host
	Schedule	Once
	Notes	

Performance Monitor

Pattern		
JMS_PM_DESTINATION.xml	**Package**	J2EE
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Links	
	Monitored Objects/Links	jmsdestination
	TQL	"NO PQL
	Schedule	"20
	Notes	
JMX_PM_CONNECTIONPOOL.xml	**Package**	J2EE
	Parse Method	Jython
	Protocol	JMX
	Discovered Objects/Links	
	Monitored Objects/Links	connectionpool
	TQL	"NO PQL
	Schedule	"20
	Notes	

Pattern		
WMI_PM_Mon_CPU.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	cpu
	TQL	"NO PQL
	Schedule	"20
	Notes	dispatched on a specific object
WMI_PM_Mon_Disk.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	disk
	TQL	"NO PQL
	Schedule	"20
	Notes	dispatched on a specific object

Pattern		
WMI_PM_Mon_IIS.xml	**Package**	IIS_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	iis
	TQL	"NO PQL
	Schedule	"20
	Notes	dispatched on a specific object
WMI_PM_Mon_Memory.xml	**Package**	Host_Resources_By_WMI
	Parse Method	Dynamic
	Protocol	WMI (DCOM)
	Discovered Objects/Links	
	Monitored Objects/Links	memory
	TQL	"NO PQL
	Schedule	"20
	Notes	dispatched on a specific object

Process Discovery

Pattern		
P2P	**Package**	P2P
	Parse Method	Jython
	Protocol	TCP
	Discovered Objects/Links	application (p2p agent)
	Monitored Objects/Links	
	TQL	new_ip_not_unknown
	Schedule	Once
	Notes	
P2P_LIS_TCP_CONNECTIONS.xml	**Package**	P2P
	Parse Method	Java
	Protocol	P2P Agent
	Discovered Objects/Links	staticprocess, ptp (link)
	Monitored Objects/Links	
	TQL	new_probe
	Schedule	Once (Listener)
	Notes	P2P Agent is a proprietary agent which sends/receives data over tcp

Pattern		
P2P_PM_STATIC_LINK.xml	**Package**	P2P
	Parse Method	Java
	Protocol	P2P Agent
	Discovered Objects/Links	staticprocess, ptp (link)
	Monitored Objects/Links	
	TQL	"NO PQL
	Schedule	"10
	Notes	
P2P_PM_STATIC_PROCESS.xml	**Package**	P2P
	Parse Method	Java
	Protocol	P2P Agent
	Discovered Objects/Links	staticprocess, ptp (link)
	Monitored Objects/Links	
	TQL	"NO PQL
	Schedule	"10
	Notes	

SNMP TRAP

Pattern		
TRAP_NET_Lis_Generic.xml	**Package**	Network
	Parse Method	Dynamic
	Protocol	
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	new_probe
	Schedule	Once (Listener)
	Notes	listens for snmp trap
TRAP_NET_Lis_Interface.xml	**Package**	Network
	Parse Method	Dynamic
	Protocol	
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	new_probe
	Schedule	Once (Listener)
	Notes	listens for snmp trape

TCP

Pattern		
NTCMD_NET_Dis_TCP_Jython.xml	**Package**	TCP_Discovery
	Parse Method	Jython
	Protocol	NTCMD
	Discovered Objects/Links	clientserver & tcp links for ipserver/ipclient
	Monitored Objects/Links	
	TQL	ip_of_ntcmd
	Schedule	
TELNET_NET_Dis_TCP_Jython.xml	**Package**	TCP_Discovery
	Parse Method	Jython
	Protocol	Telnet
	Discovered Objects/Links	clientserver & tcp links for ipserver/ipclient
	Monitored Objects/Links	
	TQL	ip_of_telnet
	Schedule	
SSH_NET_Dis_TCP_Jython.xml	**Package**	TCP_Discovery
	Parse Method	Jython
	Protocol	SSH
	Discovered Objects/Links	clientserver & tcp links for ipserver /ipclient
	Monitored Objects/Links	
	TQL	ip_of_ssh
	Schedule	

NTCMD_NET_Dis_TCP_Jython.xml

Pattern		
TCP_Discovery	**Parse Method**	Jython
	Protocols	NTCMD
	Discovered Objects/Links	clientserver & tcp links for ipserver /ipclient
	Monitored Objects/Links	
	TQL	ip_of_ntcmd
	Schedule	
	Notes	

TELNET_NET_Dis_TCP_Jython.xml

Pattern		
TCP_Discovery	**Parse Method**	Jython
	Protocols	Telnet
	Discovered Objects/Links	clientserver & tcp links for ipserver /ipclient
	Monitored Objects/Links	
	TQL	ip_of_telnet
	Schedule	
	Notes	

SSH_NET_Dis_TCP_Jython.xml

Pattern		
NTCMD_NET_Dis_TCP_Jython.xml	**Parse Method**	TCP_Discovery
	Protocols	Jython
	Discovered Objects/Links	NTCMD
	Monitored Objects/Links	clientserver & tcp links for ipserver /ipclient
	TQL	
	Schedule	ip_of_ntcmd
	Notes	
TELNET_NET_Dis_TCP_Jython.xml	**Parse Method**	TCP_Discovery
	Protocols	Jython
	Discovered Objects/Links	Telnet
	Monitored Objects/Links	clientserver & tcp links for ipserver /ipclient
	TQL	
	Schedule	ip_of_telnet
	Notes	

Pattern		
SSH_NET_Dis_TCP_Jython.xml	**Parse Method**	TCP_Discovery
	Protocols	Jython
	Discovered Objects/Links	SSH
	Monitored Objects/Links	clientserver & tcp links for ipserver /ipclient
	TQL	
	Schedule	ip_of_ssh
	Notes	

XSL Parser

Pattern		
TCP_LIS_XML_DATA.xml	**Parse Method**	Network
	Protocols	Java
	Discovered Objects/Links	
	Monitored Objects/Links	
	TQL	new_probe
	Schedule	ip_of_ntcmd
	Notes	uses xsl engine

Security Privileges for Patterns

Patterns	Requires	Creates (DARM object)[a]	Data Source
SNMP_NET_Dis_Connection.xml	SNMP Community	snmp	the various tables of a MIB
TELNET_NET_Dis_Connection.xml	Telnet User/Password	telnet	any valid telnet command, subjected to user privileges
WMI_NET_Dis_Connection.xml	WMI User/Password	wmi	any WMI valid query
NTCMD_NET_Dis_Connection.xml	Administrator Windows User/Password	ntcmd	any valid DOS command
SQL_NET_Dis_Connection.xml	Database User/Password	sqlserver, oracle, db2, sybase	any valid SQL statement, subjected to user privileges
FTP_NET_Dis_Connection.xml	FTP User/Password	ftp	any valid ftp command
jboss_connection.xml	JMX Mbean Server User/Password	jboss	any valid JMX ObjectName
JMX_APP_DIS_WEBSPHERE_ connection.xml	JMX Mbean Server User/Password	websphere	any valid JMX ObjectName
JMX_APP_Dis_WL_Connection.xml	JMX Mbean Server User/ Password	weblogic	any valid JMX ObjectName

a. Every pattern that is based on one of these objects, uses the credentials which are encapsulated in that object (e.g. snmp_community)

Specific Pattern Data Source

Pattern	Data Source (table name)
SQL_APP_Dis_Oracle.xml	DBA_DATA_FILES, DBA_DB_LINKS, DBA_JOBS, DBA_OBJECTS, DBA_SNAPSHOTS, DBA_TABLESPACES, DBA_USERS, V$BACKUP, v$CONTROLFILE, v$database, V$DATAFILE, VLOG, VLOGFILE, v$parameter, V$RECOVER_FILE, V$SESSION
SQL_APP_Dis_Db2.xml	IWH.USERFUNCTION
SQL_APP_Dis_Sqldatabase.xml	sysfiles
SQL_APP_Dis_SqlServer.xml	sysprocesses, sysdatabases, backupset, sysalerts, sysjobs, syslogins, sysjobhistory, sysjobschedules, sysperfinfo
SQL_APP_Dis_Sybase.xml	sysdatabases