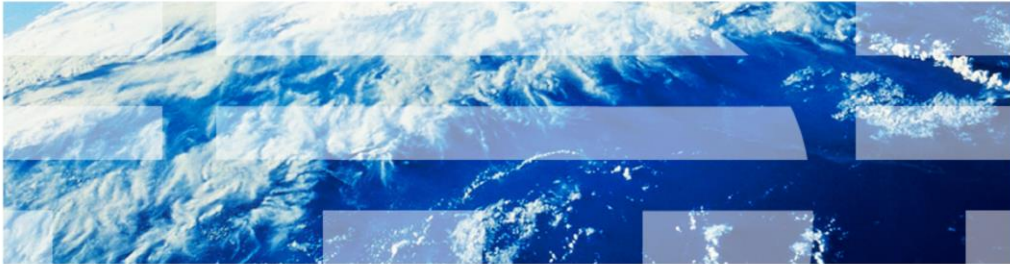


IBM Tivoli Application Dependency Discovery Manager 7.2.1

Anchors



© 2013 IBM Corporation

This presentation covers anchors in Tivoli® Application Dependency Discovery Manager version 7.2.1

Objectives

After you complete this module, you can perform these tasks:

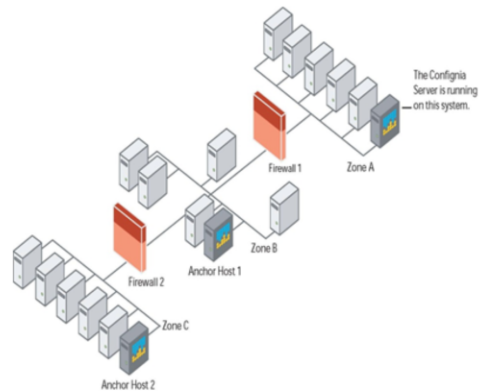
- Describe how anchors work
- Diagnose common errors

After you complete this module, you can describe how Tivoli Application Dependency Discovery Manager anchors work and diagnose common errors.

Tivoli Application Dependency Discovery Manager anchors

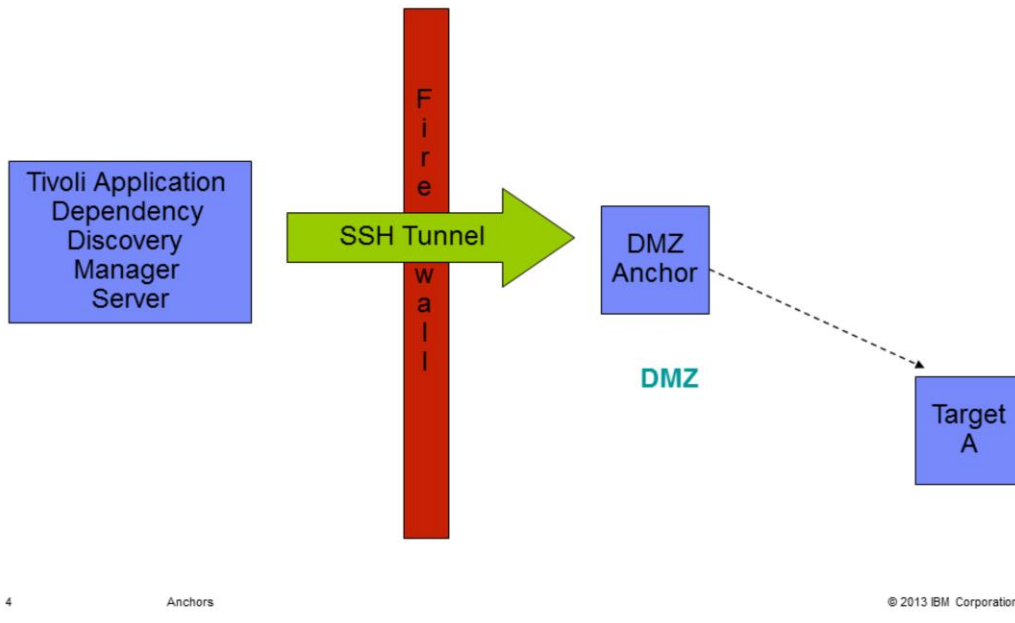
With anchors, you can run discovery across firewall zones

- Many companies restrict access to areas of their network using firewalls
- Tivoli Application Dependency Discovery Manager can use a server that is located within the firewall zone to perform discoveries on behalf of the primary root server
- If there are Windows servers within the firewall zone, a Windows Gateway is also required



Many companies use firewalls to restrict access to parts of their network, making discovery in those areas difficult. Tivoli Application Dependency Discovery Manager anchors allow for discovery across firewall zones. From an anchor in the firewall zone, Tivoli Application Dependency Discovery Manager can perform discoveries on behalf of the primary discovery management server, called the root server. If Windows servers are within the firewall zone, then you also need a Windows gateway in that zone.

Communication (1 of 2)

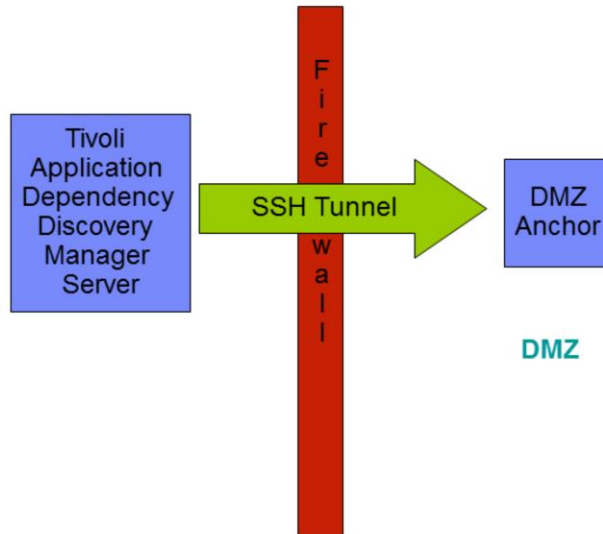


The root server is itself an anchor. It uses sockets to communicate to the remote anchor through an SSH tunnel.

A socket address is the combination of an IP address and a port into a single identity, much like one end of a telephone connection is the combination of a telephone number and a particular extension.

Communication (2 of 2)

- *Only* Port 22, the default sshPort, must be open
- All communication between the root server and the anchor happens in the SSH tunnel



5

Anchors

© 2013 IBM Corporation

Only Port 22, or whatever has been defined as the ssh port, must be open on the firewall between the root server and the anchor. All communication between the root server and the anchor is done in the SSH tunnel.

Anchor requirements

- Anchors have the same hardware and software requirements as Tivoli Application Dependency Discovery Manager root servers
- These root servers are also anchors

When you remember that the root server is itself an anchor, it is easy to remember that anchors have all the same hardware and software requirements as Tivoli Application Dependency Discovery Manager servers.

Anchors are defined by the Data Management Console

The screenshot displays the 'Anchors and Gateways' panel in the Data Management Console. A table lists existing anchors and gateways. An 'Add Anchor' dialog box is open, allowing for the configuration of a new anchor. The dialog includes a dropdown for 'Type' (set to 'Anchor'), radio buttons for 'Set By' (selected 'Address'), and an IP address field (9.42.37.75). It also features a 'Scope to search for host' section with radio buttons for 'Entire scope' and 'Limit to selected scope' (selected). A list of scopes is shown below, with 'DMZ' selected.

Type	Address	Port	Scope Set
Anchor	root server	8497	
Windows Gateway			
Windows Gateway			

Add Anchor

Type:

Set By: Address Host Name

Address: . . .

Scope to search for host

Entire scope Limit to selected scope

DMZ 9.42.42.97 (rwb1220)

DMZ anchor

WAS

In the data management console, navigate to the Anchors and Gateways panel to define the anchor and, if applicable, its scope restriction.

Files to reference (server side) (1 of 2)

dist/etc/anchor.properties

Example of contents:

```
anchor_host_1=9.42.37.75
anchor_scope_1=DMZ
port=8497
anchor_scope_0=DMZ anchor
anchor_host_0=root server
```

Note: the *anchor_scope_0* and *anchor_host_0* lines do not exist if the root server is not scope restricted

The `anchor.properties` file, which is located in the **dist/etc** subdirectory under the main Tivoli Application Dependency Discovery Manager directory, contains the defined anchors and their scope restrictions, if those exist. This file is automatically updated when you modify the Anchors and Gateways panel. You do not have to manually edit this file unless you use anchors for Network Address Translation (or NAT) environments. In the case of NAT subnets, you must assign the anchors to a zone using the anchor zone property. See the *Tivoli Application Dependency Discovery Manager User's Guide* for further instructions regarding NAT zones.

Files to reference (server side) (2 of 2)

dist/etc/scope.properties

Example contents:

```
#DO NOT EDIT THIS FILE; IT WILL BE OVERWRITTEN BY THE DATABASE
#Wed Feb 10 15:33:35 EST 2010
DMZ=9.42.42.97
RTP\ Range=9.42.37.1-9.42.37.128
WAS=9.42.18.108
DMZ\ anchor=9.42.37.75
RTP\ Subnet=9.42.36.0/255.255.255.128
test=9.42.12.235
NSJ=9.43.75.0/255.255.255.0, 9.43.74.0/255.255.255.0, 9.43.73.0/255.255.255.0
```

Do not modify the dist/etc/scope.properties file manually, because it updates automatically when you modify scopes on the data management console. It is a useful file, however, for diagnosing all anchor problems. If you report a discovery issue involving anchors to IBM Level 2 Support, provide the scope.properties file with the anchor.properties file.

Files and directories (anchor side)

The anchor directory is:

- \$HOME/coll7.1 (Tivoli Application Dependency Discovery Manager 7.1 or 7.1.2.0)
- \$HOME/coll7.1.2.1 (Tivoli Application Dependency Discovery Manager 7.1.2 + any fix pack)
- \$HOME/coll7.2 (Tivoli Application Dependency Discovery Manager 7.2)
- \$HOME/taddm7.2.1.x (Tivoli Application Dependency Discovery Manager 7.2.1 FPx)

Additionally, the property `com.ibm.cdb.taddm.anchor.root` was added in 721 to allow for customizing this location

- \$HOME/taddm7.2.2.x (Tivoli Application Dependency Discovery Manager 7.2.2 FPx)

Where \$HOME is one of these locations:

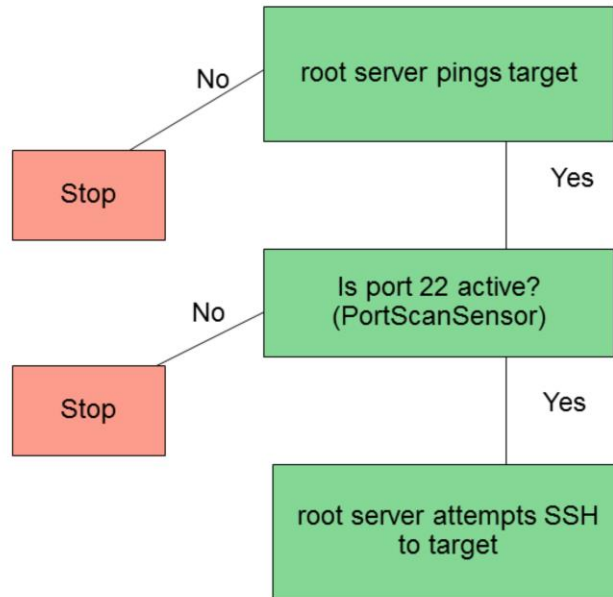
- Directory of the user ID that is used to discover the anchor (UNIX)
- %SystemRoot%\temp (Windows)

\$anchor_dir/etc/collation.properties is copied by the root server

sshd_config must be readable by the user running the anchor process and must contain have AllowTcpForwarding set to Yes

The root server copies all files that the anchor requires to run a discovery as part of the initial anchor deployment. These files are copied to the anchor directory on the anchor. The exact location of the anchor directory depends upon the platform (UNIX versus Windows), the user's home directory, and the version of Tivoli Application Dependency Discovery Manager. The directory structure below the anchor home directory will always be the same.

Discovery, first step, no anchor



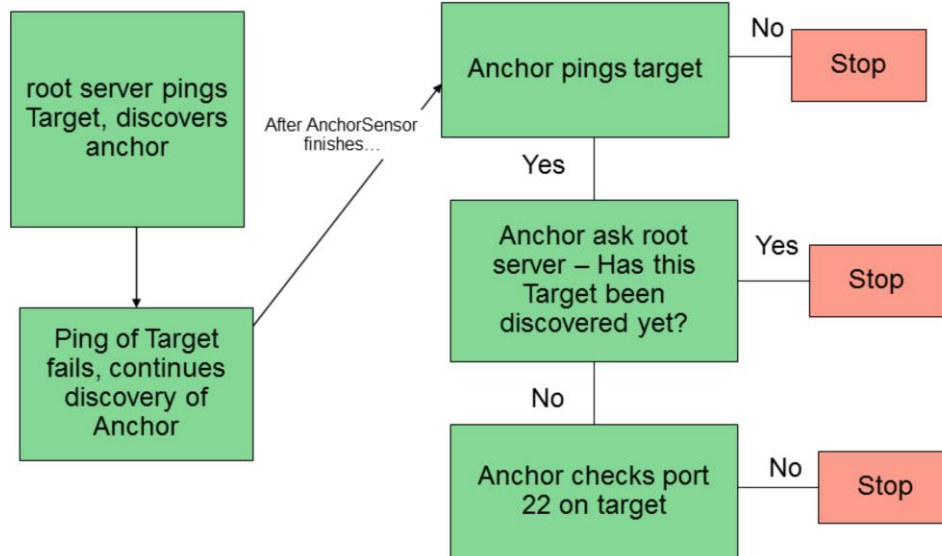
11

Anchors

© 2013 IBM Corporation

When root server runs a discovery, it first tries to ping the target IP to determine if it is an active IP. If there is no response, discovery proceeds no further. If the IP responds successfully to a ping, the root server then scans the SSH port. If the SSH port is active, the root server then attempts to discover the target. If the SSH port is not active, the root server attempts to discover the target using SNMP, but in this demonstration, for the sake of simplicity, you are not concerned with the SNMP part of discovery.

Discovery with an anchor, standard firewall between the root server and target, no scope restrictions



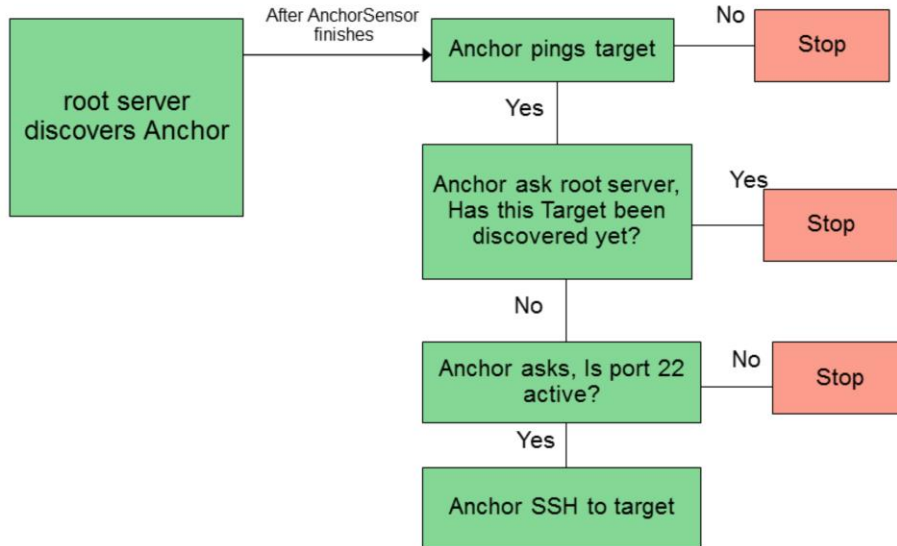
12

Anchors

© 2013 IBM Corporation

If an anchor is used for discovery and there are no scope restrictions, the root server still attempts to ping the target. After the anchor sensor completes, the anchor also attempts to ping the target. The anchor attempts further discovery if the target responds *and* the target has not already been discovered. Tivoli Application Dependency Discovery Manager keeps information that indicates whether a target has been discovered internally, but basically if the PortSensor ran successfully on the target from the root server or another anchor, then this anchor does not run discovery.

Discovery with an anchor with scope restrictions



13

Anchors

© 2013 IBM Corporation

When the root server is restricted to the anchor, it does not run the ping sensor on the target. Only the anchor runs the ping sensor against the target.

Sockets that are used locally by anchor process

- 8497 - used by remote anchor process
 - When the AnchorSensor starts the anchor Java process, that process listens on this socket
 - When discovery ends, the root server sends an `end` message to the anchor, the Java process stops, and the socket closes
 - You can configure this socket from the Tivoli Application Dependency Discovery Manager Product Console
 - Same socket is used by *all* remote anchors

- 8495 - used by local anchor process
 - When a *local* anchor is required, such as when doing WebSphere® discovery, a local Java process (`jvm`) starts and listens on 127.0.0.1 on Socket 8495
 - If an extra sensor also requires a local anchor, the next `jvm` process starts, and this new process listens on Socket 8494 (8495-1)
 - You cannot configure this socket

The remote anchor process uses Socket 8497. When the Java process starts on the anchor by the AnchorSensor, that Java process listens on Socket 8497. When discovery ends, three events occur:

The root server sends an end message to the anchor.

The Java process stops.

The socket closes.

You configure this socket, which all remote anchors use, from the Tivoli Application Dependency Discovery Manager Product Console.

The local anchor process uses Socket 8495. For example, a local anchor is necessary when you run a WebSphere discovery. In such a case, a local Java process starts and listens from 127.0.0.1 on Socket 8495. If an additional sensor also requires a local anchor, the next Java process starts. This new process listens on Socket 8494 (8495 - 1). This socket cannot be configured.

Configuring socket that remote anchors use

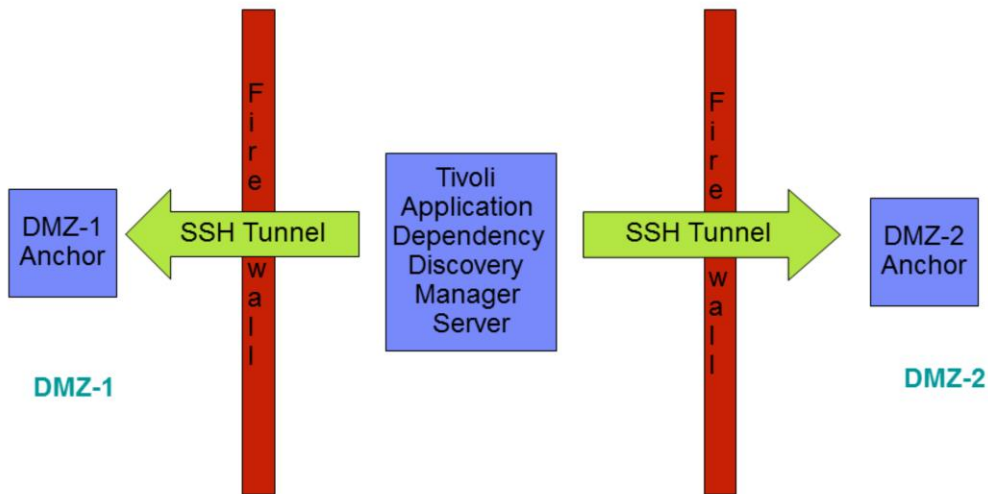
The screenshot displays the 'Anchors and Gateways' panel in the Tivoli Application Dependency Discovery Manager console. The panel contains a table with the following data:

Type	Address	Port
Anchor	root server	8499
Windows Gateway		----
Windows Gateway		----
Windows Gateway		----
Anchor		8499

An 'Edit Port Number' dialog box is open, showing the 'Port No.' field with the value '8499'. The 'Set Anchor Port' button at the bottom right of the panel is highlighted with a black arrow.

You configure the socket that remote anchors use from the Tivoli Application Dependency Discovery Manager Product Console Anchors and Gateways panel.

Configuring anchors with multiple firewalls



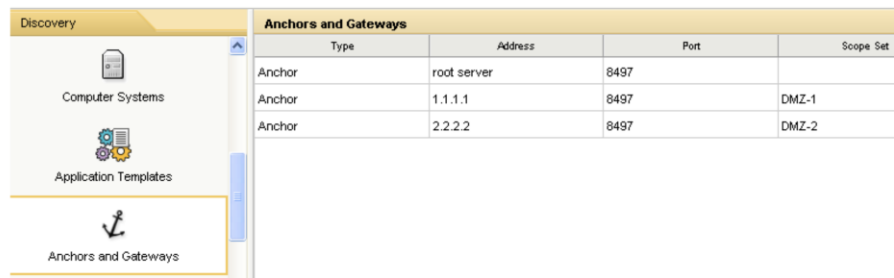
16

Anchors

© 2013 IBM Corporation

If there is only one firewall between the root server and the target, an anchor is placed beyond each firewall. Each anchor is responsible for discovery within its network zone. Only the SSH port must be open from root server to each anchor.

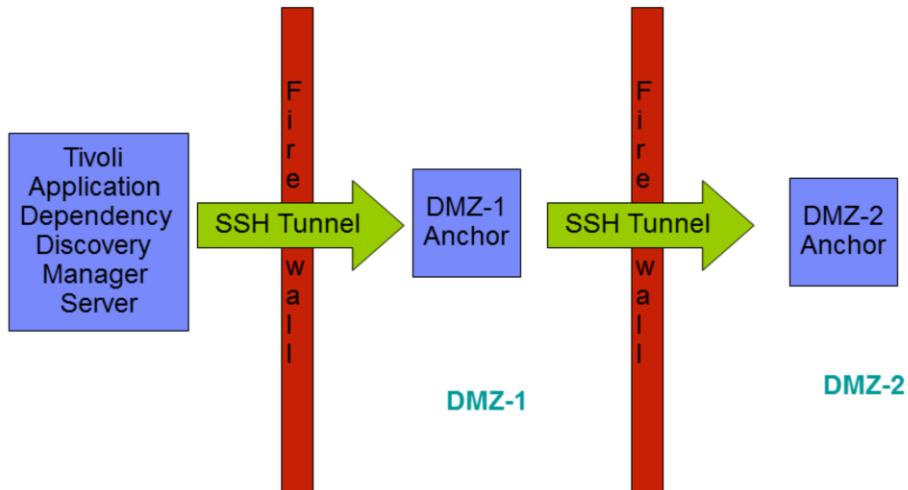
Multiple anchors



Type	Address	Port	Scope Set
Anchor	root server	8497	
Anchor	1.1.1.1	8497	DMZ-1
Anchor	2.2.2.2	8497	DMZ-2

To define the anchors for the previous example, Anchor IP 1.1.1.1 can run discoveries of all IPs in the scope DMZ-1, and Anchor IP 2.2.2.2 can run discoveries of all IPs in the scope DMZ-2.

Chaining anchors with multiple firewalls



18

Anchors

© 2013 IBM Corporation

When there are multiple firewalls between the root server and the target, an anchor is placed within each firewalled network zone. Only the SSH port must be open on the firewall between each anchor (including the root server) and the next anchor.

In this example with multiple firewalls, the root server can access only Anchor 1. Anchor 1 in turn accesses Anchor 2. Only the SSH port must be open from the root server to Anchor1 and from Anchor1 to Anchor2.

Chained anchors (1 of 2)

Scope				
Scope Sets	Method	Type	Value	
a_set	Include	Host	10.1.1.1	
anchors	Include	Host	10.2.1.1	
DMZ-1				
DMZ-2				

Scope				
Scope Sets	Method	Type	Value	
a_set	Include	Host	10.1.1.10	
anchors				
DMZ-1	Include	Host	10.1.1.11	
DMZ-2	Include	Host	10.1.1.12	

Scope				
Scope Sets	Method	Type	Value	
a_set	Include	Host	10.2.1.10	
anchors				
DMZ-1	Include	Host	10.2.1.11	
DMZ-2	Include	Host	10.2.1.12	

As an example of how to define chained anchors, first define the scopes for the anchors, DMZ-1 and DMZ-2.

Chained anchors (2 of 2)

Anchors and Gateways				
Type	Address	Port	Scope Set	
Anchor	root server	8497	anchors	
Anchor	10.1.1.1	8497	DMZ-1	
Anchor	10.2.1.1	8497	DMZ-2	

Next, you restrict the scope of each anchor to its appropriate DMZ. In this example, the root server is restricted to discover only the anchors themselves. That setup is not always necessary, but it prevents the root server from attempting to discover any targets.

Troubleshooting

For troubleshooting purposes only, you can compress and copy the **\$COLLATION_HOME/dist/support/bin** directory to **\$anchor_dir/support/bin**

Scripts that do not have to access the Tivoli Application Dependency Discovery Manager database work on the anchor

Example: **testwasconnection** and **testsshport** work, but **testssh** does not

Though the scripts under dist/support/bin directory are not officially supported, you can copy them to the anchor, in attempts to diagnose the problem. Scripts that do not require access to the Tivoli Application Dependency Discovery Manager database work on the anchor.

Troubleshooting: Sample logs from Anchor

- anchor.<FQDN>.log<.#>
 - Example:
anchor.nwimpro.tivlab.raleigh.ibm.com.log.1
 - Main anchor log, similar to DiscoverObserver.log on the root server
For example: grep PortScan anchor*
 - Eye catcher at the start of any anchor or the local-anchor log:
INFO anchor.CollationServer –
- Collation.Server.<#>.<FQDN>.out
 - Example CollationServer.01102010.1.30.07M52.nwimpro.tivlab.raleigh.ibm.com.out
 - Can contain some error information but often empty
 - If the error is the anchor itself, this log might help

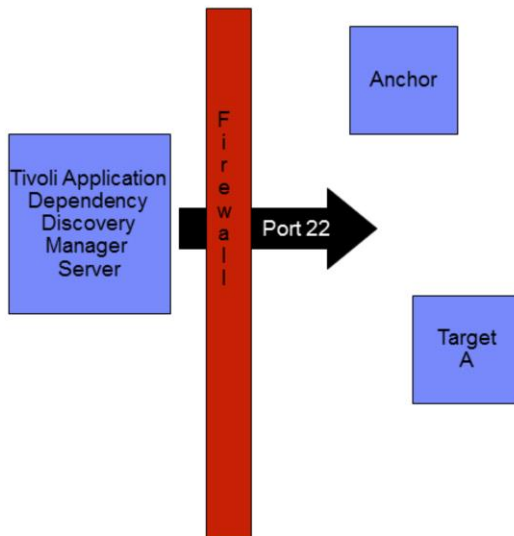
Each new discovery in main log on the anchor starts with the string “INFO anchor.CollationServer –.” The Collation.Server files on the anchor are often empty, but sometimes you can use them to diagnose a communication problem on the anchor itself.

Troubleshooting: Example of sensor logs

- The PortScanSensor log contains messages similar to these examples:
Port Ping : 10.1.2.1:[445,1741,135,23,22,389,53,1521]
Ping failed for IP address 10.1.2.1 on port 445
Ping failed for IP address 10.1.2.1 on port 1741
- If the failed ports are listed separately, at least one port pinged successfully
- If *no* ports pinged successfully, this message opens:
Ping failed for IP address 10.1.2.1 on all ports [22, 135]
- The stated messages show in the sensor logs on the server that actually performed the discovery attempt, either the root server or the anchor

Examination of the port scan sensor logs can be helpful in determining which server performed the discovery. They can show failures or, more importantly, unexpected success, from the wrong root server or anchor, which can explain discovery results.

Error scenario: SSH port is open from root server to all targets



- This situation causes a problem when root server attempts to discover applications, such as WebSphere, that use extra ports
- Telnet `<ip> <port>` is useful for troubleshooting this scenario

24

Anchors

© 2013 IBM Corporation

Having the SSH port open from the root server to all targets beyond a firewall, that is, not just the anchor, causes a problem if the root server then attempts to discover applications, such as WebSphere, that needs extra ports to communicate.

Telnet is useful in troubleshooting connectivity.

What discovery through an anchor looks like in the logs

Look at the PortScan Sensor on the root server.

- When the *root server runs this sensor*, the message indicating what ports were found open on the target looks like this example:
2010-01-08 11:08:22,301 DiscoverManager [DiscoverWorker-10] PortScanSensor-1.2.3.4 INFO portscan.PortScanSensor - PortScanAgent(1.2.3.4): **found [22]**
- When the *anchor runs this sensor*, the message in the sensor logs on the **Server** looks like this example:
2010-01-08 11:19:29,292 DiscoverManager [DiscoverWorker-4] PortScanSensor-1.2.3.4 DEBUG cdb.AnchorClient - remoteDiscover, **result: [22]**

The PortScan sensor on the root server contains the message “found [X]” when port X is found available by the root server. When port X is found available by an anchor, the same log on the root server instead contains the message “result: [X]”.

Port in use

If the anchor cannot listen on the configured socket (8497 by default), most likely another process is listening on it

The anchor sensor log then contains an error similar to these examples:

```
port in use
unable to create anchor
tunnel to server has not been created
```

If the anchor cannot listen on its configured socket, the anchor sensor log on the root server contains a error message similar to “port in use,” “unable to create anchor,” or “tunnel to server has not been created.” To fix this issue, you must log on to the anchor and determine what process is using that port and why. Then take necessary actions to free the port.

Solution to the port-in-use error

- Either an application is using the socket, in which case you must configure this application or the remote anchor port to use another socket
- If an anchor process from a previous discovery is still running; you must stop it manually
Use `netstat -a` and `ps -f U <taddm id>` or `ps -ef | grep coll` to determine if running anchor processes exist

If another application is using the socket, either change that application or change the remote anchor port from the Data Management Console, as shown earlier in this presentation. If an anchor process is incorrectly still running from a previous discovery, stop the anchor manually. If this problem continues to happen, call IBM Support for further analysis.

AllowTcpForwarding

- As per the troubleshooting documentation, you must set the AllowTcpForwarding option on anchors to Yes in the sshd_config file
If confirmed that ports are not in use, this setting is the likely cause of failure

- Example of logged errors:

```
2011-03-22 07:10:14,003 DiscoverManager [DiscoverWorker-4] AnchorSensor-
anchor-@anchor_IP-8431 DEBUG cdb.AnchorClient - ClientConnection(client(0)
): making a new client connection to: (anchor_IP,8431/8432)
2011-03-22 07:10:14,003 DiscoverManager [DiscoverWorker-4] AnchorSensor-
anchor-@anchor_IP-8431 DEBUG cdb.AnchorClient -
<snip stack trace>
2011-03-22 07:10:19,003 DiscoverManager [DiscoverWorker-4] AnchorSensor-
anchor-@anchor_IP-8431 DEBUG cdb.AnchorClient - client(0) closing
connection to: anchor_IP
2011-03-22 07:10:19,019 DiscoverManager [DiscoverWorker-4] AnchorSensor-
anchor-@anchor_IP-8431 ERROR cdb.AnchorClient - [ClientConnection.E.2]
unable to create i/o streams to address:anchor_IP, exception:
java.net.SocketException: Connection reset
```

As stated previously, and in the documentation, you must set the AllowTcpForwarding parameter to TRUE in the sshd_config file.

Windows anchors

- Ensure that the Windows Firewall is disabled
It is enabled by default for Windows 2008
- Having UAC enabled can also cause connection problems
- Because most reported problems tend to be a result of the version of cygwin that is used, check the Installation Guide for hardware and software requirements
- A good technote (titled "Windows Anchor Fails") can be found here
<http://www-01.ibm.com/support/docview.wss?tcss=Newsletter&uid=swg21459040>
- Always search the Tivoli Knowledge Base that is found at
<http://www.ibm.com/support/us/en/>
or to go directly to Tivoli Application Dependency Discovery Manager:
http://www-947.ibm.com/support/entry/portal/Overview/Software/Tivoli/Tivoli_Application_Dependency_Discovery_Manager

For problems with anchors on a Windows platform, if possible, test using a Linux box as an anchor to rule out if the issue is on the firewall or in the network configuration. Check the Installation Guide to confirm that proper hardware and software is being used.

Summary

Now that you completed this module, you can perform these tasks:

- Describe how anchors work
- Diagnose common errors

Now that you completed this module, you can describe how Tivoli Application Dependency Discovery Manager anchors work, and you can diagnose common errors.

Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, Tivoli, and WebSphere are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)" at <http://www.ibm.com/legal/copytrade.shtml>

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Windows, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.

© Copyright International Business Machines Corporation 2013. All rights reserved.