



Creating a Custom Protocol

First Published: April 4, 2006

Last Updated: April 4, 2006

Network-Based Application Recognition (NBAR) recognizes and classifies network traffic on the basis of a set of protocols and application types. You can add to the set of protocols and application types that NBAR recognizes by creating custom protocols.

Creating custom protocols is an optional process. However, custom protocols extend the capability of NBAR Protocol Discovery to classify and monitor additional static port applications and allow NBAR to classify nonsupported static port traffic.

This module contains concepts and tasks for creating a custom protocol.

Finding Feature Information in This Module

Your Cisco IOS software release may not support all of the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To reach links to specific feature documentation in this module and to see a list of the releases in which each feature is supported, use the [“Feature Information for Creating a Custom Protocol”](#) section on page 12.

Finding Support Information for Platforms and Cisco IOS and Catalyst OS Software Images

Use Cisco Feature Navigator to find information about platform support and Cisco IOS and Catalyst OS software image support. To access Cisco Feature Navigator, go to <http://www.cisco.com/go/cfn>. An account on Cisco.com is not required.

Contents

- [Prerequisites for Creating a Custom Protocol, page 2](#)
- [Information About Creating a Custom Protocol, page 2](#)
- [How to Create a Custom Protocol, page 3](#)
- [Configuration Examples for Creating a Custom Protocol, page 10](#)
- [Additional References, page 11](#)
- [Feature Information for Creating a Custom Protocol, page 12](#)



Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

© 2007 Cisco Systems, Inc. All rights reserved.

Prerequisites for Creating a Custom Protocol

Before enabling Protocol Discovery, read the information in the “[Classifying Network Traffic Using NBAR](#)” module.

Information About Creating a Custom Protocol

Before creating a custom protocol, you should understand the following concepts:

- [NBAR and Custom Protocols, page 2](#)
- [MQC and NBAR Custom Protocols, page 3](#)

NBAR and Custom Protocols

NBAR supports the use of custom protocols to identify custom applications. Custom protocols support static port-based protocols and applications that NBAR does not currently support.

**Note**

For a list of NBAR-supported protocols, see the [Classifying Network Traffic Using NBAR](#)” module.

With NBAR supporting the use of custom protocols, NBAR can map static TCP and UDP port numbers to the custom protocols.

Initially, NBAR included the following features related to custom protocols and applications:

- Custom protocols had to be named custom-xx, with xx being a number.
- Ten custom applications can be assigned using NBAR, and each custom application can have up to 16 TCP and 16 UDP ports each mapped to the individual custom protocol. The real-time statistics of each custom protocol can be monitored using Protocol Discovery.

In Cisco IOS Release 12.3(4)T, the following enhancements to custom protocols were introduced:

- The ability to inspect the payload for certain matching string patterns at a specific offset.
- The ability to allow users to define the names of their custom protocol applications. The user-named protocol can then be used by Protocol Discovery, the Protocol Discovery MIB, the **match protocol** command, and the **ip nbar port-map** command as an NBAR-supported protocol.
- The ability for NBAR to inspect custom protocols specified by traffic direction (that is, traffic heading toward a source or destination rather than traffic in both directions) if desired by the user.
- CLI support that allows a user who is configuring a custom application to specify a range of ports rather specifying each port individually.

In Cisco IOS Release 12.4(1)T, the following enhancements to custom protocols were introduced:

- The **variable** keyword, the *field-name* argument, and the *field-length* argument were added to the **ip nbar custom** command.

This additional keyword and two additional arguments allow for creation of more than one custom protocol based on the same port numbers.

- After creating a variable when creating a custom protocol, you can use the **match protocol** command to classify traffic on the basis of a specific value in the custom protocol.

**Note**

For more information about these quality of service (QoS) commands, see the [Cisco IOS Quality of Service Solutions Command Reference](#).

MQC and NBAR Custom Protocols

NBAR recognizes and classifies network traffic by protocol or application. You can extend the set of protocols and applications that NBAR recognizes by creating a custom protocol. Custom protocols extend the capability of NBAR Protocol Discovery to classify and monitor additional static port applications and allow NBAR to classify nonsupported static port traffic. You define a custom protocol by using the keywords and arguments of the **ip nbar custom** command. However, after you define the custom protocol, you must create a traffic class and configure a traffic policy (policy map) to use the custom protocol when NBAR classifies traffic. To create traffic classes and configure traffic policies, use the functionality of the Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC). The MQC is a command-line interface that allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach these traffic policies to interfaces. For more information about NBAR and the functionality of the MQC, see the “[Configuring NBAR Using the MQC](#)” module.

How to Create a Custom Protocol

This section contains the following tasks:

- [Defining a Custom Protocol, page 3](#) (required)
- [Configuring a Traffic Class to Use the Custom Protocol, page 5](#) (required)
- [Configuring a Traffic Policy, page 6](#) (required)
- [Attaching the Traffic Policy to an Interface, page 8](#) (required)
- [Displaying Custom Protocol Information, page 9](#) (optional)

Defining a Custom Protocol

Custom protocols extend the capability of NBAR Protocol Discovery to classify and monitor additional static port applications and allow NBAR to classify nonsupported static port traffic.

To define a custom protocol, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar custom** *name* [*offset* [*format value*]] [**variable** *field-name field-length*] [*source* | *destination*] [**tcp** | **udp**] [**range** *start end* | *port-number*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Router> enable	Enables privileged EXEC mode. <ul style="list-style-type: none"> Enter your password if prompted.
Step 2	configure terminal Example: Router# configure terminal	Enters global configuration mode.
Step 3	ip nbar custom name [offset [format value]] [variable field-name field-length] [source destination] [tcp udp] [range start end port-number] Example: Router(config)# ip nbar custom app_sales1 5 ascii SALES source tcp 4567	Extends the capability of NBAR Protocol Discovery to classify and monitor additional static port applications or allows NBAR to classify nonsupported static port traffic. <ul style="list-style-type: none"> Enter the custom protocol name and any other optional keywords and arguments.
Step 4	end Example: Router(config)# end	(Optional) Exits global configuration mode.

Examples

Custom Application Examples for Cisco IOS Releases Prior to 12.3(4)T

In the following example, a gaming application that runs on TCP port 8877 needs to be classified using NBAR. You can use custom-01 to map TCP port 8877 by entering the following command:

```
Router(config)# ip nbar custom-01 tcp 8877
```



Note The configuration shown in this example is supported in subsequent Cisco IOS releases but is required in all prior releases.

Custom Application Examples for Cisco IOS Release 12.3(4)T and Later Releases

In the following example, the custom protocol app_sales1 will identify TCP packets that have a source port of 4567 and that contain the term “SALES” in the fifth byte of the payload:

```
Router(config)# ip nbar custom app_sales1 5 ascii SALES source tcp 4567
```

In the following example, the custom protocol virus_home will identify UDP packets that have a destination port of 3000 and that contain “0x56” in the seventh byte of the payload:

```
Router(config)# ip nbar custom virus_home 7 hex 0x56 destination udp 3000
```

In the following example, the custom protocol media_new will identify TCP packets that have a destination or source port of 4500 and that have a value of 90 at the sixth byte of the payload:

```
Router(config)# ip nbar custom media_new 6 decimal 90 tcp 4500
```

In the following example, the custom protocol `msn1` will look for TCP packets that have a destination or source port of 6700:

```
Router(config)# ip nbar custom msn1 tcp 6700
```

In the following example, the custom protocol `mail_x` will look for UDP packets that have a destination port of 8202:

```
Router(config)# ip nbar custom mail_x destination udp 8202
```

In the following example, the custom protocol `mail_y` will look for UDP packets that have destination ports between 3000 and 4000 inclusive:

```
Router(config)# ip nbar custom mail_y destination udp range 3000 4000
```

Configuring a Traffic Class to Use the Custom Protocol

Traffic classes can be used to organize packets into groups on the basis of a user-specified criterion. For example, traffic classes can be configured to match packets on the basis of the protocol type or application recognized by NBAR. In this case, the traffic class is configured to match on the basis of the custom protocol.

To configure a traffic class to use the custom protocol, perform the following steps.



Note

The **match protocol** command is shown at Step 4. For the *protocol-name* argument, enter the protocol name used as the match criteria. For a custom protocol, use the protocol specified by the *name* argument of the **ip nbar custom** command. (See Step 3 of the “[Defining a Custom Protocol](#)” task.)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map [match-all | match-any] class-map-name**
4. **match protocol protocol-name**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Router> enable	Enables privileged EXEC mode. <ul style="list-style-type: none"> • Enter your password if prompted.
Step 2	configure terminal Example: Router# configure terminal	Enters global configuration mode.

	Command or Action	Purpose
Step 3	<pre>class-map [match-all match-any] class-map-name</pre> <p>Example: Router(config)# class-map cmap1 </p>	<p>Creates a class map to be used for matching packets to a specified class and enters class-map configuration mode.</p> <ul style="list-style-type: none"> Enter the name of the class map.
Step 4	<pre>match protocol protocol-name</pre> <p>Example: Router(config-cmap)# match protocol app_sales1 </p>	<p>Configures NBAR to match traffic on the basis of the specified protocol.</p> <ul style="list-style-type: none"> For the <i>protocol-name</i> argument, enter the protocol name used as the match criterion. For a custom protocol, use the protocol specified by the <i>name</i> argument of the ip nbar custom command. (See Step 3 of the “Defining a Custom Protocol” task.)
Step 5	<pre>end</pre> <p>Example: Router(config-cmap)# end </p>	<p>(Optional) Exits class-map configuration mode.</p>

Examples

In the following example, the **variable** keyword is used while creating a custom protocol, and class maps are configured to classify different values within the variable field into different traffic classes. Specifically, in the example below, variable scid values 0x15, 0x21, and 0x27 will be classified into class map active-craft, while scid values 0x11, 0x22, and 0x25 will be classified into class map passive-craft.

```
Router(config)# ip nbar custom ftdd 23 variable scid 1 tcp range 5001 5005

Router(config)# class-map active-craft
Router(config-cmap)# match protocol ftdd scid 0x15
Router(config-cmap)# match protocol ftdd scid 0x21
Router(config-cmap)# match protocol ftdd scid 0x27

Router(config)# class-map passive-craft
Router(config-cmap)# match protocol ftdd scid 0x11
Router(config-cmap)# match protocol ftdd scid 0x22
Router(config-cmap)# match protocol ftdd scid 0x25
```

Configuring a Traffic Policy

Traffic that matches a user-specified criterion can be organized into specific classes. The traffic in those classes can, in turn, receive specific QoS treatment when that class is included in a policy map.

To configure a traffic policy, perform the following steps.



Note

The **bandwidth** command is shown at Step 5. The **bandwidth** command configures the QoS feature class-based weighted fair queuing (CBWFQ). CBWFQ is just an example of a QoS feature that can be configured. Use the appropriate command for the QoS feature that you want to use.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. **bandwidth** {*bandwidth-kbps* | **remaining percent** *percentage* | **percent** *percentage*}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Router> enable	Enables privileged EXEC mode. <ul style="list-style-type: none"> • Enter your password if prompted.
Step 2	configure terminal Example: Router# configure terminal	Enters global configuration mode.
Step 3	policy-map <i>policy-map-name</i> Example: Router(config)# policy-map policy1	Creates or modifies a policy map that can be attached to one or more interfaces and enters policy-map configuration mode. <ul style="list-style-type: none"> • Enter the name of the policy map.
Step 4	class { <i>class-name</i> class-default }	Specifies the name of the class whose policy you want to create or change and enters policy-map class configuration mode. <ul style="list-style-type: none"> • Enter the specific class name or enter the class-default keyword.
Step 5	bandwidth { <i>bandwidth-kbps</i> remaining percent <i>percentage</i> percent <i>percentage</i> } Example: Router(config-pmap-c)# bandwidth percent 50	(Optional) Specifies or modifies the bandwidth allocated for a class belonging to a policy map. <ul style="list-style-type: none"> • Enter the amount of bandwidth as a number of kbps, a relative percentage of bandwidth, or an absolute amount of bandwidth. <p>Note The bandwidth command configures the QoS feature class-based weighted fair queuing (CBWFQ). CBWFQ is just an example of a QoS feature that can be configured. Use the appropriate command for the QoS feature that you want to use.</p>
Step 6	end Example: Router(config-pmap-c)# end	(Optional) Exits policy-map class configuration mode.

Attaching the Traffic Policy to an Interface

After a traffic policy (policy map) is created, the next step is to attach the policy map to an interface. Policy maps can be attached to either the input or output direction of the interface.



Note

Depending on the needs of your network, you may need to attach the policy map to a subinterface, an ATM PVC, a Frame Relay DLCI, or other type of interface.

To attach the traffic policy to an interface, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **pvc** [*name*] *vpi/vci* [*ilmi* | *qsaal* | *smds* | *l2transport*]
5. **exit**
6. **service-policy** {**input** | **output**} *policy-map-name*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Router> enable	Enables privileged EXEC mode. <ul style="list-style-type: none"> Enter your password if prompted.
Step 2	configure terminal Example: Router# configure terminal	Enters global configuration mode.
Step 3	interface <i>type number</i> [<i>name-tag</i>] Example: Router(config)# interface ethernet 2/4	Configures an interface type and enters interface configuration mode. <ul style="list-style-type: none"> Enter the interface type and the interface number.
Step 4	pvc [<i>name</i>] <i>vpi/vci</i> [<i>ilmi</i> <i>qsaal</i> <i>smds</i> <i>l2transport</i>] Example: Router(config-if)# pvc cisco 0/16	(Optional) Creates or assigns a name to an ATM permanent virtual circuit (PVC), specifies the encapsulation type on an ATM PVC, and enters ATM virtual circuit configuration mode. <ul style="list-style-type: none"> Enter the PVC name, the ATM network virtual path identifier, and the network virtual channel identifier. <p>Note This step is required only if you are attaching the policy map to an ATM PVC. If you are not attaching the policy map to an ATM PVC, advance to Step 6.</p>

	Command or Action	Purpose
Step 5	exit Example: Router(config-atm-vc)# exit	(Optional) Returns to interface configuration mode. Note This step is required only if you are attaching the policy map to an ATM PVC and you completed Step 4 . If you are not attaching the policy map to an ATM PVC, advance to Step 6 .
Step 6	service-policy {input output} policy-map-name Example: Router(config-if)# service-policy input policy1	Attaches a policy map to an input or output interface. <ul style="list-style-type: none"> • Enter the name of the policy map. Note Policy maps can be configured on ingress or egress routers. They can also be attached in the input or output direction of an interface. The direction (input or output) and the router (ingress or egress) to which the policy map should be attached vary according to your network configuration. When using the service-policy command to attach the policy map to an interface, be sure to choose the router and the interface direction that are appropriate for your network configuration.
Step 7	end Example: Router(config-if)# end	(Optional) Returns to privileged EXEC mode.

Displaying Custom Protocol Information

After you create a custom protocol and match traffic on the basis of that custom protocol, you can use the **show ip nbar port-map** command to display information about that custom protocol.

To display custom protocol information, complete the following steps.

SUMMARY STEPS

1. **enable**
2. **show ip nbar port-map** [*protocol-name*]
3. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Router> enable	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	show ip nbar port-map [<i>protocol-name</i>] Example: Router# show ip nbar port-map	Displays the current protocol-to-port mappings in use by NBAR. • (Optional) Enter a specific protocol name.
Step 3	exit Example: Router# exit	(Optional) Exits privileged EXEC mode.

Configuration Examples for Creating a Custom Protocol

This section provides the following configuration examples:

- [Creating a Custom Protocol: Example, page 10](#)
- [Configuring a Traffic Class to Use the Custom Protocol: Example, page 10](#)
- [Configuring a Traffic Policy: Example, page 11](#)
- [Attaching the Traffic Policy to an Interface: Example, page 11](#)
- [Displaying Custom Protocol Information: Example, page 11](#)

Creating a Custom Protocol: Example

In the following example, the custom protocol called `app_sales1` identifies TCP packets that have a source port of 4567 and that contain the term SALES in the fifth byte of the payload.

```
Router> enable
Router# configure terminal
Router(config)# ip nbar custom app_sales1 5 ascii SALES source tcp 4567
Router(config)# end
```

Configuring a Traffic Class to Use the Custom Protocol: Example

In the following example, a class called `cmap1` has been configured. All traffic that matches the custom `app_sales1` protocol will be placed in the `cmap1` class.

```
Router> enable
Router# configure terminal
Router(config)# class-map cmap1
Router(config-cmap)# match protocol app_sales1
Router(config-cmap)# end
```

Configuring a Traffic Policy: Example

In the following example, a traffic policy (policy map) called `policy1` has been configured. Policy1 contains a class called `class1`, within which CBWFQ has been enabled.

```
Router> enable
Router# configure terminal
Router(config)# policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth percent 50
Router(config-pmap-c)# end
```

**Note**

In the above example, the `bandwidth` command is used to enable Class-Based Weighted Fair Queuing (CBWFQ). CBWFQ is only an example of one QoS feature that can be applied in a traffic policy (policy map). Use the appropriate command for the QoS feature that you want to use.

Attaching the Traffic Policy to an Interface: Example

In the following example, the traffic policy (policy map) called `policy1` has been attached to ethernet interface `2/4` in the input direction of the interface.

```
Router> enable
Router# configure terminal
Router(config)# interface ethernet 2/4
Router(config-if)# service-policy input policy1
Router(config-if)# end
```

Displaying Custom Protocol Information: Example

The following is sample output of the `show ip nbar port-map` command. This command displays the current protocol-to-port mappings in use by NBAR. Use the display to verify that these mappings are correct.

```
Router# show ip nbar port-map

port-map bgp      udp 179
port-map bgp      tcp 179
port-map cuseeme  udp 7648 7649
port-map cuseeme  tcp 7648 7649
port-map dhcp     udp 67 68
port-map dhcp     tcp 67 68
```

If the `ip nbar port-map` command has been used, the `show ip nbar port-map` command displays the ports assigned to the protocol.

If the `no ip nbar port-map` command has been used, the `show ip nbar port-map` command displays the default ports. To limit the display to a specific protocol, use the `protocol-name` argument of the `show ip nbar port-map` command.

Additional References

The following sections provide references related to creating a custom protocol.

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	Cisco IOS Quality of Service Solutions Command Reference
MQC, traffic policies (policy maps), and traffic classes	“Applying QoS Features Using the MQC” module
Concepts and information about NBAR	“Classifying Network Traffic Using NBAR” module
Information about enabling Protocol Discovery	“Enabling Protocol Discovery” module
Configuring NBAR using the MQC	“Configuring NBAR Using the MQC” module
Adding application recognition modules (also known as PDLMs)	“Adding Application Recognition Modules” module

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Feature Information for Creating a Custom Protocol

[Table 1](#) lists the release history for this feature.

For information on a feature in this technology that is not documented here, see the [“Classifying Network Traffic Using NBAR Features Roadmap”](#) module.

Not all commands may be available in your Cisco IOS software release. For release information about a specific command, see the command reference documentation.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which Cisco IOS and Catalyst OS software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to <http://www.cisco.com/go/cfn>. An account on Cisco.com is not required..



Note

[Table 1](#) lists only the Cisco IOS software release that introduced support for a given feature in a given Cisco IOS software release train. Unless noted otherwise, subsequent releases of that Cisco IOS software release train also support that feature.

Table 1 **Feature Information for Creating a Custom Protocol**

Feature Name	Releases	Feature Information
NBAR – Multiple Matches Per Port	12.4(2)T	<p>Provides the ability for NBAR to distinguish between values of an attribute within the traffic stream of a particular application on a TCP or UDP port.</p> <p>The following sections provide information about the NBAR – Multiple Matches Per Port feature:</p> <ul style="list-style-type: none"> • Information About Creating a Custom Protocol, page 2 • How to Create a Custom Protocol, page 3
NBAR User-Defined Custom Application Classification	12.3(4)T	<p>Provides ability to identify TCP- or UDP-based applications by using a character string or value. The character string or value is used to match traffic within the packet payload.</p> <p>The following sections provide information about the NBAR User-Defined Custom Application Classification feature:</p> <ul style="list-style-type: none"> • Information About Creating a Custom Protocol, page 2 • How to Create a Custom Protocol, page 3

Cisco and the Cisco Logo are trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and other countries. A listing of Cisco's trademarks can be found at www.cisco.com/go/trademarks. Third party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1005R)

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 2007 Cisco Systems, Inc. All rights reserved.

