Cisco Application Visibility and Control Field Definition Guide for Third-Party Customers

November 29, 2017

Cisco Systems, Inc.
www.cisco.com

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco Application Visibility and Control Field Definition Guide for Third-Party Customers

Preface i

Objective i

Document Revision History ii

Audience ii

Organization iii

Related Documentations iii

Obtaining Documentation and Submitting a Service Request iii

Cisco Application Visibility and Control Solution Overview 1-1

Overview of the Cisco AVC 1-1

Cisco AVC Metric Definitions 2-1

NBAR2 Metrics 2-1

NBAR2 Application ID 2-2

NBAR2 HTTP Fields 2-5

HTTP Host 2-5

URI Statistics 2-7

HTTP Proxy 2-8

Application Response Time Metrics 2-8

Client Network Time [sum/min/max] 2-12

Long Lived Client Network Time [sum/min/max/num-samples] 2-12

Server Network Time [sum/min/max] 2-13

Long Lived Server Network Time [sum/min/max/num-samples] 2-14

Network Time [sum/min/max] 2-15

Long Lived Network Time [sum/min/max/num-samples] 2-16

Server Response Time [sum/min/max] 2-16

Response Time [sum/min/max] 2-18

Total Response Time [sum/min/max] 2-18

Total Transaction Time [sum/min/max] 2-19

ART Client Bytes/Packets(Layer 4) 2-19

ART Server Bytes/Packet(Layer 4) 2-19

ART Client Bytes(Layer 3) 2-20

ART Server Bytes(Layer 3) 2-20

ART Count New Connections 2-21

ART Concurrent Sessions 2-21

ART Count Responses 2-21
Responses Histogram Buckets (7-Bucket Histogram) 2-22
ART Count Late Responses 2-22
ART Count Transactions 2-23
ART Client Retransmissions Bytes 2-23
ART Client Retransmissions Packets 2-23
ART Server Retransmissions Bytes 2-24
ART Server Retransmissions Packets 2-24
Client Bytes 2-25
ART All Metrics 2-25
Cisco WAAS Interoperation Metrics 2-25
WAAS Segment Number 2-26
WAAS Passthrough Reason 2-27
WAAS DRE Input 2-27
WAAS DRE Output 2-28
WAAS Lempel-Ziv Input 2-28
WAAS Lempel-Ziv Output 2-28
WAAS Input Bytes 2-29
WAAS Output Bytes 2-29
WAAS Connection Mode 2-29
WAAS All Metrics 2-30
QoS Metrics 2-30
QoS Policy Classification Hierarchy 2-31
QoS Queue Drops 2-33
Media Performance Metrics 2-33
General Metrics 2-34
Absolute Timestamp 2-34
Option Template 2-35
Traffic Volume 2-35
Field ID Comparison 2-35

AVC Configuration Examples and Troubleshooting Tips 3-1

Troubleshooting Tips and Debug Commands for IOS Platform 4-41

Troubleshooting Tips 4-41
Debug and Show Commands 4-42

Preface

First Published: March 29, 2013
Revised: March 26, 2015

This preface describes who should read the *Cisco Application Visibility and Control Field Definition Guide*, how it is organized, and its document conventions. It contains the following sections:

- Objective, page i
- Document Revision History, page ii
- Audience, page ii
- Organization, page iii
- Related Documentations, page iii
- Obtaining Documentation and Submitting a Service Request, page iii

Objective

The Cisco Application Visibility and Control (AVC) solution is available in the Cisco ISR G2 and the Cisco ASR 1000 Series Aggregation Services Routers. It uses stateful Deep Packet Inspection (DPI) called Cisco Next Generation Network-Based Application Recognition (NBAR2) to identify, analyze, and optimize application traffic. The information gathered in NBAR2 along with performance metrics are exported through standard flow records format, such as NetFlow Version 9 and Internet Protocol Flow Information Export (IPFIX), which can be processed by Cisco Prime or third-party reporting tools.

The objective of this guide is to provide an overview of and describe the metrics and Flexible NetFlow (FNF) IDs exported by Cisco ISR G2 and the Cisco ASR 1000 Series Aggregation Services Routers.

This document covers AVC exports for MMA which is a normalization of exported data for Cisco IOS and IOS XE releases.

Document Revision History

Table 1 records technical changes to this document. The table shows the Cisco IOS software release number and document revision number for the change, the date of the change, and a brief summary of the change.

Table 1 Document Revision History

Release No.	Date	Change Summary
15.5(2)T and XE 3.15	March, 2015	Updated the ART metric section with Concurrent Session metric: • "Application Response Time Metrics" section on page 8
15.5(1)T and XE 3.14	November, 2014	Updated the ART metric section with new metrics: • "Application Response Time Metrics" section on page 8
15.4(1)T and XE 3.11S	November, 2013	Added the following FNF exported fields in Chapter A : • observationPointId • monitoringIntervalStartMilliSeconds Converged CLIs for IOS and IOS XE in the following sections: • "Application Response Time Metrics" section on page 8 • "QoS Metrics" section on page 30 Fields now represent MMA exported data.
XE 3.9.2S	August, 2013	Added L3 byte counter to XE platform. Added the following CLIs in "ART All Metrics" section on page 25: • collect connection client counter bytes network long • collect connection server counter bytes network long
XE 3.10S	July, 2013	Added support for **collect application http uri statistics** field for XE platform: • "URI Statistics" section on page 7 Added ezPM feature that provides a simple configuration for the entire AVC.
15.2(4)M2 and XE 3.9S	March, 2013	Initial version of this document.

Audience

This guide is intended for developers who need to develop reporting tools utilizing the flow records exported by the Cisco AVC solution. It is assumed that the reader is technically knowledgeable and familiar with Cisco routers and Cisco IOS software and features.

Organization

This guide includes the following chapters:

Chapter	Title	Description
Chapter 1	Cisco Application Visibility and Control Solution Overview	High-level overview of Cisco AVC solution and its features.
Chapter 2	Cisco AVC Metric Definitions	Metric definition for ISR G2 and ASR 1000.
Chapter 3	AVC Configuration Examples and Troubleshooting Tips	AVC use cases and examples.
Chapter A	AVC-Related Exported Fields	New Flexible NetFlow (FNF) fields and the CLI used to retrieve the value of the fields.
Chapter B	DPI/L7 Extracted Fields	Deep packet inspection (DPI)/L7 extracted fields and the CLI used to retrieve the value of the fields.
Chapter C	Fields that Require Punt to the Route Processor	Media monitoring/metadata metrics that require punt to the record processor.

Related Documentations

- *Cisco Application Visibility and Control User Guide*
- *Application Visibility and Control Configuration Guide*
- *Cisco IOS Flexible NetFlow Command Reference*
- *Cisco IOS NetFlow Configuration Guide*

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as an RSS feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service. Cisco currently supports RSS Version 2.0.

Cisco Application Visibility and Control Solution Overview

First Published: March 29, 2013
Revised: March 26, 2015

Overview of the Cisco AVC

The Application Visibility and Control (AVC) solution is Cisco's strategic program to add application-level intelligence to a variety of network devices, beginning with branch and WAN aggregation routers and wireless LAN controllers. AVC recognizes and classifies more than 1,000 applications, and uses this classification to perform per-application monitoring of traditional NetFlow statistics, of transactional Application Response Time metrics, and of Medianet metrics such as latency and jitter. The per-application metrics are exported via Flexible NetFlow version 9 and IPFIX for analysis, reporting, and visualization by partner network management systems. Control policies, such as Quality of Service (QoS) and Cisco Performance-based Routing (PfR), can be tuned and enhanced by matching the individual applications or categories that AVC recognizes. All of this is accomplished without the need to deploy and manage separate hardware or software probes in each network location; it is integrated directly into the Cisco devices.

For more information about AVC and its architecture, see the information available in the Application Visibility and Control Developer Center at http://developer.cisco.com/web/avc/overview

For more information about AVC on the Cisco IOS-XE platform, see the *Application Visibility and Control Configuration Guide* and the *Cisco Application Visibility and Control User Guide.*

Cisco AVC Metric Definitions

First Published: March 29, 2013
Revised: March 26, 2015

This chapter contains the following sections:

- NBAR2 Metrics, page 2-1
- Application Response Time Metrics, page 2-8
- Cisco WAAS Interoperation Metrics, page 2-25
- QoS Metrics, page 2-30
- Media Performance Metrics, page 2-33
- Option Template, page 2-35

NBAR2 Metrics

Next Generation Network-Based Application Recognition (NBAR2) metrics are the metrics and application information with the latest protocol pack that comes with the number of applications supported.

Field IDs represent the fields in a record. The format of the record consists of the order of the fields, which is communicated to the NetFlow template.

Table 2-1 lists the NBAR2 metric summary.

Table 2-1 NBAR2 Metrics Summary

Field Name	Field ID (IOS)	Field ID (IOS XE)
NBAR2 Application ID	95	95
HTTP Host	45003	45003
URI Statistics (Hit Count)	42125	—
Extracted Fields	—	45003

For information about HTTP proxy, see the "HTTP Proxy" section on page 2-8. For information about the Cisco IOS-XE-specific extracted fields, see the *Cisco Application Visibility and Control User Guide*.

NBAR2 Application ID

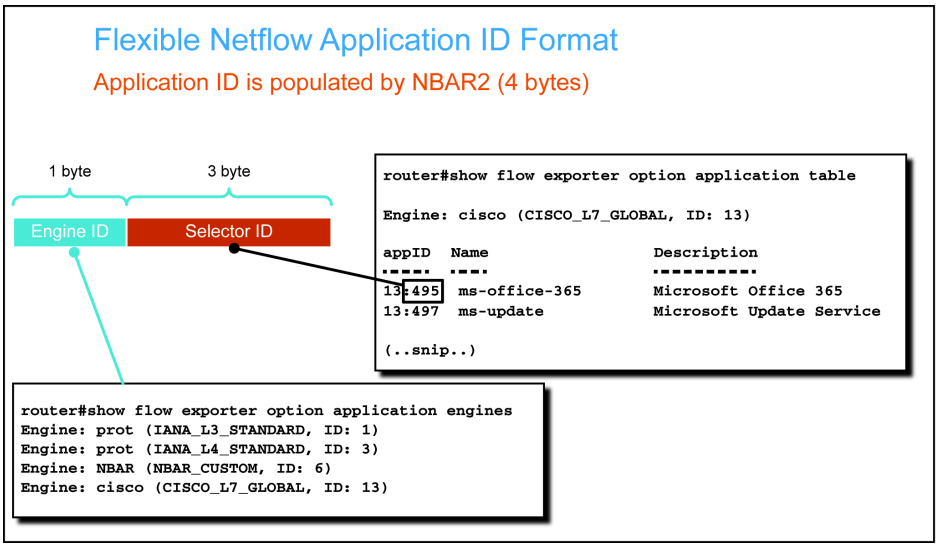
The following table lists information about the NBAR2 application ID metric.

Table 2-2 Information About NBAR2 Application ID Metric

Description	Provides the Layer 7-level information for a particular flow.
CLI	**Cisco IOS and Cisco IOS XE:** **collect application name**
Export Field ID	**Cisco IOS and Cisco IOS XE:** 95
Export Protocol	NetFlow v9, IPFIX

Figure 2-1 illustrates the Flexible NetFlow Application ID format.

Figure 2-1 Flexible NetFlow Application ID Format



An ID exported by AVC explains which application a particular flow belongs to. Figure 2-1 shows that the Application ID is divided into two parts:

- Engine ID— A unique identifier for the engine that determined the Selector ID. The Classification Engine ID defines the context for the Selector ID. The Engine ID is the first eight bits that provide information about the engine that classifies the flow. IANA-L4, CANA-L3, and so on, are some of the engines that can be classified using an engine ID. Note that the Engine ID does not represent the NBAR2 mechanism used to classify the application. For more information about the engine IDs, see the information available here: http://tools.ietf.org/html/rfc6759
- Selector ID—The remaining 24 bits that provide information about the application. 495(MS office) is one of the applications that can be classified using the classification ID.

The **collect application name** command exports only the application ID, which is a number that may not be understood by a collector. To export a mapping table between an application ID to application name and description, use the **option application-table** command in flow exporter configuration.

Example:

```
flow exporter my-exporter
   option application-table
```

The following example shows the output of the **show flow exporter option application table** command:

```
avc-2901a#show flow exporter option application table

Engine: prot (IANA_L3_STANDARD, ID: 1)

appID  Name                      Description
-----  ----                      -----------
1:8    egp                       Exterior Gateway Protocol
1:47   gre                       General Routing Encapsulation
1:1    icmp                      Internet Control Message Protocol
1:88   eigrp                     Enhanced Interior Gateway Routing Protocol
1:4    ipinip                    IP in IP
1:89   ospf                      Open Shortest Path First
1:46   rsvp                      Resource Reservation Protocol
1:0    hopopt                    DEPRECATED, traffic will not match
1:3    ggp                       Gateway-to-Gateway
1:5    st                        Stream
1:7    cbt                       CBT
1:9    igrp                      Cisco interior gateway
1:10   bbnrccmon                 BBN RCC Monitoring
1:11   nvp-ii                    Network Voice Protocol
1:12   pup                       PUP


Engine: NBAR (NBAR_CUSTOM, ID: 6)

appID  Name                      Description
-----  ----                      -----------
6:244  custom-10                 Custom protocol custom-10
6:245  custom-09                 Custom protocol custom-09
6:246  custom-08                 Custom protocol custom-08
6:247  custom-07                 Custom protocol custom-07
6:248  custom-06                 Custom protocol custom-06
6:249  custom-05                 Custom protocol custom-05
6:250  custom-04                 Custom protocol custom-04
6:251  custom-03                 Custom protocol custom-03
6:252  custom-02                 Custom protocol custom-02
6:253  custom-01                 Custom protocol custom-01


Engine: cisco (CISCO_L7_GLOBAL, ID: 13)

appID  Name                      Description
-----  ----                      -----------
13:0    unclassified              Unclassified traffic
13:1    unknown                   Unknown application
13:9    ipsec                     IPSec traffic
13:12   cuseeme                   CU-SeeMe desktop video conference
13:13   dhcp                      Dynamic Host Configuration Protocol
13:26   netbios                   netbios
13:2000  notes                     DEPRECATED, traffic will not match. Please use lotus-no
13:32   pcanywhere                Symantec pcAnywhere remote desktop
13:41   syslog                    System Logging Utility
13:47   novadigm                  Novadigm EDM
13:49   exchange                  MS-RPC Exchange
13:425  vdolive                   VDOLive streaming video
13:426  netshow                   Microsoft Netshow, media streaming protocol
13:427  streamwork                Xing Technology StreamWorks player
```

```
13:56   citrix                  Citrix Systems Metaframe 3.0
13:57   fasttrack               DEPRECATED, traffic will not match
13:58   gnutella                Gnutella Version2 Traffic, peer-to-peer file-sharing pr
13:59   kazaa2                  DEPRECATED, traffic will not match
13:61   rtp                     Real Time Protocol
13:62   mgcp                    Media Gateway Control Protocol
13:63   skinny                  Skinny Call Control Protocol
```

The following example shows an application-table option. Note that the format of the record is the same for all the export formats, but the Field IDs differ based on the export format protocol. Use **show flow exporter <exporter_name> templates** command to see the format output.

```
Exporter Format: IPFIX (Version 10)
  Template ID    : 259
  Record Size    : 83
  Template layout
  ----------------------------------------------------------------------------
  |                 Field                    |    ID | Ent.ID | Offset |  Size |
  ----------------------------------------------------------------------------
  | APPLICATION ID                           |    95 |        |      0 |     4 |
  | application name                         |    96 |        |      4 |    24 |
  | application description                  |    94 |        |     28 |    55 |
  ----------------------------------------------------------------------------
```

For more information about various attributes of a particular application, use the **option application-attributes** command in configuration mode.

```
Exporter Format: IPFIX (Version 10)
  Template ID    : 260
  Record Size    : 130
  Template layout
  ----------------------------------------------------------------------------
  |                 Field                    |    ID | Ent.ID | Offset |  Size |
  ----------------------------------------------------------------------------
  | APPLICATION ID                           |    95 |        |      0 |     4 |
  | application category name                | 45000 |      9 |      4 |    32 |
  | application sub category name            | 45001 |      9 |     36 |    32 |
  | application group name                   | 45002 |      9 |     68 |    32 |
  | p2p technology                           |   288 |        |    100 |    10 |
  | tunnel technology                        |   289 |        |    110 |    10 |
  | encrypted technology                     |   290 |        |    120 |    10 |
  ----------------------------------------------------------------------------
```

The output of the **option** command in the periodic export of NBAR are sent to the collector for each protocol. The following are the different types of application attributes:

- Category—Provides the first-level categorization for each application.
- Sub-Category—Provides the second-level categorization for each application.
- Application-Group—Identifies the group application that belongs to the same networking application.
- P2P Technology—Specifies whether an application is based on peer-to-peer technology.
- Tunnel Technology—Specifies whether an application tunnels the traffic of other protocols.
- Encrypted—Specifies whether an application is an encrypted networking protocol.

The following example shows how the data looks in an export format:

```
#259: APPLICATION_NAME:13:1 CATEGORY:other APP_SUB_CATEGORY:other APP_GROUP:other
P2P_TECHNOLOGY:unassigned TUNNEL:unassigned ENCRYPTED:unassigned

 #259: APPLICATION_NAME:3:21 CATEGORY:file-sharing APP_SUB_CATEGORY:client-server
APP_GROUP:ftp-group P2P_TECHNOLOGY:no TUNNEL:no ENCRYPTED:no

 #259: APPLICATION_NAME:3:80 CATEGORY:browsing APP_SUB_CATEGORY:other APP_GROUP:other
P2P_TECHNOLOGY:no TUNNEL:no ENCRYPTED:no

 #259: APPLICATION_NAME:1:8 CATEGORY:net-admin APP_SUB_CATEGORY:routing-protocol
APP_GROUP:other P2P_TECHNOLOGY:no TUNNEL:no ENCRYPTED:no

 #259: APPLICATION_NAME:1:47 CATEGORY:layer3-over-ip APP_SUB_CATEGORY:other
APP_GROUP:other P2P_TECHNOLOGY:unassigned TUNNEL:unassigned ENCRYPTED:unassigned

 #259: APPLICATION_NAME:1:1 CATEGORY:net-admin APP_SUB_CATEGORY:network-management
APP_GROUP:other P2P_TECHNOLOGY:no TUNNEL:no ENCRYPTED:no

 #259: APPLICATION_NAME:1:88 CATEGORY:net-admin APP_SUB_CATEGORY:routing-protocol
APP_GROUP:other P2P_TECHNOLOGY:no TUNNEL:no ENCRYPTED:no

 #259: APPLICATION_NAME:1:4 CATEGORY:layer3-over-ip APP_SUB_CATEGORY:other APP_GROUP:other
P2P_TECHNOLOGY:no TUNNEL:yes ENCRYPTED:no

 #259: APPLICATION_NAME:13:9 CATEGORY:internet-privacy
APP_SUB_CATEGORY:tunneling-protocols APP_GROUP:ipsec-group P2P_TECHNOLOGY:no TUNNEL:yes
ENCRYPTED:yes

 #259: APPLICATION_NAME:1:89 CATEGORY:net-admin APP_SUB_CATEGORY:routing-protocol
APP_GROUP:other P2P_TECHNOLOGY:no TUNNEL:no ENCRYPTED:no
```

NBAR2 HTTP Fields

AVC supports the following NBAR2-related fields in Cisco IOS Release 15.2(4)M2, Cisco IOS XE Release 3.9S, and later releases:

- HTTP Host, page 2-5
- URI Statistics, page 2-7

HTTP Host

Table 2-3 lists information about HTTP host metric:

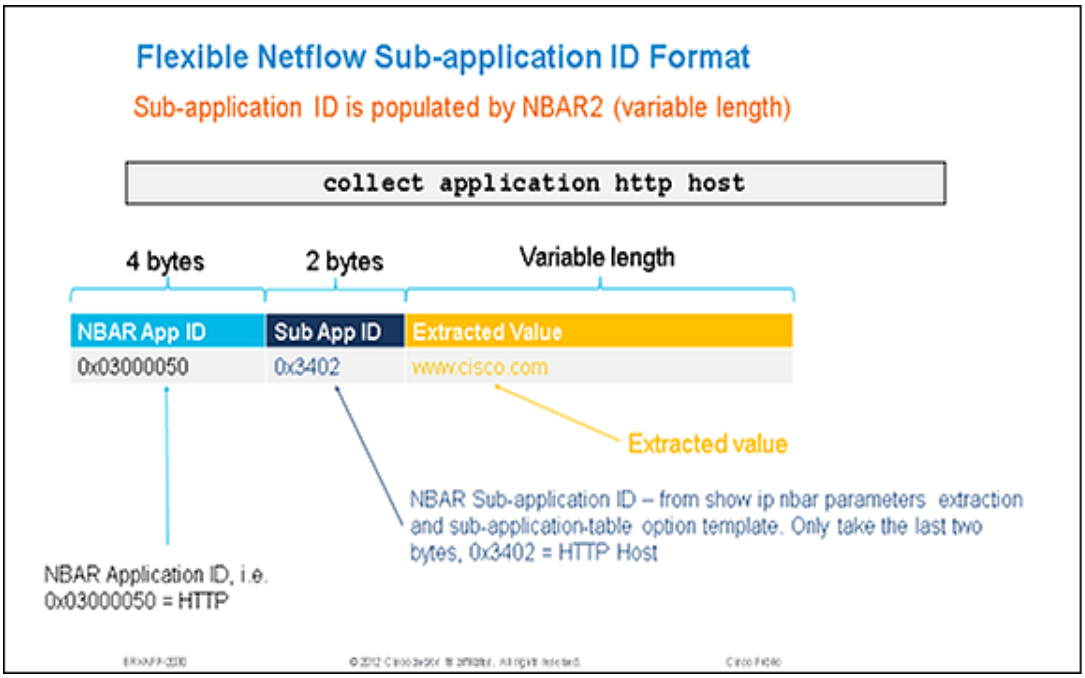
Table 2-3 HTTP Host Metric

Definition	Determines the host name of the flow.
CLI	**IOS and IOS XE:** **collect application http host**
Export Field ID	**IOS and IOS XE:** 45003
Export Protocol	IPFIX

One hostname is exported per flow record.

Figure 2-2 shows the Flexible NetFlow Subapplication ID format.

Figure 2-2 Flexible NetFlow Sub-Application ID Format



The following example shows how the information is exported for HTTP host metric.

URL: http://www.cisco.com/go/avc

The host in this example is *www.cisco.com*.

The following is a list of the various characteristics of the HTTP Host metric:

- The collector identifies the extracted field name and type based on the Application ID and Subapplication ID that are embedded in it. For an HTTP Host, the Application ID is generally 0x03000050. The first byte specifies the Engine ID and the following three bytes are for the Selector ID. The Subapplication ID for the host is 0x3402. The value is the host string, as shown in the following example:

  Application ID: 0x03000050

  Engine ID: IANA-L4

  Selector ID: decimal 80 for HTTP

  Sub-application ID: 0x3402

  Value: www.cisco.com

- The host is collected only when it is configured. It is an independent field and is not related to URI field.
- In IOS platform, P is exported as hostname if HTTP proxy is used.
- If the flow is terminated during the export interval, both the URI and hostname fields will be exported.
- In IOS platform, the maximum length of a hostname is 512 characters long. If a hostname exceeds 512 characters, it will be truncated to 512 bytes and the trailing characters will be dropped. In IOS XE, all extracted fields are variable length fields with a limit of 2KB.
- Multiple transactions are not supported for a host extraction on IOS.

- If the HTTP host is configured for a non-HTTP flow, this field will be exported with no value or zero value size. Microsoft exchange and Telnet are two examples for non-HTTP flows.

URI Statistics

Table 2-4 lists information about URI statistics.

Table 2-4 URI Statistics

Description	Collects and exports the URI and URI hit counts.
CLI	**IOS and IOS XE:** **collect application http uri statistics**
Export Field ID	**IOS and IOS XE:** 42125
Export Protocol	IPFIX

On XE platform, the URI hits can be configured when the keys used are "match connection id" or "match connection transaction-id". Therefore only one URI is reported with hits=1.

URI and URI hit count are collected and exported using the following format:

uri <delimiter> count <delimiter> uri <delimiter> count <delimiter>...

NULL (\0) is the delimiter. Count is a 2-byte value and is encoded as an integer, while URI is encoded as a string.

Note The collected and exported URI is limited to the first '/'. For example, if the URL is http://www.cisco.com/router/isr/g2, the URI collected is '/router'.

A pattern is used to export the list of URIs and the corresponding hit counts. For example, if we have the following flows during the 5-minute window (src-ip 1.1.1.1, dest-ip 2.2.2.2. des-port80, protocol TCP):

- www.yahoo.com/music (5 flows)
- www.yahoo.com/video (3 flows)
- www.yahoo.com/data (10 flows)

The result will be exported as: 1.1.1.1, 2.2.2.2, 80, TCP, www.yahoo.com, /music:5/video:3/data:10.

The following are the various characteristics of a URI metric and a URI statistic metric:

- If the **collect app http host** command is configured in the flow record, the hostname www.yahoo.com will be exported.
- The delimiter is NULL (\0) URI, and the count is always represented in binary format using fixed length or 2 bytes. The delimiters colon (:) and double colon (::) are used here for demonstration purposes.
- The collector parses the URI through the basis of delimiters.
- A FNF export field (42125) is used to export the list of URIs and the corresponding hit counts. The Encoding will be done as follows:

  {URI\0countURI\0count}

NULL terminated string, followed by 2-byte hit count and so on. For example: [music\05video\03data\010].

- Multiple transactions are not supported for host extraction. If there are multiple transactions in the same TCP connection, NBAR will provide host information from only the first transaction.
- If the flow is terminated in this export interval, both the URI and hostname fields will be exported.
- The maximum length of one URI instance is 512 characters long. If a URI name exceeds 512 characters, it will be truncated to 512 bytes and trailing characters will be dropped and will not be exported.
- The URI hit count is a 2-byte variable. The maximum URI hit count is 65,535.
- If the HTTP URI statistic is configured for non-HTTP flows, this field will be exported as NULL.
- If there are multiple transactions in the same TCP connection, NBAR will provide URI information from only the first transaction. Multi-transaction scenario is not supported for URI extraction.
- AVC on IOS supports tunneled protocol.
- AVC on IOS provides hit count for NULL URIs.
- All the parameters appearing after "?" in the URI will be stripped off and not exported. For example, if the URI is "path?h:test", the URI that is exported will be "path" only; "h:test" will be stripped off.
- If the URI is index.html, it will be exported as a separate URI and will not be aggregated with first-level " " URI hit count.

HTTP Proxy

In case of HTTP Proxy, the hostname will be a part of the URI. For example, if the browser URL is 'http://www.cisco.com/go/avc', and if there is *no* HTTP proxy, the output will be:

Hostname: www.cisco.com

URI: 'go' [first-level URI]

If HTTP proxy is present, the NBAR output will be:

Host Name: www.cisco.com

URI: http://www.cisco.com/go [Hostname is part of the URI, and the URI is at first-level only. In this case, it is 'go']

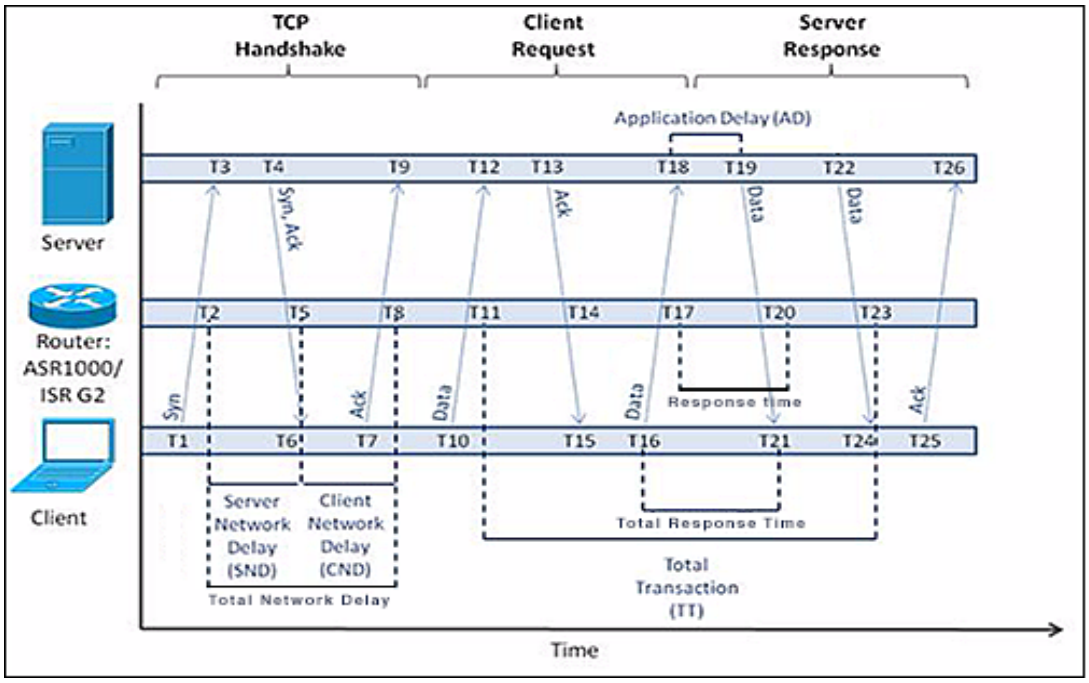
If HTTP proxy is not present, there will be one host name per 4-tuple flow. The hostname will be exported. For example, the hostname will be 'www.cisco.com' and URI will be 'go'.

Application Response Time Metrics

Application Response Time (ART) metrics are metrics extracted or calculated by the ART engine. These metrics are available only for TCP flows. ART client/server bytes and packets are for Layer 3 and Layer 4 measurements.

Figure 2-3 illustrates the TCP performance in the context of different types of delay and response time.

Figure 2-3 CP Diagram for Application Response Time Metrics



Some of the metrics are available only after certain protocol stages:

- Network time-related metrics are exported only after the TCP three-way handshake.
- Transaction time is measured and exported upon receiving either a new request from a client (which indicates the end of a current transaction) or the first FIN packet.
- Response time is measured and exported only after receiving the first response from the server.

In addition, the collector might occasionally observe zero values for TCP performance metric's active flows. Possible reasons are:

- The value of the metrics is zero according to the derived metrics calculation formula.
- Some delay metric could be less than 1 millisecond, which is the smallest granularity supported in TCP performance.

From the exported TCP Performance metrics, more metrics can be derived at the collector side.

Example:
Average Application Delay (AD) = Connection Delay Application Sum/ Connection Counter Server Responses

Table 2-5 lists the ART metrics summary.

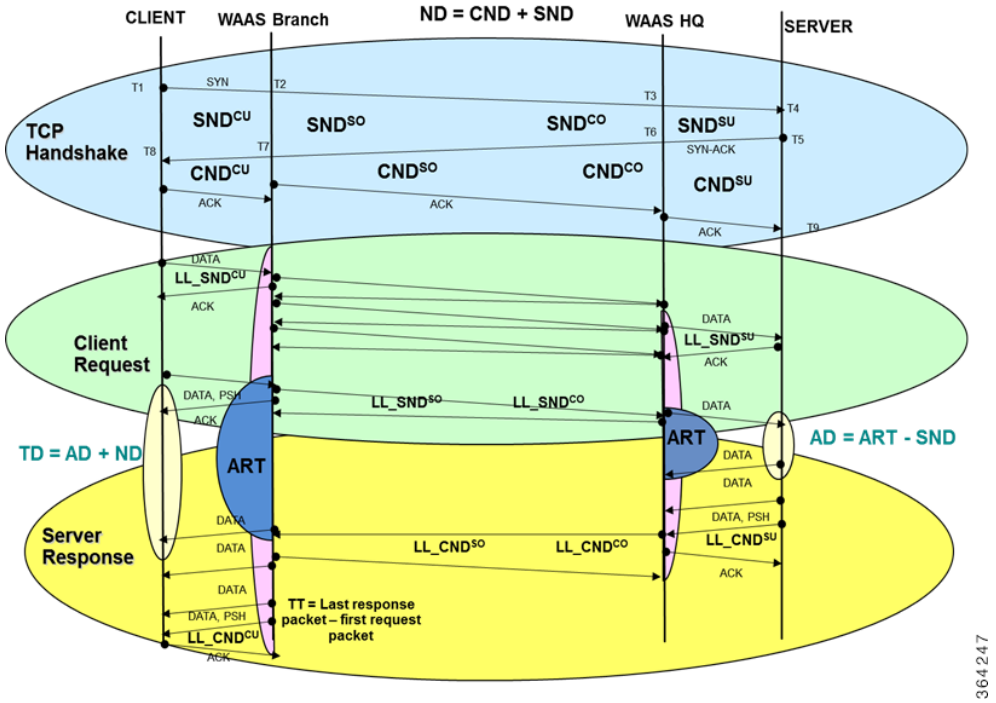
Table 2-5 ART Metrics Summary

Field Name	Field ID (IOS and IOS XE)
Client Network Time [sum/min/max]	42084(sum) 42085(max) 42086(min)
Long Lived Client Network Time [sum/min/max/num-samples]	42023(sum) 40285(max) 40286(min) 42027(num-samples)
Server Network Time [sum/min/max]	42087(sum) 42088(max) 42089(min)
Long Lived Server Network Time [sum/min/max/num-samples]	42022(sum) 40288(max) 40289(min) 42026(num-samples)
Network Time [sum/min/max]	42081(sum) 42082(max) 42083(min)
Long Lived Network Time [sum/min/max/num-samples]	42024(sum) 42082(max) 42083(min) 42025(num-samples)
Server Response Time [sum/min/max]	42074(sum) 42075(max) 42076(min)
Response Time [sum/min/max]	42071(sum) 42072(max) 42073(min)
Total Response Time [sum/min/max]	42077(sum) 42078(max) 42079(min)
Total Transaction Time [sum/min/max]	42041(sum) 42042(max) 42043(min)

Table 2-5 ART Metrics Summary (continued)

Field Name	Field ID (IOS and IOS XE)
ART Client Bytes/Packets(Layer 4)	231(octets) 298(packets)
ART Server Bytes/Packet(Layer 4)	232(octets) 299(packets)
ART Count New Connections	278
ART Concurrent Sessions	42018
ART Count Responses	42060
Responses Histogram Buckets (7-Bucket Histogram)	42061–42067
ART Count Late Responses	42068
ART Count Transactions	42040
ART Client Retransmissions Bytes	42035
ART Client Retransmissions Packets	42036
ART Server Retransmissions Bytes	42037
ART Server Retransmissions Packets	42038
ART Client Bytes(Layer 3)	41106 (octets)
ART Server Bytes(Layer 3)	41105 (octets)

Figure 2-4 Segment Diagram for Application Response Time Metrics



Client Network Time [sum/min/max]

Table 2-6 lists information about the Client Network Time metric.

Table 2-6 Client Network Time Metric

Description	Round trip between SYN-ACK and ACK. It is also called *Client Network Delay* (CND).
CLI	**IOS and IOS XE:** **collect connection delay network to-client sum** **collect connection delay network to-client minimum** **collect connection delay network to-client maximum**
Export Field ID	**IOS and IOS XE:** 40284(sum), 40285(max), 40286(min)
Export Protocol	NetFlow v9, IPFIX

In Figure 2-3, CND is calculated from T5 and T8. Therefore, CND = T8 - T5.

Table 2-7 lists information about the Client Network Time metric with WAAS.

Table 2-7 Client Network Time Metric with WAAS

CU Segment	CND^{CU} is the difference between SYN-ACK from server and ACK of the client.
SO Segment	CND^{SO} is the difference between SYN-ACK from server and ACK of the branch WAAS device observed at the branch router.
CO Segment	CND^{CO} is the difference between SYN-ACK from server and ACK of the branch WAAS device observed at the HQ router.
SU Segment	CND^{SU} is the difference between SYN-ACK from server and ACK of the branch WAAS device observed at the HQ router.

Note In case of WAAS, CND^{CU} can be used to calculate the CND. This will include CND + WAAS delay on both headend and branch. WAAS Delays on branch can be obtained by subtracting CND^{SO} from CND^{CU}. WAAS delay on HQ is CND^{CO} - CND^{SU}

Long Lived Client Network Time [sum/min/max/num-samples]

If you configure mum-sample for network delay, then Long Lived network delays are exported instead of regular Client Network Time.

Table 2-8 lists information about the Long Lived Client Network Time metric.

Table 2-8 Long Lived Client Network Time Metric

Description	Round trip between a Server DATA Packet and ACK packet. If the ACK or DATA packet is dropped, the long-lived delay sample will be ignored.
CLI	**IOS and IOS XE:** **collect connection delay network to-client num-samples** **collect connection delay network to-client sum** **collect connection delay network to-client minimum** **collect connection delay network to-client maximum**
Export Field ID	**IOS and IOS XE:** 42023(sum), 40285(max), 40286(min), 42027(num-samples) **Note** The sum export ID is different from the regular client network delay.
Export Protocol	NetFlow v9, IPFIX

Table 2-9 lists information about the Long Lived Client Network Time metric with WAAS.

Table 2-9 Long Lived Client Network Time Metric with WAAS

CU Segment	LL_CND^{CU} is the difference between a DATA Packet from branch WAAS and ACK of the client
SO Segment	LL_CND^{SO} is the difference between DATA Packet from HQ WAAS and ACK of the branch WAAS device observed at the branch router.
CO Segment	LL_CND^{CO} is the difference between DATA Packet from HQ WAAS and ACK of the branch WAAS device observed at the HQ router.
SU Segment	LL_CND^{SU} is the difference between DATA Packet from server and ACK of the HQ WAAS device observed at the HQ router.

Note In case of WAAS, LL_CND will be the CND of the segment. For example, in Segment CO or SO, the CND is between the two WAAS devices. While on Segment SU, the CND is between the WAAS device and the server. To obtain total LL_CND, sum the LL_CND^{CU} + (LL_CND^{CO} or LL_CND^{SO}) + LL_CND^{SU}.

Server Network Time [sum/min/max]

Table 2-10 lists information about the Server Network Time metric.

Table 2-10 Server Network Time Metric.

Description	Round-trip time between SYN and SYN-ACK. It is also called *Server Network Delay* (SND).
CLI	**IOS and IOS XE:** **collect connection delay network to-server sum** **collect connection delay network to-server minimum** **collect connection delay network to-server maximum**
Export Field ID	**IOS and IOS XE:** 42087(sum), 42088(max), 42089(min)
Export Protocol	NetFlow v9, IPFIX

In Figure 2-3, SND is calculated from T2 to T5. Therefore, SND = T5 - T2.

Table 2-11 lists information about the Server Network Time Metric with WAAS.

Table 2-11 Server Network Time Metric with WAAS

CU Segment	SND^{CU} is the difference between SYN from client and SYN-ACK of the server with WAAS branch and HQ network delay added observed at branch.
SO Segment	SND^{SO} is the difference between SYN from client and SYN-ACK of the server with WAAS HQ network delay added observed at branch.
CO Segment	SND^{CO} is the difference between SYN from client and SYN-ACK of the server with WAAS HQ network delay added observed at HQ.
SU Segment	SND^{SU} is the difference between SYN from client and SYN-ACK of the server observed at HQ.

Note Same as CND, SND can be used to obtain WAAS delays on both branch and headend. Subtracting SND^{CU} from SND^{SO} gives you the branch WAAS delay. Same applies for HQ WAAS.

Long Lived Server Network Time [sum/min/max/num-samples]

If you configure mum-sample for network delay, then Long Lived network server delays are exported instead of the regular Server Network Time.

Table 2-12 lists information about the Long Lived Server Network Time metric.

Table 2-12 Long Lived Server Network Time Metric.

Description	Round trip between a Client DATA Packet and ACK packet. If the ACK or DATA packet is dropped, the long-lived delay sample will be ignored..
CLI	**IOS and IOS XE:** **collect connection delay network to-server num-samples** **collect connection delay network to- server sum** **collect connection delay network to- server minimum** **collect connection delay network to- server maximum**
Export Field ID	**IOS and IOS XE:** 42022(sum), 40288(max), 40289(min), 42026(num-samples) **Note** The sum export ID is different from the regular client network delay.
Export Protocol	NetFlow v9, IPFIX

Table 2-13 lists information about the Server Network Time Metric with WAAS.

Table 2-13 Long Lived Server Network Time Metric with WAAS

CU Segment	SND^{CU} is the difference between SYN from client and SYN-ACK of the server with WAAS branch and HQ network delay added observed at branch.
SO Segment	SND^{SO} is the difference between SYN from client and SYN-ACK of the server with WAAS HQ network delay added observed at branch.
CO Segment	SND^{CO} is the difference between SYN from client and SYN-ACK of the server with WAAS HQ network delay added observed at HQ.
SU Segment	SND^{SU} is the difference between SYN from client and SYN-ACK of the server observed at HQ.

Note In case of WAAS, to obtain total LL_SND, sum the LL_SND^{CU} +(LL_SND^{CO} or LL_SND^{SO})+ LL_SND^{SU}.

Network Time [sum/min/max]

Table 2-14 lists the information about the Network Time metric.

Table 2-14 Network Time Metric

Description	Network Time is known as the round-trip time that is the summation of CND and SND. It is also called *Network Delay* (ND).
CLI	**IOS and IOS XE:** **collect connection delay network client-to-server sum** **collect connection delay network client-to-server minimum** **collect connection delay network client-to-server maximum**

Export Field ID	**IOS and IOS XE:** 42081(sum), 42082(max), 42083(min)
Export Protocol	NetFlow v9, IPFIX

In Figure 2-3, Network Delay is calculated from T2 to T8 (RT = T8 - T2). To get the value of ND, ND = SND + CND.

Note In case of WAAS, ND will depend on the value of SND and CND for the segment. ND = SND + CND.

Long Lived Network Time [sum/min/max/num-samples]

Table 2-14 lists the information about the Long Lived Network Time metric.

Table 2-15 Long Lived Network Time Metric

Description	Network Time is known as the round-trip time that is the summation of LL_CND and LL_SND. It is also called *Long Lived Network Delay* (LL_ND).
CLI	**IOS and IOS XE:** **collect connection delay network client-to-server num-samples** **collect connection delay network client-to-server sum** **collect connection delay network client-to-server minimum** **collect connection delay network client-to-server maximum**
Export Field ID	**IOS and IOS XE:** 42024(sum), 42082(max), 42083(min), 42025(num_samples)
Export Protocol	NetFlow v9, IPFIX

Note In case of WAAS, the ND will depend on the value of LL_SND and LL_CND for the segment.

LL_ND = LL_SND + LL_CND

Server Response Time [sum/min/max]

Table 2-16 lists information about the Server Response Time metric.

Table 2-16 Server Response Time Metric

Description	Time taken by an application to respond to a request. It is also called *Application Delay* (AD) or *Application Response Time*.
CLI	**IOS and IOS XE:** **collect connection delay application sum** **collect connection delay application minimum** **collect connection delay application maximum**
Export Field ID	**IOS and IOS XE:** 42074(sum), 42075(max), 42076(min)
Export Protocol	NetFlow v9, IPFIX

AD is calculated using the following formula:

Without WAAS:

AD = RT – SND

If LL_SND is configured, AD = RT – LL_SND

No valid LL_SND sample: IF RT <= LL_SND, AD = 0

With WAAS, see Table 2-17

Table 2-17 Server Response Time Metric with WAAS

CU Segment	If $RT^{CU} > SND^{CU}$, $AD^{CU} = RT^{CU} – SND^{CU}$ If LL_SND is configured: If $RT^{CU} < SND^{CU}$, $AD^{CU} = RT^{CU} – (LL\_SND^{CU} + LL\_SND^{SO})$ WAAS Local response: If $RT^{CU} < (LL\_SND^{CU} + LL\_SND^{SO})$, $AD^{CU} = RT^{CU}$
SO Segment	Invalid
CO Segment	Invalid
SU Segment	If $RT^{SU} > SND^{SU}$, $AD^{SU} = RT^{SU} – SND^{SU}$ If LL_SND is configured: If $RT^{SU} < SND^{SU}$, $AD^{SU} = RT^{SU} – (LL\_SND^{SU} + LL\_SND^{CO})$ WAAS Local response: If $RT^{SU} < (LL\_SND^{SU} + LL\_SND^{CO})$, $AD^{SU} = RT^{SU}$

Response Time [sum/min/max]

Table 2-18 lists information about the Response Time metric.

Table 2-18 Response Time Metric

Description	Amount of time between a client request and the first server response.
CLI	**IOS and IOS XE:** **collect connection delay response to-server sum** **collect connection delay response to-server minimum** **collect connection delay response to-server maximum**
Export Field ID	**IOS and IOS XE:** 40274(sum), 40275(max), 40276(min)
Export Protocol	NetFlow v9, IPFIX

A client request can contain multiple packets. In this case, use the last client packet received.

In case of WAAS, RT will be valid only for segment CU and SU.

Total Response Time [sum/min/max]

Table 2-19 lists information about the Total Response Time metric.

Table 2-19 Total Response Time Metric

Description	Total time taken from the moment a client sends a request until the first response packet from the server is delivered to the client. It is also known as *Total Delay* (TD).
CLI	**IOS and IOS XE:** **collect connection delay response client-to-server sum** **collect connection delay response client-to-server minimum** **collect connection delay response client-to-server maximum**
Export Field ID	**IOS and IOS XE:** 42077(sum), 42078(max), 42079(min)
Export Protocol	NetFlow v9, IPFIX

In Figure 2-3, TD = RT + CND. On XE and IOS, use the following formulae:

- min_totalDelay = min(RT+CND)
- max_totalDelay = max(RT+CND)
- sum_totalDelay = sum(RT+CND)

In case of WAAS, TD will be valid only for segment CU and SU.

Total Transaction Time [sum/min/max]

Table 2-20 lists information about the Total Transaction Time metric.

Table 2-20 Total Transaction Time Metric

Description	Amount of time between the client request and the final response packet from the server. It is measured and exported on receiving either a new request from a client (which indicates the end of the current transaction) or the first FIN packet.
CLI	**IOS and IOS XE:** **collect connection transaction duration sum** **collect connection transaction duration minimum** **collect connection transaction duration maximum**
Export Field ID	**IOS:** 40277(sum), 40278(max), 40279(min) **IOS XE:** 42041(sum), 42042(max), 42043(min)
Export Protocol	NetFlow v9, IPFIX

In case of WAAS, TT will be valid only for segment CU and SU.

ART Client Bytes/Packets(Layer 4)

Table 2-21 lists information about the ART Client Bytes/Packets metric.

Table 2-21 ART Client Bytes/Packets Metric

Description	Byte and packet count for all the client packets.
CLI	**IOS and IOS XE:** **collect connection client counter bytes long** **collect connection client counter packets long**
Export Field ID	**IOS and IOS XE:** 231(octet), 298(packets)
Export Protocol	NetFlow v9, IPFIX

L4 bytes and packets from client to server. Alternate commands for the same metric are **collect connection client counter bytes transport long** and **collect connection client counter packets transport long**.

In case of WAAS, each segment will reflect the client bytes/packets on the segment.

ART Server Bytes/Packet(Layer 4)

Table 2-22 lists information about the ART Server Bytes/Packets metrics.

Table 2-22 ART Server Bytes/Packets Metrics

Description	Byte and packet count for all the server packets.
CLI	**IOS and IOS XE:** **collect connection server counter bytes long** **collect connection server counter packets long**
Export Field ID	**IOS and IOS XE:** 232(octets), 299(packets)
Export Protocol	NetFlow v9, IPFIX

L4 bytes and packets from server to client. Alternate commands for the same metrics are **collect connection server counter bytes transport long** and **collect connection server counter packets transport long**.

In case of WAAS, each segment will reflect the client bytes/packets on the segment.

ART Client Bytes(Layer 3)

Table 2-22 lists information about the ART Client Bytes/Packets metric.

Table 2-23 ART Client Bytes Layer 3 Metric

Description	Byte and packet count for all the client packets(Layer 3).
CLI	**IOS and IOS XE:** **collect connection client counter bytes network long** **collect connection server counter bytes network long**
Export Field ID	**IOS and IOS XE:** 41106 (octets)
Export Protocol	NetFlow v9, IPFIX

ART Server Bytes(Layer 3)

Table 2-24 lists information about the ART Client Bytes/Packets metric.

Table 2-24 ART Server Bytes Layer 3 Metric

Description	Byte and packet count for all the server packets (Layer 3).
CLI	**IOS and IOS XE:** **collect connection client counter bytes network long** **collect connection server counter bytes network long**
Export Field ID	**IOS and IOS XE:** 41105 (octets)
Export Protocol	NetFlow v9, IPFIX

ART Count New Connections

Table 2-25 lists information about the ART Count New Connections metric.

Table 2-25 ART Count New Connections metric

Definition	Number of TCP sessions (3-way handshake) or UDP sessions established. It is also called *number of connections* (sessions).
CLI	**IOS and IOS XE:** **collect connection new-connections**
Export Field ID	**IOS and IOS XE:** 278
Export Protocol	NetFlow v9, IPFIX

ART Concurrent Sessions

Table 2-26 lists information about the ART Concurrent Sessions.

Table 2-26 ART Concurrent Sessions

Description	Number of active concurrent connections at the start of an export interval.
CLI	**IOS:** **collect connection concurrent-connections**
Export Field ID	**IOS:** 42018
Export Protocol	NetFlow v9, IPFIX

Concurrent session metrics is supported only in async/optimized mode. It is not supported in per-packet mode.

ART Count Responses

Table 2-27 lists information about the ART Count Responses metric.

Table 2-27 ART Count Responses Metric

Description	Number of Req-Rsp pair received within the monitoring interval.
CLI	**IOS and IOS XE:** **collect connection server counter responses**
Export Field ID	**IOS and IOS XE:** 42060
Export Protocol	NetFlow v9, IPFIX

In case of WAAS, Count response will be valid only for segment CU and SU and set zero for the CO, SO segments.

Responses Histogram Buckets (7-Bucket Histogram)

Table 2-28 lists information about the Responses Histogram Buckets (7-bucket histogram) metric.

Table 2-28 Responses Histogram Buckets Metric

Description	Number of responses received during the 7-bucket histogram response time.
CLI	**IOS and IOS XE:** **collect connection delay response to-server histogram**
Export Field ID	**IOS and IOS XE:** 42061–42067
Export Protocol	NetFlow v9, IPFIX

The following is the list of threshold values (response time) for the 7-buckets histogram:

- Bucket 1— Less than 2 milliseconds
- Bucket 2— Between 2 to 5 milliseconds
- Bucket 3— Between 5 to 10 milliseconds
- Bucket 4— Between 10 to 50 milliseconds
- Bucket 5— Between 50 to 100 milliseconds
- Bucket 6— Between 100 to 500 milliseconds
- Bucket 7— Between 500 to 1000 milliseconds

ART Count Late Responses

Table 2-29 lists information about the ART Count Late Responses metric.

Table 2-29 ART Count Late Responses Metric

Description	Number of responses received after the maximum response time. It is also called *Number of Late Responses* (timeouts). The current threshold of timeout is 1 second.
CLI	**IOS and IOS XE:** **collect connection delay response to-server histogram late**
Export Field ID	**IOS and IOS XE:** 42068
Export Protocol	NetFlow v9, IPFIX

In case of WAAS, Count response will be valid only for segment CU and SU and set zero for the CO, SO segments. See Figure 2-5 for a definition of the WAAS segments and information on how to incorporate these into AVC flow exports.

ART Count Transactions

Table 2-30 lists information about the ART Count Transactions metric.

Table 2-30 ART Count Transactions Metric

Description	Total number of transactions for all the TCP connections.
CLI	**IOS and IOS XE:** **collect connection transaction counter complete**
Export Field ID	**IOS and IOS XE:** 42040
Export Protocol	NetFlow v9, IPFIX

A new transaction is counted under one of the following conditions:

- Receiving a data packet from a client request while the previous packet state is server response.
- Receiving a client FIN packet while the previous packet state is server response.
- Receiving a server FIN packet while the previous packet state is server response.

In case of WAAS, the Count response will be valid only for segment CU and SU and set to zero for the CO, SO segments.

ART Client Retransmissions Bytes

Table 2-31 lists information about the ART Client Retransmissions Bytes metric.

Table 2-31 ART Client Retransmissions Bytes Metric

Description	ART Count Retransmissions metric is the byte count for all the retransmitted client packets.
CLI	**IOS and IOS XE:** **collect connection client counter bytes retransmitted**
Export Field ID	**IOS and IOS XE:** 42035
Export Protocol	NetFlow v9, IPFIX

ART Client Retransmissions Packets

Table 2-32 lists information about the ART Client Retransmissions Packets metric.

Table 2-32 ART Client Retransmissions Packets Metric

Description	ART Count Retransmissions metric is the packet count for all the retransmitted client packets.
CLI	**IOS and IOS XE:** **collect connection client counter packets retransmitted**
Export Field ID	**IOS and IOS XE:** 42036
Export Protocol	NetFlow v9, IPFIX

If the current packet's sequence number is same as the previous packet's sequence number, then it is a retransmitted packet. In case of WAAS, the retransmissions are counted per segment.

In case of WAAS, the retransmissions are counted per segment.

ART Server Retransmissions Bytes

Table 2-33 lists information about the ART Server Retransmissions Bytes metric.

Table 2-33 ART Server Retransmissions Bytes Metric

Description	ART Count Retransmissions metric is the bytes count for all the retransmitted server packets.
CLI	**IOS and IOS XE:** **collect connection server counter bytes retransmitted**
Export Field ID	**IOS and IOS XE:** 42037
Export Protocol	NetFlow v9, IPFIX

If the current packet's sequence number is same as the previous packet's sequence number, then it is a retransmitted packet. In case of WAAS, the retransmissions are counted per segment.

In case of WAAS, the retransmissions are counted per segment.

ART Server Retransmissions Packets

Table 2-34 lists information about the ART Server Retransmissions Packets metric.

Table 2-34 ART Server Retransmissions Packets Metric

Description	ART Count Retransmissions metric is the packet count for all the retransmitted client packets.
CLI	**IOS and IOS XE:** **collect connection server counter packets retransmitted**

Description	ART Count Retransmissions metric is the packet count for all the retransmitted client packets.
Export Field ID	**IOS and IOS XE:** 42038
Export Protocol	NetFlow v9, IPFIX

If the current packet's sequence number is same as the previous packet's sequence number, then it is a retransmitted packet. In case of WAAS, the retransmissions are counted per segment.

In case of WAAS, the retransmissions are counted per segment.

Client Bytes

Table 2-35 lists information about the Client Bytes metric.

Table 2-35 Client Bytes Metric

Description	Total L3 bytes sent by the initiator of a connection. Counted for TCP and UDP connections.
CLI	**IOS and IOS XE:** **collect connection client counter bytes network long**
Export Field ID	**IOS and IOS XE:** 41106
Export Protocol	NetFlow v9, IPFIX

ART All Metrics

Table 2-36 lists the information about the ART All Metric.

Table 2-36 ART All Metric

Description	Shortcut to enable all ART metrics using a single command.
CLI	**IOS:** **collect connection all**
Export Field ID	N/A
Export Protocol	N/A

Cisco WAAS Interoperation Metrics

Cisco Wide Area Application Services (WAAS) metrics are metrics, such as Data Redundancy Elimination (DRE) input bytes, that are extracted or calculated by the Cisco WAAS engine.

WAAS metrics are not available on the performance monitor record type on IOS.

Table 2-37 lists the WAAS metrics summary.

Table 2-37 WAAS Metrics Summary

Field Name	Field ID
WAAS Segment Number	42020
WAAS Passthrough Reason	42021
WAAS DRE Input	36000
WAAS DRE Output	36001
WAAS Lempel-Ziv Input	36002
WAAS Lempel-Ziv Output	36003
WAAS Input Bytes	36009
WAAS Output Bytes	36010
WAAS Connection Mode	36008
WAAS All Metrics	—

WAAS Segment Number

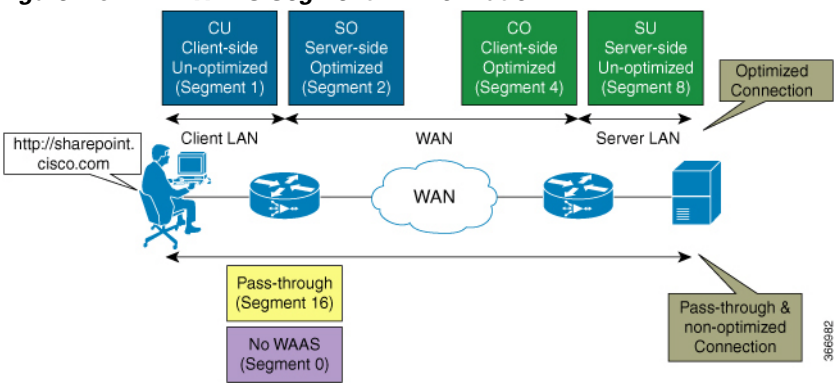
Table 2-38 lists information about the WAAS Segment Number metric.

Table 2-38 WAAS Segment Number Metric

Description	ID number of the WAAS segments.
CLI	**IOS and IOS XE:** **<collect | match> services waas segment**
Export Field ID	**IOS and IOS XE:** 42020
Export Protocol	NetFlow v9, IPFIX

Figure 2-5 shows the WAAS segment ID information.

Figure 2-5 WAAS Segment ID Information



The following are the four segments of WAAS connection in ISR:

- Segment ID 1—WAAS Client Unoptimized (CU)
- Segment ID 2—WAAS Server Optimized (SO)
- Segment ID 4—WAAS Client Optimized (CO)
- Segment ID 8—WAAS Server Unoptimized (SU)

If WAAS decides to pass through the flow, the segment ID is 16. If WAAS does not act on the flow, the segment is 0 (unknown). To receive the WAAS segment ID per flow, ensure that **match services waas segment account-on-resolution** is configured in the AVC flow record.

WAAS Passthrough Reason

Table 2-39 lists information about the Cisco WAAS Passthrough-reason Metrics.

Table 2-39 Cisco WAAS Passthrough-reason Metrics

Definition	Provides the reason if WAAS pass through a packet.
CLI	**IOS and IOS XE:** **collect services waas passthrough-reason**
Export Field ID	**IOS and IOS XE:** 42021
Export Protocol	NetFlow v9, IPFIX

WAAS DRE Input

Table 2-40 lists information about the Cisco WAAS DRE Input metric.

Table 2-40 Cisco WAAS DRE Input Metric

Description	Total input, measured in bytes, to the DRE engine for compression and decompression on a given WAAS segment.
CLI	**IOS and IOS XE:** **collect waas dre input**
Export Field ID	**IOS and IOS XE:** 36000
Export Protocol	NetFlow v9, IPFIX

WAAS DRE Output

Table 2-41 lists information about the Cisco WAAS DRE Output metric.

Table 2-41 Cisco WAAS DRE Output Metric

Description	Total output, measured in bytes, to the DRE engine for compression and decompression on a given WAAS segment.
CLI	**IOS and IOS XE:** **collect waas dre output**
Export Field ID	**IOS and IOS XE:** 36001
Export Protocol	NetFlow v9, IPFIX

WAAS Lempel-Ziv Input

Table 2-42 lists information about the Cisco WAAS Lempel-Ziv (LZ) Input metric.

Table 2-42 Cisco WAAS Lempel-Ziv (LZ) Input Metric

Description	Total input, measured in bytes, to the LZ engine for compression and decompression on a given WAAS segment.
CLI	**IOS and IOS XE:** **collect waas lz input**
Export Field ID	**IOS and IOS XE:** 36002
Export Protocol	NetFlow v9, IPFIX

WAAS Lempel-Ziv Output

Table 2-43 lists information about the Cisco WAAS Lempel-Ziv Output metric.

Table 2-43 Cisco WAAS Lempel-Ziv Output Metric

Description	Total output, measured in bytes, from the Lempel-Ziv engine for compression and decompression on a given WAAS segment.
CLI	**IOS and IOS XE:** **collect waas lz output**
Export Field ID	**IOS and IOS XE:** 36003
Export Protocol	NetFlow v9, IPFIX

WAAS Input Bytes

Table 2-44 lists information about the Cisco WAAS Input Bytes metric.

Table 2-44 Cisco WAAS Input Bytes Metric

Description	Total input, measured in bytes, to the WAAS module on a given WAAS segment.
CLI	**IOS and IOS XE** **collect waas bytes input**
Export Field ID	**IOS and IOS XE:** 36009
Export Protocol	NetFlow v9, IPFIX

WAAS Output Bytes

Table 2-45 lists information about the Cisco WAAS Output Bytes metric.

Table 2-45 Cisco WAAS Output Bytes Metric

Description	Total output, measured in bytes, from the WAAS module on a given WAAS segment.
CLI	**IOS and IOS XE:** **collect waas bytes output**
Export Field ID	**IOS and IOS XE:** 36010
Export Protocol	NetFlow v9, IPFIX

WAAS Connection Mode

Table 2-46 lists the information about the Cisco WAAS Connection Mode metric.

Table 2-46 Cisco WAAS Connection Mode Metric

Definition	Describes the optimization used on the connection.
CLI	**IOS and IOS XE:** **collect waas connection mode**
Export Field ID	**IOS and IOS XE:** 36008
Export Protocol	NetFlow v9, IPFIX

The Cisco WAAS Connection Mode is a bitmask with the following flags:

- Optimize TCP Flow Optimization (TFO)—0x1
- Optimize Data Redundancy Elimination (DRE)—0x2
- Optimize Lempel-Ziv (LZ)—0x4
- Accelerate HTTP—0x08
- Accelerate SSL—0x10

WAAS All Metrics

Table 2-47 lists information about the Cisco WAAS All Metrics.

Table 2-47 Cisco WAAS All Metrics

Definition	Collects all the WAAS-related metrics. This CLI works as a replacement for all the WAAS-related collect statements in a flow record.
CLI	**IOS and IOS XE:** **collect waas all**
Export Field ID	—
Export Protocol	—

QoS Metrics

Quality of Service (QoS) provides prioritization, shaping, and rate-limiting of traffic. High-priority, latency-sensitive traffic can be put into the priority queue. It can also guarantee minimal bandwidth available to an application or group of applications within a QoS traffic class.

For AVC, QoS class map statements allow matching on all the new NBAR2-supported applications and Layer 7 application fields or protocols, as well as on the NBAR2 attributes, which can co-exist with all other traditional QoS match attributes such as IP, subnet, and DSCP.

Table 2-48 lists the QoS metrics summary.

Table 2-48 QoS Metrics Summary

Field Name	Field ID (IOS XE)
QoS Policy Classification Hierarchy	41000
QoS Queue Drops	42129
Queue ID	42128

QoS Policy Classification Hierarchy

Table 2-49 lists information about the QoS Policy Classification Hierarchy.

Table 2-49 QoS Policy Classification Hierarchy

Definition	Identifies which hierarchy a QoS queue belongs to.
CLI	**IOS and IOS XE:** **<collect | match> policy qos classification hierarchy** (for QoS class) **<collect | match> policy performance-monitor classification hierarchy** (for perf-mon class)
Export Field ID	**IOS and IOS XE:** 41000
Export Protocol	NetFlow v9, IPFIX

To associate the QoS queue of a particular flow, AVC will export the hierarchy of the class the flow matches with. This hierarchy will be exported in the flow record as a list of IDs. Each ID will be in a separate FNF field. The value of the missing or unnecessary fields defaults to 0. The ID for name mapping will be exported as an option template.

The following example shows the configuration for the basic QoS hierarchy export. The example shows the QoS configuration for parent policy P1 and child policy P11.

```
class-map match-all C1
 match any
class-map match-all C11
 match ip dscp ef
class-map match-all C12
 match ip dscp cs2
!
policy-map P11
 class C11
  bandwidth remaining percent 10
class C12
  bandwidth remaining percent 70
class class-default
  bandwidth remaining percent 20

policy-map P1
 class C1
  shaping average 16000000
  service-policy P11
```

Table 2-50 shows a sample mapping table.

The class hierarchy shows hierarchy information up to 5 class level. Each of these ID is a 4-byte integer representing a C3PL policy-map or class-map. The ID to name mapping will be exported as an option template.

Table 2-50 Sample Mapping Table

Flow ID	Class Hierarchy (41000)	Queue id (42128)
Flow 1	P1, C1, C11, 0, 0, 0	1
Flow 2	P1, C1, C11, 0, 0, 0	1
Flow 3	P1, C1, C12, 0, 0, 0	2

The queue id for a particular class hierarchy will be exported using export field ID 42128.

Two option templates are used to export the class and policy information. The first template is for class ID and class name mapping, and the second template is for policy ID and policy name mapping. The configuration example and the information the option template contains are shown below.

Example:

```
flow exporter my-exporter
    option c3pl-class-table
    option c3pl-policy-table

QoS Class ID Export
  Client: Option classmap option table
  Exporter Format: NetFlow Version 9
  Template ID    : 263
  Source ID      : 0
  Record Size    : 304
  Template layout
  ____________________________________________________________________________
  |                 Field                    |  Type | Offset |  Size  |
  ----------------------------------------------------------------------------
  | v9-scope system                          |     1 |      0 |      4 |
  | c3pl class cce-id                        | 41001 |      4 |      4 |
  | c3pl class name                          | 41002 |      8 |     40 |
  | c3pl class type                          | 41003 |     48 |    256 |
  ----------------------------------------------------------------------------
  Client: Option policymap option table
  Exporter Format: NetFlow Version 9
  Template ID    : 264
  Source ID      : 0
  Record Size    : 304
  Template layout
  ____________________________________________________________________________
  |                 Field                    |  Type | Offset |  Size  |
  ----------------------------------------------------------------------------
  | v9-scope system                          |     1 |      0 |      4 |
  | c3pl policy cce-id                       | 41004 |      4 |      4 |
  | c3pl policy name                         | 41005 |      8 |     40 |
  | c3pl policy type                         | 41006 |     48 |    256 |
  ----------------------------------------------------------------------------
```

QoS Queue Drops

Table 2-51 lists information about the QoS Queue Drops.

Table 2-51 QoS Queue Drops

Definition	Exports two types of tables. The first table contains data pertaining to each flow and the second table captures the data when the TCP Performance timer expires.
CLI	**IOS and IOS XE:** **collect policy qos queue drops**
Export Field ID	**IOS and IOS XE:** 42129
Export Protocol	NetFlow v9, IPFIX

Media Performance Metrics

Table 2-52 lists all the media monitoring-related fields.

Table 2-52 Media Monitoring-Related Fields

Field Name	Description	Field ID (IOS and IOS XE)
[collect | match] transport rtp ssrc	RTP SSRC.	37022
collect transport rtp payload-type	RTP payload type.	37041
collect transport rtp jitter minimum	Minimum jitter for the RTP stream.	37024
collect transport rtp jitter maximum	Maximum jitter for the RTP stream.	37025
collect transport packets lost counter	A count of the number of lost packets from sequencing information.	37019
collect transport packets expected counter	Expected number of packets from sequencing information.	37014
collect transport event packet-loss counter	A count of sets of packets that were lost.	37017
collect counter packets dropped	A count of the packets dropped.	37000
collect application media bytes counter	A count of the number of packets with a media payload.	37004
collect application media bytes rate	Byte rate for the media stream.	37006
collect application media packets counter	A count of the number of packets with a media payload.	37007
collect application media packets rate	Packet rate for the media stream.	37009
collect application media event	Flags indicating media events.	37011
collect monitor event	Flags indicating monitor events.	37012
The following fields require records to be punted to the route processor (RP).		
collect counter flows	Total number of flows.	3
collect transport rtp flow count	Number of RTP flows.	37040
collect application media packets rate variation	Variation in packet rate from configured expected rate.	37010
collect application media packets rate variation minimum	Minimum variation in packet rate from configured expected rate.	37038
collect application media packets rate variation maximum	Maximum variation in packet rate from configured expected rate.	37039
collect application media event	Flags indicating media events.	37011
collect monitor event	Flags indicating monitor events.	37012
collect transport rtp jitter mean	Mean jitter for the RTP stream.	37023
collect transport packets lost rate	Packet loss rate from sequencing information.	37021

Table 2-52 Media Monitoring-Related Fields (continued)

Field Name	Description	Field ID (IOS and IOS XE)
collect transport packets lost rate minimum	Minimum packet loss rate in the aggregated flows.	37047
collect transport packets lost rate maximum	Maximum packet loss rate in the aggregated flows.	37048
collect application media bytes rate	Byte rate for the media stream.	37006
collect application media packets rate	Packet rate for the media stream.	37009
The following fields are metadata-related fields.		
collect application version	Application version id.	105
collect application version name	Application name.	106
collect application vendor	Application vendor-id.	107
collect metadata global-session-id	Metadata global-session-id.	37054
collect metadata multi-party-session-id	Metadata multi-party-session-id.	37055
collect metadata clock-rate	Metadata clock-rate.	37056

General Metrics

Absolute Timestamp

Table 2-53 lists information about the Absolute Timestamp.

Table 2-53 Absolute Timestamp

Definition	Absolute timestamp of the first packet and the last packet of the flow.
CLI	**IOS and IOS XE:** **collect timestamp absolute first** **collect timestamp absolute last**
Export Field ID	**IOS and IOS XE:** 152 (first) 153 (last)
Export Protocol	NetFlow v9, IPFIX

Option Template

A flow record exported to a mapping table is called an *option template*. The following is an example of the CLI:

```
flow exporter my-export
    export-protocol ipfix
    template data timeout <timeout>
    option interface-table timeout <timeout>
    option vrf-table timeout <timeout>
    option sampler-table timeout <timeout>
    option application-table timeout <timeout>
    option application-attributes timeout <timeout>
    option sub-application-table timeout <timeout>
```

Traffic Volume

Traffic volume fields are similar to the FNF fields. For more information, see: http://www.cisco.com/en/US/technologies/tk648/tk362/technologies_white_paper09186a00800a3db9.html.

Field ID Comparison

Cisco ISR G2 and Cisco ASR 1000 should export compatible data. The following are the differences in the solution due to the architectural dissimilarities:

- URL export—In IOS XE platforms, URLs are exported per transaction, while in IOS the URLs are exported as a concatenated field over 4-tuple.

AVC Configuration Examples and Troubleshooting Tips

First Published: March 29, 2013
Revised: March 26, 2015

For configuration examples and troubleshooting tips, see *Cisco Application Visibility and Control User Guide*.

Troubleshooting Tips and Debug Commands for IOS Platform

First Published: March 29, 2013
Revised: March 26, 2015

Troubleshooting Tips

The following tips are for IOS platform. For troubleshooting information on IOS XE platform, see *Cisco Application Visibility and Control User Guide.*

- Whenever there is an issue, turn on the relative debugs and search for keywords such as *Error* or *Fail*. Such a search will usually provide useful information about further debugging.
- Use the **show policy-map type mace interface** command if you want to know the number of Layer 4 flows that match the MACE policy. This is commonly used to check the basic statistics because MACE is a flow-based feature.
- Occasionally the user may observe zero values in the metrics. These are as expected if the flow is still active. Check whether the flow is terminated or not.
- Some of the metrics are only available after certain protocol stages. For example, the HTTP Host and URI metrics are only available after the flow is terminated.
- Some of the metrics support only IPFIX export protocol. These include the HTTP and URI metrics. Be sure to choose the right export protocol under the flow exporter configuration.
- Use the **show flow export template** command to get the properties of the exported flow data.
- Use MACE show commands to view the current export data on the router. These commands are very useful for basic debugging.
- If you encounter issues on the collector, the user can get the hex dump of the packets on the collector side and manually check if there is anything that does not conform to the NetFlow v9 or IPFIX protocol.

Debug and Show Commands

Table 4-1 lists the debug and show commands supported on PA.

Table 4-1 Debug and Show Commands Supported on PA

Command	Description
debug mace cp	PA Control-Plane debugging.
debug mace dp	PA Data-Plane debugging.
debug mace art	PA ART Engine debugging.
debug flow monitor type performance-monitor [*name* [cache[raw]]]	Turn on flow monitor debugging.
debug flow record type performance-monitor	Turn on flow record debugging.
debug policy-map type performance-monitor [*name* | interface]	Turn on policy-map debugging.
debug ip nbar trace detail *access-list name*	Turn on NBAR debugging.
show ip nbar trace detail	Show NBAR debugging.
clear ip nbar trace detail	Clear NBAR debugging.
show policy-map type mace	Shows mace policy-map configuration.
show flow exporter *name*	Shows flow exporter configuration.
show flow record type mace *name*	Shows flow record configuration.
show flow monitor type mace *name*	Shows flow monitor configuration.
show flow exporter template	Shows the size, ID , and so on of the flow data that is exported to collector.
show cef interface *name*	Shows whether the PA or NBAR is enabled on the interface or not.
show policy-map type mace interface *name*	Shows whether the traffic flow matches the class-map or not.
show mace metrics [*src ip dest ip dest port protocol flow-mon* [art | waas]]	Shows the metrics that are collected at the last export timeout. If you do not specify any argument (src ip, dest ip, and so on), the command output will show metrics for all flows. Not all configured flows are shown in the output. Only a fixed subset of the metrics is shown.
show mace metrics 1.1.1.1	Shows metrics for flows only from source IP 1.1.1.1.
show mace metrics any 2.2.2.2	Shows metrics from any source IP, and destination IP with IP address 2.2.2.2.
show mace metrics any any 80	Shows metrics from any source IP, any destination IP, and destination port with port number 80.
show mace metrics any any any 6	Shows metrics from any source IP, any destination IP, any destination port, and protocol with protocol ID 6.

Command	Description
show mace metrics flow-mon1	Shows those metrics on which flow-mon1 action is taken. Note that the metrics shown will not have a one-to-one correspondence with those configured in the record used inside flow-mon1.
show mace metrics art	Shows only ART metrics.
show mace metrics waas	Shows only WAAS metrics.
show mace metrics summary	Shows summary of metrics.
show flow exporter statistics	Shows the number of records, bytes, and packets exported.
show flow exporter option application table	Shows all the application ID and the name mapping.
show ip nbar parameter extraction	Shows the NBAR field extraction format which concatenates the App ID, Sub-classification ID, and Value.

AVC-Related Exported Fields

First Published: March 29, 2013
Revised: March 26, 2015

Table A-1 describes the Flexible NetFlow (FNF) fields and the CLI used to retrieve the value of the fields, added for Cisco AVC.

In addition to the these new fields, an AVC record can include FNF fields defined prior to IOS XE 3.8. For information about FNF fields, see: *Cisco IOS Flexible NetFlow Command Reference*.

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
56	sourceMacAddress	No	macAddress	identifier	IEEE 802 source MAC address field. This field is collected only for a monitor attached in the ingress direction.	MAC	<collect | match> datalink mac source address input
57	postDestinationMac Address	No	macAddress	identifier	The definition of this information element is identical to the definition of information element "destinationMacAddress," except that it reports a potentially modified value caused by a middlebox function after the packet has passed the observation point.	MAC	<collect | match> datalink mac destination address output
58	vlanId	No	unsigned16	identifier	IEEE 802.1Q VLAN identifier (VID) extracted from the tag control Information field that was attached to the IP packet. This field is collected only for a monitor attached in the ingress direction.	number	<collect | match> datalink source-vlan-id

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
59	postVlanId	No	unsigned16	identifier	The definition of this information element is identical to the definition of information element “vlanId,” except that it reports a potentially modified value caused by a middlebox function after the packet has passed the observation point.	number	<collect | match> datalink destination-vlan-id
80	destinationMacAddress	No	macAddress	identifier	IEEE 802 destination MAC address field. This field is collected only for a monitor attached in the ingress direction.	MAC	<collect | match> datalink mac destination address input
81	postSourceMacAddress	No	macAddress	identifier	The definition of this information element is identical to the definition of information element “sourceMacAddress,” except that it reports a potentially modified value caused by a middlebox function after the packet has passed the observation point.	MAC	<collect | match> datalink mac source address output

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
138	observationPointId	No	unsigned64	identifier	An identifier of an observation point that is unique for each observation domain. It is recommended that this identifier be unique for each IPFIX device. Typically, this information element is used for limiting the scope of other information elements. The field contains 8 bytes: The 4 most significant bytes (MSBs) indicate the type. Currently, only type 1 is supported. When type is "1," the 4 least significant bytes (LSBs) indicate the interface SNMP index, which is also listed in the interface option template. **Example**: Observation Point Id: 4294967309 = 0x000000010000000D The 4 MSBs indicate a type of 1. The 4 LSBs indicate that the SNMP index for the interface is 0xD.	number	<collect | match> flow observation point
209	tcpOptions	No	unsigned64	flags	TCP options in packets of this flow. The information is encoded in a set of bit fields. For each TCP option, there is a bit in this set. The bit is set to 1 if any observed packet of this flow contains the corresponding TCP option. Otherwise, if no observed packet of this flow contained the respective TCP option, the value of the corresponding bit is 0.	bitmap	collect transport tcp option map
231	initiatorOctets	No	unsigned64	identifier	Total number of layer 4 payload bytes in a flow from the initiator. The initiator is the device that triggered the session creation, and remains the same for the life of the session.	octets	collect counter initiator bytes long

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
232	serverOctets	No	unsigned64	identifier	Total number of layer 4 payload bytes in a flow from the server. The server is the device that replies to the client, and remains the same for the life of the session.	octets	collect connection server counter bytes long
235	egressVRFID	No	unsigned32	identifier	Unique identifier of the VRF name where the packets of this flow are being sent. This identifier is unique per metering process.	number	<collect | match> routing vrf output
239	biflowDirection	No	unsigned8	identifier	A description of the direction assignment method used to assign the Biflow Source and Destination. This Information Element may be present in a Flow Data Record, or applied to all flows exported from an Exporting Process or Observation Domain using IPFIX Options. If this Information Element is not present in a Flow Record or associated with a Biflow via scope, it is assumed that the configuration of the direction assignment method is done out-of-band. Note that when using IPFIX Options to apply this Information Element to all flows within an Observation Domain or from an Exporting Process, the Option SHOULD be sent reliably. If reliable transport is not available (i.e., when using UDP), this Information Element SHOULD appear in each Flow Record.		collect connection initiator
278	connectionCountNew	No	unsigned32	deltaCounter	This information element counts the number of TCP or UDP connections which were opened during the observation period. The observation period may be specified by the flow start and end timestamps.	number	collect connection new-connections

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
279	connectionSumDurationSeconds	No	unsigned64		This information element aggregates the total time in seconds for all of the TCP or UDP connections which were in use during the observation period. For example if there are 5 concurrent connections each for 10 seconds, the value would be 50 s.	seconds	collect connection sum-duration
280	connectionTransactionId	No	unsigned64	identifier	Identifies a transaction within a connection. A transaction is a meaningful exchange of application data between two network devices or a client and server. A transactionId is assigned the first time a flow is reported, so that later reports for the same flow will have the same transactionId. A different transactionId is used for each transaction within a TCP or UDP connection. The identifiers need not be sequential.	number	match connection transaction-id
298	clientPackets	No	unsigned64	identifier	Total number of layer 4 packets in a flow from the client. The client is the device that triggered the session creation, and remains the same for the life of the session.	packets	collect connection client counter packets long
299	serverPackets	No	unsigned64	identifier	Total number of layer 4 packets in a flow from the server. The server is the device that replies to the client, and remains the same for the life of the session.	packets	collect connection server counter packets long
359	monitoringIntervalStartMilliSeconds	Yes	dateTimeMilliseconds		The absolute timestamp at which a monitoring interval starts. A monitoring interval is the period during which the metering Process is running.	milliseconds	match timestamp absolute monitoring-interval start
37083	tcpWindowSizeMin	Yes	unsigned32	identifier	Minimum TCP window size.	octets	collect transport tcp window-size minimum
37084	tcpWindowSizeMax	Yes	unsigned32	identifier	Maximum TCP window size.	octets	collect transport tcp window-size maximum

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
37086	tcpMaximumSegmentSize	Yes	unsigned16	identifier	TCP maximum segment size.	octets	collect transport tcp maximum-segment-size
37092	tcpWindowSizeSum	Yes	unsigned64	identifier	Sum of TCP window size values. Divide by packet counter to get average.	octets	collect transport tcp window-size sum
42036	retransPackets	Yes	unsigned32	deltaCounter	Number of packets retransmitted by the client	packets	collect connection client counter packets retransmitted
42040	transactionCountDelta	Yes	unsigned32	deltaCounter	Total number of completed transactions observed for this flow.		collect connection transaction counter complete
42041	sumTransactionTime	Yes	unsigned32	Duration	Transaction time is the time between the client request and the corresponding last response packet from the server, as observed at the observation point. The value is the sum of all transaction times observed for this flow. For the average, this field must be divided by transactionCountDelta (42040).	milliseconds	collect connection transaction duration sum
42042	maxTransactionTime	Yes	unsigned32	Duration	Maximum transaction time observed for this flow.	milliseconds	collect connection transaction duration max
42043	minTransactionTime	Yes	unsigned32	Duration	Minimum transaction time observed for this flow.	milliseconds	collect connection transaction duration min
42060	numRespsCountDelta	Yes	unsigned32	deltaCounter	Total number of responses sent by the server.	responses	collect connection server counter responses
42061	numResps1CountDelta	Yes	unsigned32	deltaCounter	Histogram Bucket 1 for response time. The bucket boundary should be specified in an option template or predefined in the reporting entity.	responses	collect connection delay response to-server histogram
42062	numResps2CountDelta	Yes	unsigned32	deltaCounter	Histogram Bucket 2 for response time. The bucket boundary should be specified in an option template or predefined in the reporting entity.	responses	collect connection delay response to-server histogram

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
42063	numResps3CountDelta	Yes	unsigned32	deltaCounter	Histogram Bucket 3 for response time. The bucket boundary should be specified in an option template or predefined in the reporting entity.	responses	collect connection delay response to-server histogram
42064	numResps4CountDelta	Yes	unsigned32	deltaCounter	Histogram Bucket 4 for response time. The bucket boundary should be specified in an option template or predefined in the reporting entity.	responses	collect connection delay response to-server histogram
42065	numResps5CountDelta	Yes	unsigned32	deltaCounter	Histogram Bucket 5 for response time. The bucket boundary should be specified in an option template or predefined in the reporting entity.	responses	collect connection delay response to-server histogram
42066	numResps6CountDelta	Yes	unsigned32	deltaCounter	Histogram Bucket 6 for response time. The bucket boundary should be specified in an option template or predefined in the reporting entity.	responses	collect connection delay response to-server histogram
42067	numResps7CountDelta	Yes	unsigned32	deltaCounter	Histogram Bucket 7 for response time. The bucket boundary should be specified in an option template or predefined in the reporting entity.	responses	collect connection delay response to-server histogram
42068	numLateRespsCountDelta	Yes	unsigned32	deltaCounter	Total number of late responses sent by the server. A late response is a response whose time is greater than the last bucket. This informational element can be treated as the last bucket that has no end limit.	responses	collect connection delay response to-server histogram
42071	sumRespTime	Yes	unsigned32	Delay	Response time is the time between the client request and the corresponding first response packet from the server, as observed at the observation point. The value of this information element is the sum of all response times observed for the responses of this flow. For the average, this field must be divided by numRespsCountDelta (42060).	milliseconds	collect connection delay response to-server sum
42072	maxRespTime	Yes	unsigned32	Delay	Maximum response time observed for this flow.	milliseconds	collect connection delay response to-server max

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
42073	minRespTime	Yes	unsigned32	Delay	Minimum response time observed for this flow.	milliseconds	collect connection delay response to-server min
42074	sumServerRespTime	Yes	unsigned32	Delay	Yes	milliseconds	collect connection delay application sum
42075	maxServerRespTime	Yes	unsigned32	Delay	Maximum application delay observed for the responses of this flow.	milliseconds	collect connection delay application max
42076	minServerRespTime	Yes	unsigned32	Delay	Minimum application delay observed for the responses of this flow.	milliseconds	collect connection delay application min
42077	sumTotalRespTime	Yes	unsigned32	Delay	Total delay is the time between the client request and the first response packet from the server, as seen by the client. This is the sum of all total delays observed for the responses of this flow. For the average, this field must be divided by numRespsCountDelta (42060)	milliseconds	collect connection delay response client-to-server sum
42078	maxTotalRespTime	Yes	unsigned32	Delay	Maximum total delay observed for the responses of this flow.	milliseconds	collect connection delay response client-to-server max
42079	minTotalRespTime	Yes	unsigned32	Delay	Minimum total delay observed for the responses of this flow.	milliseconds	collect connection delay response client-to-server min
42081	sumNwkTime	Yes	unsigned32	Delay	Network delay is the round-trip time between the client and the server, as measured by the observation point, calculated once per session. The value of this information element is the sum of all network delays observed for the sessions of this flow. For the average, this field must be divided by connectionCountNew (278).	milliseconds	collect connection delay network client-to-server sum
42082	maxNwkTime	Yes	unsigned32	Delay	Yes	milliseconds	collect connection delay network client-to-server max

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
42083	minNwkTime	Yes	unsigned32	Delay	Yes	milliseconds	collect connection delay network client-to-server min
42084	sumClientNwkTime	Yes	unsigned32	Delay	Client network delay is the round-trip time between the observation point and the client, calculated once per session. The value of this information element is the sum of all client network delays observed for the sessions of this flow. For the average, this field must be divided by connectionCountNew (278).	milliseconds	collect connection delay network to-client sum
42085	maxClientNwkTime	Yes	unsigned32	Delay	Maximum client network delay observed for the sessions of this flow.	milliseconds	collect connection delay network to-client max
42086	minClientNwkTime	Yes	unsigned32	Delay	Minimum client network delay observed for the sessions of this flow.	milliseconds	collect connection delay network to-client min
42087	sumServerNwkTime	Yes	unsigned32	Delay	Server network delay is the round-trip time between the observation point and the server, calculated once per session. The value of this information element is the sum of all server network delays observed for the sessions of this flow. For the average, this field must be divided by connectionCountNew (278)	milliseconds	collect connection delay network to-server sum
42088	maxServerNwkTime	Yes	unsigned32	Delay	Maximum server network delay observed for the sessions of this flow.	milliseconds	collect connection delay network to-server max
42089	minServerNwkTime	Yes	unsigned32	Delay	Minimum server network delay observed for the sessions of this flow.	milliseconds	collect connection delay network to-server min

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
45004	clientIPv4Address	Yes	ipv4Address	identifier	The IPv4 client address in the IP packet header. This may be the source or destination IP address, depending on the first packet of the connection. The client is the device that triggered the session creation, and remains the same for the life of the session.	address	<collect | match> client ipv4 address
45005	serverIPv4Address	Yes	ipv4Address	identifier	The IPv4 server address in the IP packet header. The server is the device that replies to the client, and remains the same for the life of the session.	address	<collect | match> server ipv4 address
45006	clientIPv6Address	Yes	ipv6Address	identifier	The IPv6 client address in the IP packet header. The client is the device that triggered the session creation, and remains the same for the life of the session.	address	<collect | match> client ipv6 address
45007	serverIPv6Address	Yes	ipv6address	identifier	IPv6 server address in the IP packer header. The server is the device that replies to the client, and remains the same for the life of the session.	address	<collect | match> server ipv6 address
45008	clientTransportPort	Yes	unsigned16	identifier	Client transport port identifier. This may be the source or destination transport port. The client is the device that triggered the session creation, and remains the same for the life of the session.	number	<collect | match> client transport port
45009	serverTransportPort	Yes	unsigned16	identifier	Server transport port identifier. This may be the source or destination transport port. The server is the device that replies to the client, and remains the same for the life of the session.	number	<collect | match> server transport port
41000	classHierarchy	Yes	Var-Len	identifier	Identifies the policy-map hierarchy for different policy-map types. The field contains the policy-id, followed by a list of classes representing the policy hierarchy: {Pi | Ck … Cl}. A dedicated option template contains the policy and class id mapping to name and type.	number	<collect | match> policy performance-monitor classification hierarchy

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
42020	servicesWaasSegment	Yes	unsigned8	identifier	WAAS optimization "segment" can have one of the following values:	number	<collect | match> services waas segment
					Unknown	0	
					Client Unoptimized	1	
					Server Optimized	2	
					Client Optimized	4	
					Server Unoptimized	8	
					Pass-Through	16	
42021	servicesWaasPassThroughReason	Yes	unsigned8	identifier	WAAS optimization pass-through reason can have one of the following values:	number	collect services waas passthrough-reason
					PT_NO_PEER	1	
					PT_RJCT_CAP	2	
					PT_RJCT_RSRCS	3	
					PT_RJCT_NO_LICENSE	4	
					PT_APP_CONFIG	5	
					PT_GLB_CONFIG	6	
					PT_ASYMMETRIC	7	
					PT_IN_PROGRESS	8	
					PT_INTERMEDIATE	9	

Table A-1 New FNF Exported Fields

Field ID	Name	Enterprise Specific	Data Type	Data Type Semantics	Description	Units	CLI
					PT_OVERLOAD	10	
					PT_INT_ERROR	11	
					PT_APP_OVERRIDE	12	
					PT_SVR_BLACKLIST	13	
					PT_AD_VER_MISMTCH	14	
					PT_AD_AO_INCOMPAT	15	
					PT_AD_AOIM_PROGRESS	16	
					PT_DIRM_VER_MISMTCH	17	
					PT_PEER_OVERRIDE	18	
					PT_AD_OPT_PARSE_FAIL	19	
					PT_AD_PT_SERIAL_-MODE	20	
					PT_SN_INTERCEP-TION_ACL	21	
					PT_IP_FRAG_UNSUP-P_PEER	22	
					PT_CLUSTER_MEM-BER_INDX	23	
					PT_FLOW_QUERY_-FAIL_INDX	24	
					PT_FLOWSW_INT_A-CL_DENY_INX	25	
					PT_UNKNOWN_INDX	26	
					PT_FLOWSW_PL-CY_INDX	27	
					PT_SNG_OVER-LOAD_INDX	28	
					PT_CLUSTER_DE-GRADE_INDX	29	
					PT_FLOW_LEARN_-FAIL_INDX	30	
					PT_OVERALL_INDX	31	
					PT_ZBFW	32	
					PT_RTSP_ALG	33	
42128	policyQosQueueID	No	unsigned32	identifier	QoS Policy queue ID	number	match policy qos queue id
42129	policyQosQueueDrop	No	unsigned64	counter	QoS Policy drops per queue	number	collect policy qos queue drops
45010	connectionId	Yes	unsigned32	identifier	Identifies a connection. A connection identifier is created when a new TCP or UDP flow is created between server and client. A single connection can hold several transactions.	number	match connection id

DPI/L7 Extracted Fields

First Published: March 29, 2013
Revised: March 26, 2015

Table B-1 describes deep packet inspection (DPI)/L7 extracted fields and the CLI used to retrieve the value of the fields.

Table B-1 AVC DPI/L7 Extracted Fields

Field Name	Release	Protocol Pack	Type	Application ID EngID	Application ID Sel ID	Sub Application ID	Description	Data Source	CLI
sslCommonName	3.9	7.0	String	13	453	13313	Common name extracted from a SSL certificate.	NBAR	collect application ssl common-name
httpUrl	3.7	—	String	3	80	13313	URL extracted from the HTTP transaction. The URL is required per transaction.	NBAR	collect application http url
httpHostName	3.7	—	String	3	80	13314	Host Name extracted from the HTTP transaction. The URL is required per transaction.	NBAR	collect application http host
httpUserAgent	3.7	—	String	3	80	13315	User agent field extracted from the HTTP transaction.	NBAR	collect application http user-agent
httpReferer	3.7	—	String	3	80	13316	REFERER extracted from the HTTP transaction.	NBAR	collect application http referer
rtspHostName	3.7	—	String	3	554	13313	RTSP host name extracted from the RTSP transaction.	NBAR	collect application rtsp host-name
smtpServer	3.7	—	String	3	25	13313	Server name extracted from an SMTP transaction.	NBAR	collect application smtp server
smtpSender	3.7	—	String	3	25	13314	Sender name extracted from an SMTP transaction.	NBAR	collect application smtp sender
pop3Server	3.7	—	String	3	110	13313	Server name extracted from a POP3 transaction.	NBAR	collect application pop3 server
nntpGroupName	3.7	—	String	3	119	13313	Group name extracted from an NNTP transaction.	NBAR	collect application nntp group-name
sipSrcDomain	3.7	—	String	3	5060	13314	Source domain extracted from a SIP transaction.	NBAR	collect application sip source
sipDstDomain	3.7	—	String	3	5060	13313	Destination domain extracted from a SIP transaction.	NBAR	collect application sip destination

Notes

- Beginning with IOS XE release 3.7, the fields are exported using the field subApplicationValue (ID=45003). The field is encoded as {**applicationID** (4B), **subApplicationID** (2B), Value (Variable Len)} merged together. If the field is not observed, the size of the field is 6 and includes only **applicationTag** and **subApplicationTag**.
- The **sub-application-table** option template maps the extracted field ID to name and description, as follows:
  - Extracted field ID: **subApplicationTag** (ID=97)
  - Name: **subApplicationName** (ID=109)
  - Description: **subApplicationDesc** (ID=110)
- All HTTP-based applications, such as YouTube, SharePoint, and so on, use the same sub-application ID, defined by the **subApplicationID**, as defined by the HTTP application.

Fields that Require Punt to the Route Processor

First Published: March 29, 2013
Revised: March 26, 2015

Table C-1 describes the media monitoring/metadata metrics that require punt to the route processor (RP).

Table C-1 Media Monitoring/Metadata Metric Fields

Metric	NetFlow ID
Media Monitoring related fields	
collect counter flows	3
collect application media bytes rate	37006
collect application media packets rate	37009
collect application media packets rate variation	37010
collect application media event	37011
collect monitor event	37012
collect timestamp interval	37013
collect transport packets lost rate	37021
collect transport rtp jitter mean	37023
collect application media packets rate variation min	37038
collect application media packets rate variation max	37039
collect transport rtp flow count	37040
collect transport packets lost rate min	37047
collect transport packets lost rate max	37048
timestamp absolute monitoring-interval start	65500
timestamp absolute monitoring-interval end	65501
Metadata related fields	
collect application version	105
collect application version name	106
collect application vendor	107
collect metadata global-session-id	37054

Metric	NetFlow ID
Media Monitoring related fields	
collect metadata multi-party-session-id	37055
collect metadata clock-rate	37056