Oracle® Communications

Diameter Signaling Router

Diameter Security Application User's Guide

Release 8.2

E90636 Revision 02

December 2022

ORACLE

Oracle Communications Diameter Signaling Router Diameter Security Application User's Guide, Release 8.2

My Oracle Support (MOS) (https://support.oracle.com) is your initial point of contact for all product support and training needs. A representative at Customer Access Support (CAS) can assist you with MOS registration.

Call the CAS main number at 1-800-223-1711 (toll-free in the US), or call the Oracle Support hotline for your local country from the list at http://www.oracle.com/us/support/contact/index.html.

See more information on My Oracle Support (MOS).

Table of Contents

1. Introduction 7
1.1 References 7
1.2 Overview of DSA Tasks 7
1.3 Intended Scope and Audience 7
1.4 Acronyms and Terms 7
1.5 Content Organization 10
1.6 Customer Training 10
2. Understanding DSA Functionality and Logic 10
2.1 DSA Overview 11
2.2 Understanding DSA Functionality 11
2.3 DSA Logic Process 12
2.3.1 DSA Mandatory Configuration 14
2.4 DSA Stateless Countermeasure Logic 14
2.4.1 Application-ID Whitelist Screening (AppIdWL) 14
2.4.2 Application-ID and Command-Code Consistency Check (AppCmdCst) 16
2.4.3 Origin Realm and Destination Realm Whitelist Screening (RealmWLScr) 16
2.4.4 Origin Host and Origin Realm Consistency Check (OhOrCstChk) 17
2.4.5 Destination-Realm and Origin-Realm Match Check (DrOrMatch) 17
2.4.6 Visited-PLMN-ID and Origin-Realm Consistency Check (VplmnORCst) 17
2.4.7 Realm and IMSI Consistency Check (RealmIMSICst) 18
2.4.8 Subscriber Identity Validation (SubsIdenValid) 18
2.4.9 Specific AVP Screening (SpecAVPScr) 19
2.4.10 AVP Multiple Instance Check (AVPInstChk) 19
2.5 DSA Stateful Countermeasure Logic 20
2.5.1 Message Rate Monitoring (MsgRateMon) 20
2.5.2 Time-Distance Check (TimeDistChk) 20
2.5.3 Previous Location Check (PreLocChk) 21
2.5.4 Source Host Validation HSS (SrcHostValHss) 22
2.5.5 Source Host Validation MME (SrcHostValMme) 22
3. Configuring DSA 23
3.1 DSA Pre-Activation Activities 23
3.2 Activate DSA 24
3.3 Configure DSA Business Logic and Database Schema 24
3.4 Configure SBR DB Name Mapping 25
3.5 Configure DSA Mandatory Options 25
3.6 ART Configuration for DSA 26

3.7 Enable DSA 26
3.8 Disable DSA 26
3.9 Deactivate DSA 27
4. DSA Tables 27
4.1 Configure DSA Tables 29
4.2 Provision DSA Tables 29
4.3 DSA Table Details 31
4.3.1 Security_Countermeasure_Config Table 31
4.3.2 Foreign_WL_Peers_Cfg_Sets Table 35
4.3.3 System_Config_Options Table 37
4.3.4 MCC_MNC_List Table 41
4.3.5 TTL_Config Table 43
4.3.6 AppIdWL_Config Table 44
4.3.7 Realm_List Table 45
4.3.8 VplmnORCst_Config Table 46
4.3.9 SpecAVPScr_Config Table 47
4.3.10 AVPInstChk_Config Table 49
4.3.11 TimeDistChk_Config Table 52
4.3.12 MsgRateMon_Config Table 53
4.3.13 AppCmdCst_Config Table 54
5. DSA MEALs 55
5.1 Configure DSA MEALs 55
5.2 Measurement 56
5.2.1 ProcessedBy<Countermeasure ShortName> 56
5.2.2 DetectedBy<Countermeasure ShortName> 56
5.2.3 DroppedBy<Countermeasure ShortName> 57
5.2.4 RejectedBy<Countermeasure ShortName> 57
5.2.5 FailedExec<Countermeasure ShortName> 58
5.3 SysMetric 59
5.3.1 VulnerableBy<Countermeasure ShortName> 59
5.3.2 MsgRatePerPeer 60
5.4 <Countermeasure ShortName>ExecFailed Alarm 61
6. DSA Vulnerable Message Logs 61
6.1 Configure Vulnerable Message Logging 62
6.2 dsa_application.cron File Script and Log 63
Appendix A. General Recommendations 64
Appendix B. My Oracle Support (MOS) 64

List of Tables

Table 1. DSA Configuration Tables 27
Table 2. Security_Countermeasure_Config Fields 31
Table 3. Field Details for Security_Countermeasure_Config 33
Table 4. Foreigh_WL_Peers_Cfg_Sets Fields 36
Table 5. Field Details for Foreigh_WL_Peers_Cfg_Sets 36
Table 6. System_Config_Options Fields 37
Table 7. Field Details for System_Config_Options 39
Table 8. MCC_MNC_List Fields 41
Table 9. Field Details for MCC_MNC_List 42
Table 10. TTL_Config Fields 43
Table 11. Field Details for TTL_Config 43
Table 12. AppIdWL_Config Fields 44
Table 13. Field Details for AppIdWL_Config 44
Table 14. Realm_List Fields 45
Table 15. Field Details for Realm_List 45
Table 16. VplmnORCst_Config 46
Table 17. Field Details for VplmnORCst_Config 46
Table 18. SpecAVPScr_Config Fields 47
Table 19. Field Details for SpecAVPScr_Config 48
Table 20. AVPInstChk_Config Fields 49
Table 21. Field Details for AVPInstChk_Config 50
Table 22. TimeDistChk_Config Fields 52
Table 23. Field Details for TimeDistChk_Config 52
Table 24. MsgRateMon_Config Fields 53
Table 25. Field Details for MsgRateMon_Config 53
Table 26. AppCmdCst_Config Fields 54
Table 27. Field Details for AppCmdCst_Config 54
Table 28. ProcessedBy<Countermeasure ShortName> Measurement 56
Table 29. DetectedBy<Countermeasure ShortName> Measurement 56
Table 30. DroppedBy<Countermeasure ShortName> Measurement 57
Table 31. RejectedBy<Countermeasure ShortName> Measurement 58
Table 32. FailedBy<Countermeasure ShortName> Measurement 58
Table 33. VulnerableBy<Countermeasure ShortName> SysMetric 59
Table 34. MsgRatePerPeer SysMetric 60
Table 35. <Countermeasure ShortName>ExecFailed Alarm 61
Table 36. fetchLogDsa.ini File Configuration Options 63

Table 37. Contermeasure Configuration 64

1. Introduction

Diameter Security Application (DSA) allows the home network operator to protect their network from vulnerable diameter messages. To achieve this, DSA enables the home network operators to define certain configurations, which are used by various countermeasures, for detecting vulnerable diameter messages from the roaming networks.

DSA menu options allow you to work with:

- Custom Measurements, Events, Alarms, and Logs (MEALs)
- General options
- Trial MPs assignment
- Application control
- System Options (SO Only)

DSA is a Diameter Custom Application (DCA) framework application. Like other DCA framework applications, you can use DSA to work with the DCA framework functions. If the Diameter Security Application is visible in the DCA framework GUI menu, the application is already activated and provisioned.

1.1 References

[1] Diameter Custom Applications Feature Activation Guide

[2] Diameter User's Guide

[3] DCA Programmer's Guide

[4] Session Binding Repository (SBR) User's Guide

1.2 Overview of DSA Tasks

This document provides the following types of information about DSA tasks:

- DSA logic
- Procedures to configure and manage DSA components, including DSA provisioning tables
- Information about DSA components and GUI elements
- References to related documentation including the DCA Programmer's Guide and DCA Feature Activation

1.3 Intended Scope and Audience

This content is intended for personnel who perform DSA tasks, and it includes procedures for performing tasks using the product GUI.

This content does not describe how to install or replace software or hardware.

The DSA software application interacts with SBR. For this reason, this content includes references to the shared applications, and might describe GUI options that are not visible or applicable to DSA.

1.4 Acronyms and Terms

Acronym or Term	Definition
AIR/A	Authentication-Information Request/Answer
ART	Application Routing Table

Acronym or Term	Definition
AVP	Attribute-Value Pair The Diameter protocol consists of a header followed by one or more attribute-value pairs (AVPs). An AVP includes a header and is used to encapsulate protocol-specific data (for example, routing information) as well as authentication, authorization or accounting information.
CLR/A	Cancel-Location Request/Answer
DCA	Diameter Custom Application
DRA	Diameter Relay Agent
DRL	Diameter Routing Layer The software layer of the stack that implements Diameter routing.
DSA	Diameter Security Application
DSR	Diameter Signaling Router A set of co-located Message Processors which share common Diameter routing tables and are supported by a pair of OAM servers. A DSR Network Element may consist of one or more Diameter nodes.
DSR/A	Delete-Subscriber-Data Request/Answer
FQDN	Fully Qualified Domain Name The complete domain name for a specific computer on the Internet (for example, www.oracle.com). A domain name that specifies its exact location in the tree hierarchy of the DNS.
GUI	Graphical User Interface The term given to that set of items and facilities which provides you with a graphic means for manipulating screen data rather than being limited to character based commands.
HSS	Home Subscriber Server A central database for subscriber information.
IDA	Insert-Subscriber-Data Request
IDR	Insert-Subscriber-Data Answer
IMSI	International Mobile Subscriber Identity A unique internal network ID identifying a mobile subscriber.
IP	Internet Protocol IP specifies the format of packets, also called datagrams, and the addressing scheme. The network layer for the TCP/IP protocol suite widely used on Ethernet networks, defined in STD 5, RFC 791. IP is a connectionless, best-effort packet switching protocol. It provides packet routing, fragmentation and re-assembly through the data link layer.
IPX	IP exchange
KPI	Key Performance Indicator
LTE	Long-Term Evolution
MAP	Mobile Application Part An application part in SS7 signaling for mobile communications systems.

Acronym or Term	Definition
MCC	Mobile Country Code A three-digit number that uniquely identifies a country served by wireless telephone networks. The MCC is part of the International Mobile Subscriber Identity (IMSI) number, which uniquely identifies a particular subscriber. See also MNC, IMSI.
MEAL	Measurements, Events, Alarms, and Logs
MNC	Mobile Network Code A number that identifies a mobile phone carrier. Used in combination with a Mobile Country Code (MCC) to uniquely identify a mobile phone operator/carrier. See also MCC.
MNO	Mobile Network Operator
MP	Message Processor The role of the Message Processor is to provide the application messaging protocol interfaces and processing. However, these servers also have OAM components. All Message Processors replicate from their Signaling OAM's database and generate faults to a Fault Management System.
NOAMP	Network Operations, Administration, Maintenance, and Provisioning
NOR/A	Notify Request/Answer
PLMN	Public Land Mobile Network A wireless communications network that uses land-based radio transmitters or base stations, intended for public use by terrestrial subscribers in vehicles or on foot. A PLMN is identified by its Mobile Country Code (MCC) and Mobile Network Code (MNC).
PRT	Peer Route Table or Peer Routing Table
PUR/A	Purge-UE Request/Answer
RSR/A	Reset-Subscriber Request/Answer
SBR	Session Binding Repository A highly available, distributed database for storing Diameter session binding data.
SOAM	System Operations, Administration, and Maintenance
SS7	Signaling System #7 A communications protocol that allows signaling points in a network to send messages to each other so that voice and data connections can be set up between these signaling points. These messages are sent over its own network and not over the revenue producing voice and data paths. The EAGLE is an STP, which is a device that routes these messages through the network.
ULR/A	Update-Location Request/Answer
U-SBR	Universal-SBR
VPLMN	Visited Public Land Mobile Network The PLMN to which a mobile subscriber has roamed when leaving the subscriber's Home Public Land Mobile Network.

1.5 Content Organization

The content in this document is organized as follows:

- General information about DSA including overview and logic information, the organization of this content, and how to get technical assistance.
- Understanding DSA Functionality and Logic describes DSA logic.
- Configuring DSA provides information about customizing DSA resources.
- DSA Tables provides information about provisioning DSA database.
- DSA MEALs provides information about DSA Measurements, SysMetrics, and Alarms.
- DSA Vulnerable Message Logs describes the usage of vulnerable message logging interface.

1.6 Customer Training

Oracle University offers training for service providers and enterprises. Visit our web site to view, and register for, Oracle Communications training:

http://education.oracle.com/communication

To obtain contact phone numbers for countries or regions, visit the Oracle University Education web site:

www.oracle.com/education/contacts

2. Understanding DSA Functionality and Logic

This section describes DSA functionality and logic.

DSA is a business logic application that functions within the DCA framework. The DCA framework is a prerequisite for DSA.

DSA must be activated to access DSA GUI menu and functionality.

Note: DCA framework is a set of APIs and services that are made available to DCA developers who need to develop applications.

The following documents contain information about the DCA framework applications and functionality:

- DCA Feature Activation
  - Activating and enabling DCA applications and framework
  - Deactivating DCA applications and framework
- DCA Programmer's Guide
  - Provisioning DCA
  - Developing stateful DCA applications
  - Monitoring DCA applications
  - Using DCA applications
  - Using Custom Meals
  - Using the DCA GUI
  - Understanding the development and environment
  - Using DCA APIs
  - Implementing DCA best practices

2.1 DSA Overview

Most security threats observed in a SS7 network (for example, Location Tracking, Call Intercept, Subscriber Denial Service, SMS Spams etc.) use messages from the Mobile Application Part (MAP) in the control plane. Similar kind of attacks can be simulated by the hacker using MAP equivalent Diameter Message in a LTE network. Most of the Diameter security vulnerabilities are introduced from roaming networks through IPX or directly from roaming partner networks. Therefore, there is a need for Mobile Network Operators (MNOs) to protect their home network from various diameter vulnerabilities by filtering out vulnerable Diameter messages received from various roaming partners.

DSA lets the operator protect its LTE network from various threats/attacks from roaming partners. This application defines various validation procedures (called countermeasures), which can be independently enabled/disabled as per the user's requirement. Some of these countermeasures require data from previous diameter messages to validate the current diameter message. In these cases, U-SBR is used to preserve the data of the previous diameter message, which is later retrieved for validating subsequent diameter messages.

During the message validation by a countermeasure, if the message is found as vulnerable by the countermeasure's business logic, DSA allows the operator to either discard the vulnerable message or send an error answer to the vulnerable message or continue processing the vulnerable message (to find more vulnerabilities).

DSA is configured as the owner of a U-SBR database. To avoid overloading DSA, the Application Routing Table (ART) is configured to route only messages from foreign networks (Incoming Roaming Traffic, meaning, messages that have Origin-Realm that do not match the realm of the operator's home network and Destination-Realm that match the realm of the operator's home network) to DSA. Some countermeasures are required to process outgoing diameter messages that are being sent to a foreign network from the operator's home network. These outgoing diameter messages to the foreign networks (Outgoing Traffic to foreign network, meaning, messages that are have Origin-Realm that match the realm of the operator's home network and Destination-Realm that does not match the realm of the operator's home network) are also routed to DSA.

DSA can be enabled and disabled as a DCA framework application. Disabling DSA on a specific site is possible only if DSA has been disabled on all the DA-MPs for that specific site. DSA can be completely configured at the SO.

The DCA framework creates applications on top of the Diameter Signaling Router (DSR) allowing for a faster development cycle. There can be up to 10 versions of each DCA in the various states.

To use DSA for DCA, the DCA framework must be activated on the NO. Activation needs to be performed only once. See the [1] Diameter Custom Applications Feature Activation Guide for instructions on how to activate the DCA framework.

When DSA is initially installed, it is disabled, and you must manually enable it. To do so, navigate to **Diameter > Maintenance > Applications** and enable the application for every DMAP using DSA.

If DSA is in the DCA framework GUI menu, this means the application is already enabled, but that does not guarantee it is provisioned. You can also disable DSA from the **Diameter > Maintenance > Applications**.

DCA framework application functionality varies between the SO and NO, for example, System Options is available on the SO only.

2.2 Understanding DSA Functionality

DSA allows the operator to screen various diameter messages received from roaming partners for possible vulnerability. It should be deployed at DSR, which is acting as DEA for the operator's home network so all roaming traffic can be screened for vulnerability by DSA.

DSA screens the incoming diameter message for vulnerability by a set of countermeasures. Each countermeasure has a predefined validation process, which is performed to validate the incoming

diameter message for vulnerability. The validation process requires some DSA specific configuration data for performing validation. Apart from DSA specific configuration, some of the countermeasures also require data from an earlier diameter message. Based on this, the countermeasures are broadly divided into two categories.

- Stateful countermeasures
- Stateless countermeasures

Stateful countermeasures require data from an earlier diameter message (apart from DSA configuration data) for checking vulnerability of a given incoming diameter message. U-SBR is used in this case to save data from a diameter message. The saved data are later fetched by the countermeasure for performing the validation procedure. A list of stateful countermeasures the DSA provides includes:

- Message Rate Monitoring
- Time-Distance Check
- Previous Location Check
- Source Host Validation HSS
- Source Host Validation MME

Stateless countermeasures do not requires any data from earlier diameter message for checking vulnerability of a given incoming diameter message. The message is screened for vulnerability by using DSA configuration data. So stateless countermeasures do not require U-SBR for performing validation procedure. A list of stateful countermeasures DSA provides includes:

- Application-ID Whitelist Screening
- Application-ID and Command-Code Consistency Check
- Origin Realm and Destination Realm Whitelist Screening
- Origin host and Origin Realm Consistency Check
- Destination-Realm and Origin-Realm Match Check
- Visited-PLMN-ID and Origin-Realm Consistency Check
- Realm and IMSI Consistency Check
- Subscriber Identity Validation
- Specific AVP Screening
- AVP Multiple Instance Check

2.3 DSA Logic Process

To trigger DSA logic, some prerequisite conditions are required. For example, the DCA framework must be activated and DSA must be activated, enabled, and provisioned. See Configuring DSA and DSA Tables.

DSA logic is triggered when DSA receives a diameter message. Once a diameter message is received:

1. DSA starts executing the provisioned countermeasures, which are enabled, in a predefined sequence irrespective of the countermeasure's provisioning sequence. Refer to 4.3.1 for provisioning countermeasures using **countermeasure_Type**.
2. Each countermeasure can be enabled or disabled independently for screening the message for vulnerability. Refer to 4.3.1 for enabling/disabling countermeasure using **Admin_Status**.

3. The stateless countermeasures are performed first followed by stateful countermeasures for better efficiency.
   - The stateless countermeasures are executed in the this sequence (if configured and enabled):
     1. Application-ID Whitelist Screening (AppIdWL)
     2. Application-ID and Command-Code Consistency Check (AppCmdCst)
     3. Origin Realm and Destination Realm Whitelist Screening (RealmWLScr)
     4. Origin Host and Origin Realm Consistency Check (OhOrCstChk)
     5. Destination-Realm and Origin-Realm Match Check (DrOrMatch)
     6. Visited-PLMN-ID and Origin-Realm Consistency Check (VplmnORCst)
     7. Realm and IMSI Consistency Check (RealmIMSICst)
     8. Subscriber Identity Validation (SubsIdenValid)
     9. Specific AVP Screening (SpecAVPScr)
     10. AVP Multiple Instance Check (AVPInstChk)
   - The stateful countermeasures are executed in this sequence (if configured and enabled):
     1. Message Rate Monitoring (MsgRateMon)
     2. Time-Distance Check (TimeDistChk)
     3. Previous Location Check (PreLocChk)
     4. Source Host Validation HSS (SrcHostValHss)
     5. Source Host Validation MME (SrcHostValMme)
4. Provisioning is provided to specify the list of foreign peers (refer to 4.3.2 for configuring whitelist foreign peers using **Foreign_WL_Peer_Cfg_Set**) for which a given countermeasure is executed for checking vulnerability. This is provided for flexibility to apply individual countermeasures only to untrusted foreign network peers.
5. When a message is found to be vulnerable by a countermeasure, action is performed depending upon the countermeasure's operating mode. Refer to 4.3.1 for configuring countermeasure's **Operating_Mode**. The supported operating modes are:
   - **Detection_Only**: The countermeasure works on monitoring mode. The vulnerable message is only reported to the user by measurements (refer to 5).
   - **Detection_And_Correction_By_Drop**: The vulnerable message is rejected at DSR and is not processed/relayed any further.
   - **Detection_And_Correction_By_Send_Answer**: The vulnerable message is discarded by DSR by sending an Error Answer and is not processed/relayed any further. Error Answer can be customized per countermeasure by configuring Result-Codes, Error-Messages, and Vendor-ID. Refer to 4.3.1 for configuring countermeasure's **Result_Code**, **Error_Message**, and **Vendor_ID**.
6. When a countermeasure finds a message to be vulnerable and **Detection_Only** is configured as operating mode, then the user can choose to continue processing the message and checking for more vulnerability by the remaining configured countermeasure. Refer to 4.3.1 for configuring countermeasure's **Continue_If_Vulnerable** option.
7. Once the message is processed by all the provisioned countermeasures and not found as vulnerable or found as vulnerable, but continue if vulnerable option is selected, then DSA does not take any action on the message. This message is further processed by DSR for relaying it.
8. An option is available for logging the vulnerable message details into a file that can be used for analyses. A user can choose to enable/disable logging. Refer to 4.3.3 for enabling/disabling DSA

vulnerable message logging using **Enable_Tracing** option. Refer to 6 for DSA vulnerable message logging framework.

2.3.1 DSA Mandatory Configuration

To screen the incoming message for vulnerability, DSA uses various values provisioned in DSA tables (refer to 4.3) for executing countermeasure's business logic. A few of these tables are required to be provisioned for enabling DSA business logic. Reaming tables are specific to countermeasure's business logic and needs to be provisioned only if the countermeasure is provisioned.

Countermeasure specific DSA tables are discussed in the respective countermeasures in more details. This is a list of configuration that must be done to enable DSA business logic.

- At least one countermeasure needs to be provisioned in the Security_Countermeasure_Config Table.

  These provisioned values define the list of countermeasures that screen the incoming message for vulnerability.

- At least one Home network's MCC and MNC needs to be provisioned in the MCC_MNC_List Table.

  These provisioned values determine the **Roaming Status** (Inbound Roaming Subscriber vs. Outbound Roaming Subscriber) of any given subscriber. If the MCC and MNC portion of the subscriber's IMSI matches with the Home network's MCC and MNC, then the subscriber is treated as an outbound roaming subscriber. Otherwise, the subscriber is treated as an inbound roaming subscriber.

- At least one Home networks' Realm needs to be provisioned in the Realm_List Table.

  These provisioned values determine the **Message Type** (Ingress Message vs Egress Message) of any incoming diameter message. If the incoming message's Origin-Realm AVP value does not match the Home network's Realm, then the message is treated as an ingress message from a roaming network. If the incoming message's Origin-Realm AVP value matches the Home network's Realm, and Destination-Realm AVP value does not match the Home network's Realm, then the message is treated as a home network's egress message destined to a roaming network.

- System_Config_Options Table needs to be provisioned with an entry.

  This provisioned value defines the behavior of DSA when an U-SBR failure occurs or any logical error occurs while executing DSA Perl business logic or enabling/disabling logs of vulnerable message details. It also defines a few countermeasure-specific options, which are discussed in more detail in the countermeasure's business logic section.

2.4 DSA Stateless Countermeasure Logic

Stateless countermeasures do not require maintenance of any State-Data (in U-SBR) for validating vulnerability of the diameter message.

2.4.1 Application-ID Whitelist Screening (AppIdWL)

This countermeasure screens the ingress diameter request message to check if the Peer from which the message is received is allowed to send this diameter message.

This countermeasure considers the ingress diameter request message as vulnerable if any of these conditions are true:

- The Application-ID of the ingress diameter message is not configured
- The Application-ID of the ingress diameter message is configured but the Peer from which the diameter message is received is not configured in the Whitelist Foreign Peer List of Security_Countermeasure_Config Table.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following table for this countermeasure.

AppIdWL_Config Table:	For configuring allowable Application-ID and Peer list combinations used by this countermeasure for screening.

2.4.2 Application-ID and Command-Code Consistency Check (AppCmdCst)

This countermeasure screens the ingress diameter request message to check if the received Application-ID and Command-Code combination is allowed for a given Roamer Type.

This countermeasure considers the ingress diameter request message as vulnerable if any of these conditions are true:

- Subscriber is an Inbound Roaming Subscriber, but the received Application-ID and Command-Code is not configured as an allowable combination for an Inbound Roamer.
- Subscriber is an Outbound Roaming Subscriber, but the received Application-ID and Command-Code is not configured as an allowable combination for an Outbound Roamer.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following table for this countermeasure.

AppCmdCst_Config Table: For configuring allowable Application-ID and Command-Code combinations for Inbound and Outbound Roamers which are used by this countermeasure for screening.

2.4.3 Origin Realm and Destination Realm Whitelist Screening (RealmWLScr)

This countermeasure screens the ingress diameter request message to check if the received Origin-Realm and Destination-Realm are allowed from the ingress Peer or. This ingress diameter message screening is done for both Inbound Roaming Subscribers and Outbound Roaming Subscribers.

This countermeasure also screens the egress diameter request message to check if DSR is allowed to send a diameter request message with the given Destination-Realm. The egress diameter message screening is only done for Inbound Roaming Subscribers.

Screening of ingress diameter message for Origin-Realm, screening of ingress diameter message for Destination-Realm, and screening of egress diameter message for Destination-Realm can be enabled/disabled independently.

This countermeasure considers the incoming diameter request message as vulnerable if any of these conditions are true:

- The Origin-Realm of the ingress diameter message is not configured as Foreign network's Realm.
- The Origin-Realm of the ingress diameter message is configured as Foreign network's Realm but the Peer from which the diameter message is received is not configured in the Whitelist Foreign Peer List.
- The Destination-Realm of the ingress diameter message is not configured as Home network's Realm.
- The Destination-Realm of the ingress diameter message is configured as Home network's Realm but the Peer from which the diameter message is received is not configured in the Whitelist Foreign Peer List.
- For an Inbound Roamer, the Destination-Realm of the egress diameter message is not configured as Foreign network's Realm.

Note: Appropriate ART configuration needs to be done for routing the egress request messages (only toward foreign networks) to DSA so that screening of egress diameter message for Destination-Realm can be performed. See ART Configuration for DSA for more details.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following tables for this countermeasure.

Realm_List Table: For configuring allowable Realm and Peer list combinations for Home network and Foreign network which are used by this countermeasure for screening.

System_Config_Options Table: Option for enabling/disabling screening of:

- ingress diameter message for Origin-Realm
- ingress diameter message for Destination-Realm
- egress diameter message for Destination-Realm

2.4.4 Origin Host and Origin Realm Consistency Check (OhOrCstChk)

This countermeasure screens the ingress diameter request message to check if the FQDN string of Origin-Host ends with the Origin-Realm string.

The option is available to provision an exception list of Realms. Any ingress diameter request message with Origin-Realm matching the exception list is exempted from this countermeasure's screening.

This countermeasure considers the ingress diameter request message as vulnerable if this condition is true:

- The Origin-Realm is not configured in the exception list of Realms and the Origin-Host's FQDN string is not ending with Origin-Realm's string.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following table for this countermeasure.

System_Config_Options Table: For configuring exception list of Realms, which are exempted from this countermeasure's screening.

2.4.5 Destination-Realm and Origin-Realm Match Check (DrOrMatch)

This countermeasure screens the ingress diameter request message to check if the Origin-Realm and Destination-Realm are having different value.

This countermeasure considers the ingress diameter request message as vulnerable if this condition is true:

- The Origin-Realm and Destination-Realm of the ingress diameter request are having same value.

Apart from the mandatory Configuration discussed in 2.3.1 DSA Mandatory Configuration, no other tables need to be configured for this countermeasure.

2.4.6 Visited-PLMN-ID and Origin-Realm Consistency Check (VplmnORCst)

This countermeasure screens the ingress diameter request message to check if the MCC and MNC values in Visited-PLMN-ID AVP match the MCC and MNC values in the Origin-Realm AVP.

The option is available to configure the Application-ID and Command-Code combinations this countermeasure uses for screening.

Below are the pre-conditions for executing this countermeasure. If any of these conditions are not met, then the ingress diameter request message is not screened for vulnerability.

- The Application-ID and Command-Code of the ingress diameter request message must be configured.
- Visited-PLMN-ID AVP must be present in the ingress diameter request message.
- The Origin-Realm AVP must be in the format as defined in 3GPP 23.003.

This countermeasure considers the ingress diameter request message as vulnerable if this condition is true:

- The MCC and MNC values in Visited-PLMN-ID AVP do not match the MCC and MNC values in the Origin-Realm AVP

Note: As per Section 19.2 of 3GPP 23.003, the Realm should be in the form of:

epc.mnc<MNC>.mcc<MCC>.3gppnetwork.org

where <MNC> and <MCC> fields correspond to the MNC and MCC of the operator's PLMN. Both fields are three (3) digits. If the MNC of the PLMN is two (2) digits, then add a zero to the beginning.

For example, for a network with MCC = 234 and MNC = 15, Realm/Domain name is epc.mnc015.mcc234.3gppnetwork.org.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following table for this countermeasure.

VplmnORCst_Config Table: For configuring the Application-ID and Command-Code combinations used by this countermeasure for screening.

2.4.7 Realm and IMSI Consistency Check (RealmIMSICst)

This countermeasure screens the ingress diameter request message to check if the MCC and MNC values present in IMSI match the MCC and MNC values in the Origin-Realm/Destination-Realm AVP.

For Inbound Roaming Subscriber, MCC and MNC values of the Origin-Realm AVP are used for matching; and for Outbound Roaming Subscriber, MCC and MNC values of the Destination-Realm AVP are used for matching.

Below are the pre-conditions for executing this countermeasure. If any of these conditions are not met, then the ingress diameter request message is not screened for vulnerability.

- For an Inbound Roamer, the countermeasure screens only S6a/d IDR, RSR, DSR or CLR messages.
- Screening is performed only if the Origin-Realm AVP is in the format as defined in 3GPP 23.003.
- For an Outbound Roamer, the countermeasure screens only S6a/d AIR, ULR, PUR, or NOR messages.
- Screening is performed only if the Destination-Realm AVP is in the format as defined in 3GPP 23.003.

This countermeasure considers the ingress diameter request message as vulnerable if any of these conditions are true:

- For an Inbound Roamer, the MCC and MNC values present in Origin-Realm AVP do not match the MCC and MNC values in the IMSI.
- For an Outbound Roamer, the MCC & MNC value present in Destination-Realm AVP do not match the MCC and MNC values in the IMSI.

Note: For S6a IDR, DSR, CLR, AIR, ULR, PUR, and NOR messages, User-Name AVP is used to fetch the MCC and MNC of the IMSI.

For S6a RSR messages, User-ID AVP is used to fetch the MCC and MNC of the IMSI.

Note: As per Section 19.2 of 3GPP 23.003, the Realm should be in the form of:

epc.mnc<MNC>.mcc<MCC>.3gppnetwork.org

where <MNC> and <MCC> fields correspond to the MNC and MCC of the operator's PLMN. Both fields are three (3) digits. If the MNC of the PLMN is 2 digits, then add a zero to the beginning.

For example, for a network with MCC = 234 and MNC = 15, Realm/Domain name is epc.mnc015.mcc234.3gppnetwork.org.

Apart from the mandatory Configuration discussed in 2.3.1 DSA Mandatory Configuration, no other tables need to be configured for this countermeasure.

2.4.8 Subscriber Identity Validation (SubsIdenValid)

This countermeasure screens the ingress diameter request message for an Inbound Roaming Subscriber to check if the Subscriber's identity is valid.

This countermeasure considers the ingress diameter request message for an Inbound Roaming Subscriber as vulnerable if this condition is true:

- The MCC and MNC values present in the User-Name AVP are not provisioned as MCC and MNC of a Foreign network.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following table for this countermeasure.

MCC_MNC_List Table: For configuring MCC and MNC combinations of Foreign networks used by this countermeasure for validating Subscriber's identity.

2.4.9 Specific AVP Screening (SpecAVPScr)

This countermeasure screens the ingress diameter request/answer message for checking invalid AVP value(s).

The option is available to configure the list of AVP values used by this countermeasure for performing screening.

This countermeasure considers the ingress diameter request/answer message as vulnerable if this condition is true:

- One of the AVP in the ingress request/answer message matches the configured AVP value, which is provisioned as an invalid value.

Note: Appropriate ART configuration needs to be done for routing the egress request messages (only toward foreign networks) to DSA so the ingress answer message from the foreign peers can be screened for vulnerability by this countermeasure. See ART Configuration for DSA for more details.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following table for this countermeasure.

SpecAVPScr_Config Table For configuring values for AVP(s) used by this countermeasure for screening. AVP value, applicable Application-ID, Command-Code, and the Message Type (Request/Answer) combination are defined.

2.4.10 AVP Multiple Instance Check (AVPInstChk)

This countermeasure screens the ingress diameter request/answer message for checking minimum and maximum allowable instance of AVP(s).

The option is available to configure the list of AVPs along with the allowable minimum and maximum instance values used by this countermeasure for performing screening.

This countermeasure considers the ingress diameter request/answer message as vulnerable if any of these conditions are true:

- One of the AVP in the ingress request/answer message is having lesser number of instances than the configured minimum allowed number of instances.
- One of the AVP in the ingress request/answer message is having higher number of instances than the configured maximum allowed number of instances.

Note: Appropriate ART configuration needs to be done for routing the egress request messages (only towards foreign networks) to DSA so that ingress answer message from the foreign peers can be screened for vulnerability by this countermeasure. See ART Configuration for DSA for more details.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following table for this countermeasure.

AVPInstChk_Config Table — For configuring minimum and maximum allowable instance of AVPs used by this countermeasure for screening. AVP minimum and maximum instances, the applicable Application-ID, Command-Code, and the Message Type (Request/Answer) combination are defined.

2.5 DSA Stateful Countermeasure Logic

Stateful countermeasures require maintenance of some State-Data (depending upon the countermeasure's business logic) for validating various diameter messages. U-SBR is used for maintaining the State-Data record.

First the State-Data is created for the Subscriber when the reference diameter message is received (depending upon the countermeasure type, the reference diameter message varies). For subsequent diameter messages for that subscriber, the State-Data is used to validate against the incoming diameter message content.

Note: For all the stateful countermeasures (except Message Rate Monitoring (MsgRateMon)), the State-Data is created only after DSA processes the referenced diameter message. The countermeasures mark the non-vulnerable message as vulnerable if appropriate State-Data is not present for that subscriber.

Therefore, it is important that after a stateful countermeasure is enabled, all the outbound and inbound roamers must be forced to re-register so DSA can process the reference diameter messages first or, alternatively, keep the stateful countermeasure's Operating Mode as **Detection Only**.

2.5.1 Message Rate Monitoring (MsgRateMon)

This countermeasure screens various ingress diameter request message to check if the current aggregate request message rate for a given diameter message type is less than the threshold value.

The option is available to configure the threshold value for various diameter message types (that is, Application-ID and Command-Code combinations) used by this countermeasure for screening.

For each diameter message type, aggregate rate is maintained foreign peers (the foreign peers list is the Foreign_WL_Peer_Cfg_Set of Security_Countermeasure_Config Table for this countermeasure).

This countermeasure considers the ingress diameter request message as vulnerable if this condition is true:

- The current aggregate request message rate of the diameter message type and ingress peer combination is greater than the configured threshold value.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following table for this countermeasure.

MsgRateMon_Config Table: — For configuring Application-ID and Command-Code combinations along with its allowable threshold value used by this countermeasure for screening.

2.5.2 Time-Distance Check (TimeDistChk)

This countermeasure screens S6a/d ULR and AIR messages of Outbound Roaming Subscribers currently in international roaming to check if it is physically possible for a Subscriber to move from its previous location to the new location within the current transit time.

This countermeasure screens the S6a/d ULR and AIR messages for vulnerability only if there is a successful registration record.

The Outbound Roaming Subscriber is considered successfully registered to an MME when an ingress S6a/d ULR/A message (ULA with Result-Code as 2xxx) is processed by DSA.

The option is available to configure minimum transit time required between two locations used by this countermeasure for screening. A user can chose to configure the minimum transit time between two countries (that is, MCC of the Foreign network) or between two Visited-PLMN-IDs (that is, MCC and MNC of the Foreign network), but not both.

The option is also available to consider the S6a/d ULR and AIR messages as vulnerable if the minimum transit time between the previous and current location of the subscriber is not configured.

This countermeasure considers the S6a/d ULR and AIR messages as vulnerable if an earlier successful registration is already processed by DSA and any of these conditions are true

- The transit time between the previous and current location is configured, but the actual transit time is less than the configured minimum transit time.
- The transit time between the previous and current location is not configured and the configuration says to mark the message as vulnerable, if matching configuration not found.

Note: International Roaming is identified by matching the Home MCCs configured in MCC_MNC_List Table (for example, first three digits of MCC_MNC with Network_Type as Home_Network) against the MCC value in Visited-PLMN-Id AVP.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following tables for this countermeasure.

TimeDistChk_Config Table:	For configuring minimum transit time between two locations used by this countermeasure for screening.
System_Config_Options Table:	Option to indicate the Source and Destination locations configured in TimeDistChk_Config Table are MCCs or Visited-PLMN-IDs. Option to define the behavior when no matching Source and Destination location is configured.
TTL_Config Table:	For configuring the TTL value of the State-Data created for this countermeasure.

2.5.3 Previous Location Check (PreLocChk)

This countermeasure screens S6a/d PUR and NOR messages of Outbound Roaming Subscribers to check if the MME from which the PUR/NOR message is received is the same MME on which the subscriber is currently registered.

The Outbound Roaming Subscriber is considered successfully registered to a Foreign network MME when an ingress S6a/d ULR/A (ULA with Result-Code as 2xxx) is processed by DSA.

The Outbound Roaming Subscriber is considered de-registered from the Foreign network MME when:

- An egress S6a/d CLR is processed by DSA, or
- An egress S6a/d RSR is processed by DSA, or
- A non-vulnerable ingress PUR message is processed by DSA.

This countermeasure considers the ingress S6a/d PUR and NOR message as vulnerable if any of these conditions are true:

- The subscriber has not registered to any MME.
- The MME from which the PUR/NOR message is received is different from the MME on which the subscriber is registered.

Note: Appropriate ART configuration needs to be done for routing the egress request messages (only towards foreign networks) to DSA so that the egress CLR can be processed by this countermeasure. See ART Configuration for DSA for more details.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following table for this countermeasure.

TTL_Config Table	For configuring the TTL value of the State-Data created for this countermeasure.

2.5.4 Source Host Validation HSS (SrcHostValHss)

This countermeasure screens S6a/d IDR, DSR and CLR message of Inbound Roaming Subscribers to check if the HSS from which the IDR/DSR/CLR/RSR message is received is the same HSS to which earlier registration request has been sent successfully.

The Inbound Roaming Subscriber is considered successfully registered with the Home network when an egress S6a/d ULR/A (ULA with Result-Code as 2xxx) is processed by DSA.

The Inbound Roaming Subscriber is considered de-registered from the Home network when:

- An egress S6a/d PUR is processed by DSA, or
- A non-vulnerable ingress CLR or RSR(with appropriate range of User-Ids) message is processed by DSA.

This countermeasure considers the ingress S6a/d IDR, DSR and CLR message as vulnerable if any of these conditions are true:

- The subscriber has not registered with the Home network.
- The HSS from which the IDR/DSR/CLR message is received is different from the HSS to which earlier registration request has been sent.

Note: Appropriate ART configuration needs to be done for routing the egress request messages (only towards foreign networks) to DSA so that the egress ULR and PUR can be processed by this countermeasure. See ART Configuration for DSA for more details.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following table for this countermeasure.

TTL_Config Table	For configuring the TTL value of the State-Data created for this countermeasure.
System_Config_Options Table:	Check field **Process_Foreign_RSR_Msg**, if RSR message needs to be process by this counter measure

2.5.5 Source Host Validation MME (SrcHostValMme)

This countermeasure screens S6a/d ULR and PUR message of Outbound Roaming Subscribers to check if the MME from which these messages are received is valid. This countermeasure also validates the sequential ordering of authentication and registration process when the subscriber moves from one foreign network to another foreign network.

The Outbound Roaming Subscriber is considered successfully authenticated by the Home network when a ingress S6a/d AIR/A (AIA with Result-Code as 2xxx) is processed by DSA.

The Outbound Roaming Subscriber is considered as successfully registered to a Foreign network when a non-vulnerable ingress S6a/d ULR/A (ULA with Result-Code as 2xxx) is processed by DSA.

The subscriber is considered de-registered from the Foreign network when:

- An egress S6a/d CLR is processed by DSA, or
- An egress S6a/d RSR is processed by DSA, or
- A non-vulnerable ingress PUR message is processed by DSA.

This countermeasure considers the ingress S6a/d ULR message as vulnerable if any of these conditions are true:

- The subscriber has not authenticated by the Home network.
- The Visited-PLMN-Id from which the subscriber has authenticated is not matching with the Visited-PLMN-Id from which registration request is received.

This countermeasure considers the ingress S6a/d PUR message as vulnerable if any of these conditions are true:

- The subscriber has not authenticated by the Home network.
- The subscriber has not registered with the Home network.
- The MME from which the PUR message is received is different from the MME on which the subscriber is registered.

Note: Appropriate ART configuration needs to be done for routing the egress request messages (only towards foreign networks) to DSA so that the egress CLR/RSR can be processed by this countermeasure. See ART Configuration for DSA for more details.

Apart from the mandatory configuration discussed in 2.3.1 DSA Mandatory Configuration, configure the following table for this countermeasure.

TTL_Config Table	For configuring the TTL value of the State-Data created for this countermeasure.

3. Configuring DSA

This section contains information about DSA and describes the procedures used to activate, configure, and deactivate DSA.

DSA uses these tables for holding configuration values:

- Security_Countermeasure_Config Table
- Foreign_WL_Peers_Cfg_Sets Table
- System_Config_Options Table
- MCC_MNC_List Table
- TTL_Config Table
- AppIdWL_Config Table
- Realm_List Table
- VplmnORCst_Config Table
- SpecAVPScr_Config Table
- AVPInstChk_Config Table
- TimeDistChk_Config Table
- MsgRateMon_Config Table
- AppCmdCst_Config Table

Some of these tables are specific to countermeasures used only during that countermeasure's business logic execution.

3.1 DSA Pre-Activation Activities

Before activating DSA as a DCA application, DCA framework must be activated on the NO. See [1] Diameter Custom Applications Feature Activation Guide.

Following DCA framework activation, DSA can be activated.

Note: After DSA is activated, by default the application is in the disabled state. While disabled, no diameter traffic is delivered to DSA. See [2] Diameter User's Guide for the procedure to enable an application.

DSA's operational status is unavailable until a successful compiled version (production or trial version) of DSA is configured.

3.2 Activate DSA

This procedure activates DSA.

See [1] Diameter Custom Applications Feature Activation Guide for detailed information.

1. Make sure the DCA framework has already been activated. See **DCA Feature Activation**.
2. Activate DSA using the DCA Application Activate procedure. See **DCA Feature Activation**.
3. Recommended DCA Short Name: **DSA**
4. Recommended DCA Long Name: **Diameter Security Application**
5. Post DSA activation, check the visibility of DSA subtree in the main menu **DCA Framework > Diameter Security Application**.

This procedure verifies DSA is activated before enabling DSA and performing provisioning activities.

1. Confirm the DSA folder is visible on the GUI under the main menu: **DCA Framework**.
2. All measurements and KPIs associated with the DCA framework are visible on the **Measurements > Report** and **Status & Manage > KPIs** screens.

   After activation, DSA becomes visible across DSR (for example, ART and maintenance).

Note: After activating DCA, the DCA framework allocates a default set of resources to it. Due to the complexity of DSA, it is recommended to increase the resource allocation to achieve the desired throughput.

This procedure sets DSA's desired resource allocation.

1. Log into the active SO server using SSH as **admusr**.
2. Execute the **update_dca_thread_count_damp_profile.sh** script.
3. Select **1** to increase thread counts.
4. Restart the DAMPs hosting DSA under this SO..

3.3 Configure DSA Business Logic and Database Schema

This procedure imports DSA business logic and the configuration database schema using the DSA JSON file.

DSA NO JSON filename: **Diameter_Security_Application-Version1.json**

See [3] DCA Programmer's Guide for detailed information.

1. From the NO GUI main menu, navigate to **DCA Framework > Diameter Security Application > Application Control**.
2. Select the newly added **DSA Version Name**.
3. Click **Business Logic** in the Import section of the Application Control page.
4. Click **Browse** and select the **Diameter_Security_Application-Version1.json** file from the File upload screen.

5. Mark the **Abort on first error** checkbox to abort the import process in case of error.
6. Click **Import** to start the import process.

This procedure verifies DSA JSON has successfully imported before enabling DSA and performing provisioning activities.

1. From the NO GUI main menu, navigate to **DCA Framework > Diameter Security Application > Application Control**, and make sure an entry is added in DCA application version details table.
2. Select the newly added version and click **Config Tables and Data**.
3. Make sure all DSA configuration tables are listed.
4. Select the newly added version and click **Development Environment**.
5. Make sure DSA Perl business logic is present.
6. Select the newly added version and click **SBR DB Name Mapping**.
7. Make sure **global_sbr** is listed as a SBR database Logical Name in the SBR DB Name Mapping table.

3.4 Configure SBR DB Name Mapping

This procedure assigns the physical U-SBR DB name to DSA logical SBR DB Name. This is required only for Stateful countermeasures. Before assigning the physical U-SBR DB name, a U-SBR DB must be configured and available. See [4] Session Binding Repository (SBR) User's Guide for configuring a U-SBR DB.

See [3] DCA Programmer's Guide for detailed information.

1. From the NO GUI main menu, navigate to **DCA Framework > Diameter Security Application > Application Control**.
2. Select the newly added DSA **Version Name** and click **SBR DB Name Mapping**.
3. Select the entry with SBR Database Logical Name as **global_sbr** from the SBR DB Name Mapping table and click **Edit**.
4. Add the U-SBR DB created for DSA to the **Included SBR Databases** list and click **Apply/OK**.

3.5 Configure DSA Mandatory Options

This procedure configures various DSA Mandatory Options.

Increase the maximum supported State-Data size:

1. From the NO GUI main menu, navigate to **DCA Framework > Configuration**
2. Set the **Maximum Size of Application State** to **1600**.
3. Click **Apply**.

Configure general options:

1. From the NO GUI main menu, navigate to **DCA Framework > Diameter Security Application > General Options**.
2. Update **Perl Subroutine for Diameter Request** to **process_request**.
3. Update **Perl Subroutine for Diameter Answer** to **process_answer**.
4. Unselect the **Read Only U-SBR Access as Guest** option if DSA is not the owner of the U-SBR DB used in SBR DB Name Mapping. See Configure SBR DB Name Mapping more details.
5. Update **Max. U-SBR Queries per Message** to **10**.

6. Unselect the **Enable Opcodes Accounting** option to disable opcode accounting.
7. Click **Apply**.

3.6 ART Configuration for DSA

DSA processes ingress diameter messages received from foreign networks to check vulnerability. For this:

- Create an ART to route all the ingress traffic to DSA.
- Assign the ART to all the foreign peers.

If you do not want to screen ingress diameter messages from a specific foreign peer, then skip the ART configuration for that peer.

DSA also processes egress diameter messages to send to a foreign network from a home network. For this:

- Create an ART to route only egress traffic from a home network toward a foreign network to DSA, that is, messages where
  - Origin-Realm matches the Home network Realm, and
  - Destination-Realm does not match the Home network Realm.
- Assign the ART only to those home network peers that can send egress messages to a foreign network.

If you want to screen the diameter message using any of these countermeasures, then assign the ART to the home peers that can send egress messages to a foreign network:

Stateless countermeasures:

- Origin Realm and Destination Realm Whitelist Screening (RealmWLScr)
- Specific AVP Screening (SpecAVPScr)
- AVP Multiple Instance Check (AVPInstChk)

Stateful countermeasures:

- Previous Location Check (PreLocChk)
- Source Host Validation HSS (SrcHostValHss)
- Source Host Validation MME (SrcHostValMme)

Note: For the above stateful countermeasures, if egress traffic is not routed to DSA, then the countermeasure business logic does not work, which may lead to traffic loss due to wrongly marking the messages as vulnerable by the countermeasures.

3.7 Enable DSA

This procedure enables DSA on the SO.

1. Navigate to **Diameter > Maintenance > Applications**.
2. Select DCA_DSA entries and click **Enable**.

3.8 Disable DSA

This procedure disables DSA on the SO.

1. Navigate to **Diameter > Maintenance > Applications**.
2. Select DCA_DSA entries and click **Disable**.

3.9 Deactivate DSA

This procedure deactivate DSA. You cannot deactivate DSA while a version of the respective application is still in the Production and/or Trial state.

Before deactivation can take place, DSA must be disabled on all MPs in the network and no ART rules should refer to DSA.

See [1] Diameter Custom Applications Feature Activation Guide for detailed information.

1. Disable DSA for all the MPs from the SO GUI main menu, navigate to **Diameter > Maintenance > Applications**.
2. Delete ART rules referring to DSA.
3. Deactivate DSA using DCA Application Activate procedure. See **DCA Feature Activation**.

4. DSA Tables

DSA database schema defines various tables used to define and customize the application behavior. Table 1 lists these DSA configuration tables.

All these DSA configuration tables are SO level tables, that is, provisioning in these tables is allowed only from the SO GUI.

Note: To maintain the subscriber's states (for Stateless countermeasure business logic execution), DSA keeps subscriber's state related records in a U-SBR Generic State database indexed by the subscriber's IMSI. The database details are accessible from the **SBR > Configuration > SBR Databases** screen.

Table 1. DSA Configuration Tables

Table Name	Description	Single Row Indicator	Table Level	Table Fields
Security_Countermeasure_Config	This table includes configuration for each supported countermeasure's Type, Admin Status, Operating Mode, Result-Code, Error-Message, Vendor-ID, Continue If vulnerable and Foreign_WL_Peer_Cfg_Set.	No	SO	Table 3
Foreign_WL_Peers_Cfg_Sets	This table is used to create different set of Whitelist Foreign Peers for which countermeasure needs to be applied. Each set contains 5 list (can be increased if required) in which foreign peers can be configured.	No	SO	Table 5
System_Config_Options	This table contains common configurable options required to process various countermeasure business logic.	Yes	SO	Table 7

Table Name	Description	Single Row Indicator	Table Level	Table Fields
MCC_MNC_List	This table includes the list of MCC-MNCs of the Operator's network and its supported Roaming networks. The combined length of MCC-MNC can be either 5 digits or 6 digits long (depending upon the MNC length).	No	SO	Table 9
TTL_Config	This table contains the Time To Live (TTL) value that is used while creating/updating U-SBR records for stateful countermeasures.	No	SO	Table 11
AppIdWL_Config	This table defines the Application-ID and an associated Foreign Peer List (Foreign_WL_Peer_Cfg_Set) from which this Application-ID can be expected.	No	SO	Table 13
Realm_List	This table defines various Home and Foreign network Realms. It also allows to configure Peer List (Foreign_WL_Peer_Cfg_Set) from which these Realms can be expected.	No	SO	Table 15
VplmnORCst_Config	This table defines the list of Application-ID and its Supported Command-Code combinations.	No	SO	Table 17
SpecAVPScr_Config	This table defines the list of AVP's that needs to be screened in the incoming messages.	No	SO	Table 19
AVPInstChk_Config	This table defines the list of AVPs that needs to be screened for its instance count in the incoming messages.	No	SO	Table 21
TimeDistChk_Config	This table define minimum transition time (in minutes) between a Source-ID and Destination-ID where Source/Destination-ID can be VPLMN-ID or MCC of the VPLMN-ID.	No	SO	Table 23

Table Name	Description	Single Row Indicator	Table Level	Table Fields
MsgRateMon_Config	This table defines the Request Message Types (by specifying Application-ID and Command-Code combination) which needs to be monitored along with its threshold value.	No	SO	Table 25
AppCmdCst_Config	This table defines the Application-ID and supported Command-Codes for a given Roamer Type.	No	SO	Table 27

4.1 Configure DSA Tables

This procedure configures DSA configuration tables.

DSA configuration tables are pre-populated if DSA is configured using DSA JSON file. Refer to 3.3 Configure DSA Business Logic and Database Schema.

Alternatively DSA configuration tables can be configured manually using the following steps. See the [3] DCA Programmer's Guide for detailed information

1. From the NO GUI main menu, navigate to **DCA Framework > Diameter Security Application > Application Control**.
2. Select the newly added **DSA Version Name**.
3. Click **Config Table and Data**.

   If DSA JSON is not used to import DSA business logic and the configuration database schema, then the configured table list is empty.
4. Click **Insert**.
5. Fill out the fields to define the table.
6. Click **Add** to add multiple Table Fields.
7. Click **OK/Apply**.
8. Repeat steps 4 to 7 for each table listed in Table 1. DSA Configuration Tables.

4.2 Provision DSA Tables

This procedure imports DSA default provisioning data using DSA JSON file.

DSA SO JSON filename: **Diameter_Security_Application-Version1_Default_Config.json**

See the [3] DCA Programmer's Guide for detailed information.

1. From the SO GUI main menu, navigate to **DCA Framework > Diameter Security Application > Application Control**.
2. Select the newly added **DSA Version Name**.
3. Click **B Level Config Data** in the Import section of the Application Control page.
4. Click **Browse** and select the **Diameter_Security_Application-Version1_Default_Config.json** file.
5. Mark the **Abort on first error** checkbox to abort the import process in case of error.

6. Click **Import** to start the import process.

Apart from the default entries, additional provisioning needs to be done manually using the following procedure. See the [3] DCA Programmer's Guide for detailed information.

1. From the SO GUI main menu, navigate to **DCA Framework > Diameter Security Application > Application Control**.
2. Select the newly added **DSA Version Name**.
3. Click **Config Data**.

   If DSA JSON is not used import DSA business logic and the configuration database schema, then the configured table list is empty.
4. Select the table that needs to be provisioned.
5. Click **Provision Table**.
6. Click **Insert**.
7. Fill out the values for required fields of the table.
8. Click **OK/Apply**.

4.3 DSA Table Details

4.3.1 Security_Countermeasure_Config Table

This table is used to configure various supported countermeasures. It allows to customize the countermeasure behavior using the following options.

Table 2. Security_Countermeasure_Config Fields

Field	Description
Countermeasure Type	CounterMeasure_Type lists the countermeasure name (suffixed with their short-names).
Admin Status	Admin_Status defines the current Admin State of the countermeasure. If the Admin_Status is configured as **Enable**, then only the countermeasure business logic is executed. If the Admin_Status is configured as **Disable**, then the countermeasure business logic is not executed.
Operating Mode	Defines the action taken if a message is found to be vulnerable by the countermeasure. If the Operating_Mode is configured as **Detection_Only**, then the countermeasure works on monitoring mode. The vulnerable message is only reported to the user. DSA further processes the message (depending upon Continue If vulnerable configuration) for executing the next available countermeasure. If the Operating_Mode is configured as **Detection_And_Correction_By_Drop**, then the vulnerable diameter message is discarded at DSR and is not processed/relayed any further. If the Operating_Mode is configured as **Detection_And_Correction_By_Send_Answer**, then the vulnerable diameter message is rejected by DSR by sending an Error Answer and is not processed/relayed any further.

Field	Description
Result Code	Result_Code defines the Result Code that is added in DSA generated Error Answer message when the Operating_Mode is configured as **Detection_And_Correction_By_Send_Answer** and the message is found to be vulnerable by the countermeasure.
Error Message	Defines the error text added in DSA generated Error Answer message when the Operating_Mode is configured as **Detection_And_Correction_By_Send_Answer** and the message is found to be vulnerable by the countermeasure. If Error_Message is configured, Error-Message AVP is added with the specified error text; otherwise, no Error-Message AVP is added.
Vendor ID	Indicates the configured Result_Code is added to Result-Code AVP or Experimental-Result AVP. If Vendor_ID is configured, then the Result_Code is added to the Experimental-Result AVP with the configured Vendor_ID; otherwise, the Result_Code is added to the Result-Code AVP.
Continue If Vulnerable	Defines if the message is found to be vulnerable and Operating_Mode is **Detection_Only**, then the message is processed further by remaining countermeasures. If Continue_If_Vulnerable is configured as **Yes**, then the vulnerable message is processed by remaining countermeasures for checking more vulnerability. If Continue_If_Vulnerable is configured as **No**, then the vulnerable message is not processed further by DSA.
Foreign WL Peer Cfg Set	Foreign_WL_Peer_Cfg_Set defines the Foreign Whitelist Peer Configuration Set name (configured in Foreign_WL_Peers_Cfg_Sets Table). This configuration lists the foreign peers for which the countermeasure is executed for checking vulnerability.

Table 3 describes the field details for the Security_Countermeasure_Config Table.

Note: Upon enabling a new countermeasure, make sure the associated configuration table is configured properly for countermeasure to take effect. Any misconfiguration will lead to the countermeasure not working properly.

For both stateless and stateful countermeasures, Oracle recommends setting the **Operating Mode** parameter in the Security_Countermeasure_Config table as **Detection_Only** first to analyze and validate the configurations. This helps avoid traffic loss due to misconfiguration. Once configuration is validated, the **Operating Mode** parameter in the Security_Countermeasure_Config table can be changed as desired.

For stateful countermeasures, Oracle recommends setting the **Operating Mode** parameter in the Security_Countermeasure_Config table as **Detection_Only** for at least the first 24 hours. This allows the security application to learn about any subscribers who are already roaming in partner networks without impacting their service. The operating mode can be changed to **Detection and Correction** after that period, if desired by the operator.

Table 3. Field Details for Security_Countermeasure_Config

Field Name	Unique	Mandatory	Data type, Range, and Default Value	Description
countermeasure_Type	Yes	Yes	Enumerated Range: Application_ID_and_Command_Code_consistency_check_AppCmdCst: 1 Origin_Realm_and_Destination_Realm_whitelist_screening_RealmWLScr: 2 Subscriber_Identity_validation_SubsIdenValid: 3 Specific_AVP_screening_SpecAVPScr: 4 Origin_host_and_Origin_Realm_consistency_check_OhOrCstChk: 5 Visited_PLMN_ID_and_Origin_Realm_consistency_check_VplmnORCst: 6 Realm_and_IMSI_consistency_check_RealmIMSICst: 7 Destination_Realm_and_Origin_Realm_match_check_DrOrMatch: 8 AVP_Multiple_Instance_check_AVPInstChk: 9 Application_Id_whitelist_screening_AppIdWL: 10 Previous_Location_Check_PreLocChk: 11 Time_Distance_Check_TimeDistChk: 12 Source_Host_validation_MME_SrcHostValMme: 13 Message_rate_monitoring_MsgRateMon: 14 Source_Host_validation_HSS_SrcHostValHss: 15 Default: N/A	List of various supported countermeasures.
Admin_Status	No	Yes	Enumerated Range: Disable: 1 Enable: 2 Default: Disable	Countermeasure's Admin Status. If enabled, countermeasure is applied to the message; otherwise, skipped.

Field Name	Unique	Mandatory	Data type, Range, and Default Value	Description
Operating_Mode	No	Yes	Enumerated Range: Detection_Only: 1 Detection_And_Correction_By_Drop: 2 Detection_And_Correction_By_Send_Answer: 3 Default: Detection_Only	Countermeasure's Mode of Operation. Detection_Only: Monitor Diameter Traffic and report Diameter Vulnerabilities. Detection_And_Correction_By_Drop: Drop messages if vulnerable. Detection_And_Correction_By_Send_Answer: Send Answer if vulnerable.
Result_Code	No	No	Integer Range: 1000–5999 Default: N/A	This configuration is applicable when the countermeasure's Operating_Mode is set to Detection_And_Correction_By_Send_Answer. This value is used to set the Result-Code AVP of the Answer Message.
Error_Message	No	No	UTF8String Range: 1–64 characters Default: N/A	This configuration is applicable when the countermeasure's Operating_Mode is set to Detection_And_Correction_By_Send_Answer. If specified, the Answer Message is added with Error-Message AVP with the specified Text.
Vendor_ID	No	No	Integer Range: 1–4294967295 Default: N/A	This configuration is applicable when the Operating_Mode is set to Detection_And_Correction_By_Send_Answer. If the value is specified, the Answer Message consists of Experimental-Result grouped AVP with the specified Vendor-ID

Field Name	Unique	Mandatory	Data type, Range, and Default Value	Description
Continue_If_Vulnerable	No	Yes	Enumerated Range: No: 1 Yes: 2 Default: No	This configuration is applicable when the Operating_Mode operation mode is set to Detection_Only. Specifies if subsequent countermeasures are required to be executed for same Diameter Message, which has been tagged as vulnerable by this countermeasure.
Foreign_WL_Peer_Cfg_Set	No	Yes	UTF8String Range: 1–64 characters Default: N/A	The Whitelist Foreign Peer configuration set name (configured in Foreign_WL_Peers_Cfg_Sets Table) applicable for this countermeasure.

4.3.1.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Apart from that, below additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- The table cannot be empty at any given point of time. At least one countermeasure needs be provisioned.
- If Operating_Mode is configured as **Detection_And_Correction_By_Send_Answer**, then Result_Code needs to be configured.
- The **Foreign_WL_Peer_Cfg_Set** name needs to be configured in Foreign_WL_Peers_Cfg_Sets Table before using it in the Security_Countermeasure_Config Table.

4.3.2 Foreign_WL_Peers_Cfg_Sets Table

This table is used to configure different groups of Foreign Whitelist Peers. These peer groups are used by the below DSA tables to indicate a given configuration is applicable to a certain peer group.

- Security_Countermeasure_Config Table
- AppIdWL_Config Table
- Realm_List Table

This table groups Foreign Peers using the following options.

Table 4. Foreigh_WL_Peers_Cfg_Sets Fields

Field	Description
Whitelist Peer Configuration Set Name	WL_Peer_Cfg_Set_Name defines the Name of the Foreign Peer Group, which can be referenced by other DSA configuration tables.
Peer Lists	Defines the Foreign Peers that are part of the Foreign Peer Group. Peer_List_1, Peer_List_2, Peer_List_3 and Peer_List_4 are the fields where the foreign peers can be provisioned. Multiple fields are provided to accommodate more peers in a single group.

Note: By default each Whitelist Peer Configuration Set can hold a maximum of 310 foreign peers (provided all the Peer Names are of 32 characters). If you need to configure more than 310 foreign peers for a Whitelist Peer Configuration Set, then the schema can be enhanced by adding more columns with Name as **Peer_List_<n>** and Data type as **UTF8String**.

Table 5 describes the field details for the Foreign_WL_Peers_Cfg_Sets Table.

Table 5. Field Details for Foreigh_WL_Peers_Cfg_Sets

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
WL_Peer_Cfg_Set_Name	Yes	Yes	UTF8String Range: 1–64 characters Default: N/A	A name that uniquely identifies the Foreign Whitelist Peers configuration set. Valid Characters: A–Z, a–z, 0–9 and "_"
Peer_List_1	No	Yes	UTF8String Range: 1–2048 characters Default: N/A	The list of Foreign Peer Names (semicolon (;) separated) that are part of this configuration set.
Peer_List_2	No	No	UTF8String Range: 1–2048 characters Default: N/A	The extension list of Foreign Peer Names (semicolon (;) separated) that are part of this configuration set.
Peer_List_3	No	No	UTF8String Range: 1–2048 characters Default: N/A	The extension list of Foreign Peer Names (semicolon (;) separated) that are part of this configuration set.
Peer_List_4	No	No	UTF8String Range: 1–2048 characters Default: N/A	The extension list of Foreign Peer Names (semicolon (;) separated) that are part of this configuration set.
Peer_List_5	No	No	UTF8String Range: 1–2048 characters Default: N/A	The extension list of Foreign Peer Names (semicolon (;) separated) that are part of this configuration set.

4.3.2.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Apart from that, below additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- The table cannot be empty at any given point of time. At least one Foreign Peer Group needs be provisioned
- The allowable characters for configuring **WL_Peer_Cfg_Set_Name** are A–Z, a–z, 0–9, and underscore (_)
- The allowable separator for configuring multiple Peers is semicolon (;)
- The Peers must be a valid Diameter Peer. Navigate to the SO GUI main menu **Diameter > Configuration > Peer Nodes** for the list of valid peers.
- Duplicate Peer Names cannot be provisioned.
- If additional extension Peer List column is added for supporting more than 310 peers, then the newly added extension Peer List column name should be in the format: Peer_List_<N>

4.3.3 System_Config_Options Table

This table is used to configure various options, which customizes various countermeasure behavior.

Table 6. System_Config_Options Fields

Field	Description
MCC or VPLMN-ID	Indicates the source and destination node IDs configured in TimeDistChk_Config Table are MCCs or VPLMN-IDs. If MCC_Or_VPLMNID is configured as **MCC_Based**, then the source and destination node IDs are treated as MCC values. If MCC_Or_VPLMNID is configured as **VPLMNID_Based**, then the source and destination node IDs are treated as VPLMN-ID values.
Vulnerable If Time Distance entry Not Configured	Defines the behavior when no matching source and destination node ID is configured while executing Time-Distance Check (TimeDistChk) business logic. If vulnerable_If_TimeNotConfigured is configured as **Yes**, then the message is considered as vulnerable when no matching source and destination node is configured. If vulnerable_If_TimeNotConfigured is configured as **No**, the message is not considered as vulnerable when no matching source and destination node is configured. The message is processed further by other countermeasures (if provisioned).
Ingress Message Validation For Origin-Realm Screening	Defines the behavior to screen or not to screen the Origin-Realm AVP of the ingress diameter message for vulnerability by Origin Realm and Destination Realm Whitelist Screening (RealmWLScr). If Ingress_Msg_Chk_For_OR_Scr is configured as **Yes**, then the Origin-Realm AVP of the ingress diameter message is checked for vulnerability. If Ingress_Msg_Chk_For_OR_Scr is configured as **No**, then the Origin-Realm AVP of the ingress diameter message is not checked for vulnerability.

Field	Description
Ingress Message Validation For Destination-Realm Screening	Defines the behavior to screen or not to screen the Destination-Realm AVP of the ingress diameter message for vulnerability by Origin Realm and Destination Realm Whitelist Screening (RealmWLScr). If Ingress_Msg_Chk_For_DR_Scr is configured as **Yes**, then the Destination-Realm AVP of the ingress diameter message is checked for vulnerability. If Ingress_Msg_Chk_For_DR_Scr is configured as **No**, then the Destination-Realm AVP of the ingress diameter message is not checked for vulnerability.
Egress Message Validation For Destination-Realm Screening	Defines the behavior to screen or not to screen the Destination-Realm AVP of the egress diameter message for vulnerability by Origin Realm and Destination Realm Whitelist Screening (RealmWLScr). If Egress_Msg_Chk_For_DR_Scr is configured as **Yes**, then the Destination-Realm AVP of the egress diameter message is checked for vulnerability. If Egress_Msg_Chk_For_DR_Scr is configured as **No**, then the Destination-Realm AVP of the egress diameter message is not checked for vulnerability.
Exception Realms For OhOrCstChk	Exception_Realms_For_OhOrCstChk holds the list of Whitelist Realms. If received as Origin-Realm in the ingress diameter message, then the message is not screened by Origin Host and Origin Realm Consistency Check (OhOrCstChk) countermeasure for checking vulnerability.
Error Action if U-SBR Failure	Defines the action performed if a U-SBR failure occurs while executing the business logic of a Stateful countermeasure. If Error_Action_for_USBR_Failure is configured as **Continue_Processing**, then the message is treated as non-vulnerable by the countermeasure under process and is passed to the next countermeasure (if provisioned) to process further. If Error_Action_for_USBR_Failure is configured as **Drop**, then the message is discarded at DSR and is not processed/relayed any further.
Error Action if countermeasure's business logic execution failure	Defines the action performed if any logical error occurs while executing the countermeasure's business logic. If Error_Action_for_CmExec_Failure is configured as **Continue_Processing**, then the message is treated as non-vulnerable by the countermeasure under process and is passed to the next countermeasure (if provisioned) to process further. If Error_Action_for_CmExec_Failure is configured as **Drop**, then the message is discarded at DSR and is not processed/relayed any further.
Enable Tracing	Defines DSA tracing status. If Enable_Tracing is configured as **Yes**, then vulnerable message details are added to DSA log file. If Enable_Tracing is configured as **No**, then vulnerable message details are not added to DSA log file. See DSA Vulnerable Message Logs for more details.

Field	Description
Process_Foreign_RSR_Msg	If checked, the DSA Application will process the ingress RSR message from a foreign node. If not checked, the DSA Application will ignore the ingress RSR Message from a foreign node.

Table 7 describes the field details for the System_Config_Options Table.

Note: While the failure of a U-SBR is rare, loss of connectivity to a remote U-SBR can sometimes occur due to network fluctuations. Loss of connectivity is also treated by the DSA as a U-SBR failure and it is therefore desirable to set the value for the **Error Action if U-SBR Failure** parameter (in the System_Config_Options table) as **Continue Processing**. This ensures the requests are not dropped and roaming subscribers continue to receive service.

In the rare case of a U-SBR failure that results in loss of a significant amount of data in the database, Oracle recommends switching the Operating mode for any enabled stateful countermeasures (in the Security_Countermeasure_Config table) to **Detection_Only** for 24 hours. The setting can be reverted to its original setting after 24 hours.

Table 7. Field Details for System_Config_Options

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
MCC_Or_VPLMNID	Yes	Yes	Enumerated Range: MCC_Based: 1 VPLMNID_Based: 2 Default: MCC_Based	To check the mode of configuration for TimeDistChk_Config Table. MCC_Based: Source and Destination ID configuration is MCC based. VPLMNID_Based: Source and Destination ID configuration is VPLMNID based.
Vulnerable_If_TimeNot Configured	N/A	N/A	Boolean Range: Yes/No Default: No	To decide whether mark the message as vulnerable by Time-Distance Check (TimeDistChk) countermeasure if no matching Source and Destination ID is configured in TimeDistChk_Config Table. Yes: Mark vulnerable No: Ignore the message
Ingress_Msg_Chk_For _OR_Scr	N/A	N/A	Boolean Range: Yes/No Default: Yes	To decide whether to screen Origin-Realm for ingress Diameter Request messages for vulnerability by Origin Realm and Destination Realm whitelist screening (RealmWLScr). Yes: Check for vulnerability No: Do not check for vulnerability

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Ingress_Msg_Chk_For_DR_Scr	N/A	N/A	Boolean Range: Yes/No Default: Yes	To decide whether to screen Destination-Realm for ingress Diameter Request messages for vulnerability by Origin Realm and Destination Realm whitelist screening (RealmWLScr). Yes: Check for vulnerability No: Do not check for vulnerability
Egress_Msg_Chk_For_DR_Scr	N/A	N/A	Boolean Range: Yes/No Default: No	To decide whether to screen Destination-Realm for egress Diameter Request messages for vulnerability by Origin Realm and Destination Realm Whitelist Screening (RealmWLScr). Yes: Check for vulnerability No: Do not check for vulnerability
Exception_Realms_For_OhOrCstChk	Yes	No	UTF8String Range: 1–2048 characters Default: N/A	List of Whitelist Realms (in valid format) separated by semicolon ";" for which Origin host and Origin Realm consistency is not checked.
Error_Action_for_USBR_Failure	Yes	Yes	Enumerated Range: Continue_Processing: 1 Drop: 2 Default: Continue_Processing	Error action performed if USBR failure occurs. Continue_Processing: The message is treated as non-vulnerable and is processed further. Drop: The message is treated as vulnerable and is dropped.
Error_Action_for_CmExec_Failure	Yes	Yes	Enumerated Range: Continue_Processing: 1 Drop: 2 Default: Continue_Processing	Error action performed if countermeasure execution failed. Continue_Processing: The message is treated as non-vulnerable and is processed further. Drop: The message is treated as vulnerable and is dropped.
Enable_Tracing	N/A	N/A	Boolean Range: Yes/No Default: No	Log the message details if found vulnerable by a countermeasure. Yes: Log the message details No: Do not log the message details
Process_Foreign_RSR_Msg	N/A	N/A	Boolean Range: Yes/No Default: No	To decide whether to process RSR Message received from a Foreign Network Yes: Process RSR Message No: Don't process RSR Message

4.3.3.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- The table cannot be empty at any given point of time.
- The allowable separator for configuring multiple Realms in **Exception_Realms_For_OhOrCstChk** is semicolon (;)
- Realms configured in **Exception_Realms_For_OhOrCstChk** must be in valid Realm Format. Valid Ream Format Rules are:
  - It should consists of a list of labels separated by dot(s)
  - Each label may contain letters, digits, dashes (-) and underscore (_).
  - A label must start with a letter, digit or underscore (_) and must end with a letter or digit.
  - Underscores (_) may be used only as the first character.
  - A label must be at most 63 characters long

4.3.4 MCC_MNC_List Table

This table is used to configure the MCC-MNCs of the Home network and supported Roaming networks. The configured Home network MCC-MNCs are used to identity if the subscriber belongs to the Home network or is a Roamer. This table is also used to customize the behavior of Subscriber Identity Validation (SubsIdenValid) countermeasure.

Table 8. MCC_MNC_List Fields

Field	Description
Network Type	Indicates the type of network. If Network_Type is configured as **Home_Network**, then the configured MCC_MNC is used as Home network's MCC-MCC. If Network_Type is configured as **Foreign_Network**, then the configured MCC_MNC is used as Foreign network's MCC-MCC.
MCC-MNC	Defines a MCC-MNC combination. The value configured in MCC_MNC is treated as Home network's MCC-MNC or Foreign network's MCC-MNC depending upon the value configured in Network_Type.

Note: The MCC is always three (3) digits; however, the MNC can be two (2) digits (European standard) or three (3) digits (North American standard). The combined length of MCC-MNC can be either five (5) digits or six (6) digits (depending upon the MNC length).

Configure the MCC-MNCs with this format:

MCC 3 digit + MNC 3 digit (for example, for MCC as 310 and MNC as 150 (3 digits), the configuration is **310150**)

MCC 3 digit + MNC 2 digit (for example, for MCC as 460 and MNC as 00 (2 digits), the configuration is **46000**)

Table 9 describes the field details for the MCC_MNC_List Table.

Table 9. Field Details for MCC_MNC_List

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Network_Type	No	Yes	Enumerated Range: Home_Network: 1 Foreign_Network: 2 Default: N/A	Type of network to which this MCC_MNC belongs.
MCC_MNC	Yes	Yes	Integer Range: 10000–999999 Default: N/A	MCC+MNC of the network in format: 3 Digit MCC + 2 Digit MNC or 3 Digit MCC + 3 Digit MNC. Examples: XXXYY, where XXX is 3 digit MCC and YY is 2 digit MNC. XXXZZZ, where XXX is 3 digit MCC and ZZZ is 3 digit MNC.

4.3.4.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- At least one Home network MCC-MNC needs to be provisioned so the Roamer Type (Inbound Roamer or Outbound Roamer) can be identified for executing countermeasure business logic.
- If Subscriber Identity Validation (SubsIdenValid) countermeasure is provisioned in Security_Countermeasure_Config Table, then at least one Foreign network MCC-MCC needs to be provisioned.

4.3.5 TTL_Config Table

This table is used to configure the Time-To-Live (TTL) value for each Stateful countermeasure. The configured TTL value defines the time until the State-Data for a Stateful countermeasure is maintained in the U-SBR DB. Post expiry of TTL, the record is deleted from the U-SBR DB.

Table 10. TTL_Config Fields

Field	Description
Countermeasure Type	Countermeasure_Type lists the Stateful countermeasure Name (suffixed with their short-names)
State-Data's Time-To-Live value	State_Data_TTL defines the TTL value in seconds.

Note: If TTL is not configured for a given Stateful countermeasure, then a default TTL value (that is, 21600 seconds) is used. Refer to Section 2.5 for Stateful countermeasure specific detail TTL values.

Table 11 describes the field details for the TTL_Config Table.

Table 11. Field Details for TTL_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
countermeasure_Type	Yes	Yes	Enumerated Range: Previous_Location_Check_PreLocChk: 11 Time_Distance_Check_TimeDistChk: 12 Source_Host_validation_MME_SrcHostValMme: 13 Message_rate_monitoring_MsgRateMon: 14 Source_Host_validation_HSS_SrcHostValHss: 15 Default: N/A	List of various supported Stateful countermeasures.
State_Data_TTL	No	Yes	Integer Range: 60–604800 Default: N/A	The TTL value for the countermeasure's state-data record stored in the U-SBR, in seconds.

4.3.5.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. No additional validation is required for this table.

4.3.6 AppIdWL_Config Table

This table is used to customize the behavior of Application-ID Whitelist Screening (AppIdWL) countermeasure by using these options.

Table 12. AppIdWL_Config Fields

Field	Description
Application ID	Application_ID defines diameter Application-ID.
Foreign WL Peer Cfg Set	Foreign_WL_Peer_Cfg_Set defines the Foreign Whitelist Peer Configuration Set name (configured in Foreign_WL_Peers_Cfg_Sets Table). This configuration lists the foreign peers from which diameter message can be received with the configured Application_ID. If "*" is configured then the configured Application_ID can be received from any peer.

Table 13 describes the field details for the AppIdWL_Config Table.

Table 13. Field Details for AppIdWL_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Application_ID	Yes	Yes	Integer Range: 0–4294967295 Standard Application-IDs: 0–16777215 Vendor specific Application-IDs: 16777216–4294967294 Relay: 4294967295 Default: N/A	Application-ID is used to identify a specific Diameter Application.
Foreign_WL_Peer_Cfg_Set	No	Yes	UTF8String, Range: 1–64 characters Default: "*"	The White List Peer Configuration set to which this Application-ID and Command-Code combination is applicable. If only "*" is configured, then applicable to all peers.

4.3.6.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- If Application-ID Whitelist Screening (AppIdWL) countermeasure is provisioned in Security_Countermeasure_Config Table, then this table cannot be empty. At least one entry needs be provisioned.
- For values other than "*" in **Foreign_WL_Peer_Cfg_Set**, the configuration set name needs to be configured in Foreign_WL_Peers_Cfg_Sets Table before using it in Security_Countermeasure_Config Table.
- Both "*" and a configuration set name cannot be provisioned in **Foreign_WL_Peer_Cfg_Set**.

4.3.7 Realm_List Table

This table is used to configure Realms of the Home network and supported Roaming networks. The configured Home network Realm identifies an egress Diameter Message generated by the Home network, which is sent to a foreign network. This table is also used to customize the behavior of Origin Realm and Destination Realm Whitelist Screening (RealmWLScr) countermeasure.

Table 14. Realm_List Fields

Field	Description
Network Type	Indicates the type of network. If Network_Type is configured as **Home_Network**, then the configured Realm is used as Home network's Realm. If Network_Type is configured as **Foreign_Network**, then the configured Realm is used as Foreign network's Realm.
Realm	Defines the Realm.
Foreign WL Peer Cfg Set	Foreign_WL_Peer_Cfg_Set defines the Foreign Whitelist Peer Configuration Set name (configured in Foreign_WL_Peers_Cfg_Sets Table). This configuration lists the foreign peers from which diameter message can be received with the configured Realm. If "*" is configured then the configured Realm can be received from any peer.

Table 15 describes the field details for the Realm_List Table.

Table 15. Field Details for Realm_List

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Network_Type	No	Yes	Enumerated Range: Home_Network: 1 Foreign_Network: 2 Default: N/A	Type of network to which this Realm belongs.
Realm	Yes	Yes	UTF8String Range: 1–255 characters Default: N/A	Realm (in valid format). Realm consists of labels separated by dots. Each label (max 63 chars) may contain a–z, A–Z, 0–9, "–" & "_" (only as 1st char) and must not start with "–" or ends with "–" & "_".
Foreign_WL_Peer_Cfg_Set	No	Yes	UTF8String Range: 1–64 characters Default: "*"	The White List Peer Configuration set to which this Realm screening is applicable. If only "*" is configured, then applicable to all Peers.

4.3.7.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- At least one Home network Realm needs to be configured so an egress Diameter Message generated by the Home network, which is sent to a foreign network can be identified.

- Realm configured in **Realm** must be in valid format. Valid Realm format rules are:
  - It should consist of a list of labels separated by dot(s).
  - Each label may contain letters, digits, dashes (–) and underscore (_).
  - A label must start with a letter, digit, or underscore (_) and must end with a letter or digit.
  - Underscores (_) may be used only as the first character.
  - A label must be at most 63 characters long.
- For values other than "*" in **Foreign_WL_Peer_Cfg_Set**, the configuration set name needs to be configured in Foreign_WL_Peers_Cfg_Sets Table before using it in MCC_MNC_List Table.
- Both "*" and a configuration set name cannot be provisioned in **Foreign_WL_Peer_Cfg_Set**.

4.3.8 VplmnORCst_Config Table

This table is used to customize the behavior of Visited-PLMN-ID and Origin-Realm Consistency Check (VplmnORCst) countermeasure by using the following options.

Table 16. VplmnORCst_Config

Field	Description
Application ID	Application_ID defines diameter Application-ID.
Command Codes	Command_Codes defines the list of supported Command-Codes (semicolon ";" delimited) for the given Application_ID.

Table 17 describes the field details for the VplmnORCst_Config Table.

Table 17. Field Details for VplmnORCst_Config

Field Name	Unique	Mandatory	Data type, Range, and Default Value	Description
Application_ID	Yes	Yes	Integer Range: 0–4294967295 Standard Application-IDs: 0–16777215 Vendor specific Application-IDs: 16777216–4294967294 Relay: 4294967295 Default: N/A	Application-ID is used to identify a specific Diameter Application.
Command_Codes	No	Yes	UTF8String Range: 1–2048 characters Valid Command-Code Range: 0–16777215 Default: N/A	List of Command-Codes supported for the given Application-ID (semicolon (;) separated).

4.3.8.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- If Visited-PLMN-ID and Origin-Realm Consistency Check (VplmnORCst) countermeasure is provisioned in Security_Countermeasure_Config Table, then this table cannot be empty. At least one entry needs be provisioned.
- The allowable separator for configuring multiple Command-Codes in **Command_Codes** is semicolon (;).
- Command-Codes configured in **Command_Codes** must be in valid Format. Valid Command-Code Range is 0–16777215.
- A Command-Code can be configured only once for a given Application-ID. No duplicate Command-Code is allowed.

4.3.9 SpecAVPScr_Config Table

This table is used to customize the behavior of Specific AVP Screening (SpecAVPScr) countermeasure by using the following options.

Table 18. SpecAVPScr_Config Fields

Field	Description
Application ID	Application_ID defines diameter Application-ID.
Command Code	Command_Code defines the supported Command-Codes for the given Application_ID.
Message Type	Defines the type of diameter message. If Message_Type is configured as **Request**, then the given configuration is applicable to only diameter Request messages. If Message_Type is configured as **Answer**, then the given configuration is applicable to only diameter Answer messages. If Message_Type is configured as **Both**, then the given configuration is applicable to both diameter Request and Answer messages.
AVP Name	AVP_Name defines the name of the AVP. This AVP Name should match exactly (case sensitive) with the Name configured in Diameter AVP dictionary (Refer to the SO GUI Main Menu **Diameter > AVP Dictionary > All-AVP Dictionary**). Grouped AVP name can be defined with its Parent AVP Names (Max 5 level including the child AVP) separated by semicolon (;). For example: • Parent1AVPName;AVPName. • Parent1AVPName;Parent2AVPName;AVPName. • Parent1AVPName;Parent2AVPName;Parent3AVPName;Parent4AVPName;AVPName.
AVP Data Type	AVP_Value_Type defines the type of the data configured in AVP_Value. Depending upon the configured data type, the value configured in AVP_Value is used. Support data types are OctetString, Integer32, Integer64, Unsigned32, Unsigned64, Float32, Float64, Address, Time, UTF8String, Diameter-Identity, Diameter-URI, and Enumerated. In case of Grouped AVP, only the data type of the child AVP needs to be configured.
AVP Value	Defines the AVP value used during screening. The value is type casted used as per the configured AVP_Value_Type. ***Note:*** For **Enumerated** AVP_Value_Type, provision the Integer value of the Enumerated AVP as present in the Enumerations MO (Refer to the SO GUI Main Menu **Diameter > Mediation > Enumerations**).

Table 19 describes the field details for the SpecAVPScr_Config Table.

Table 19. Field Details for SpecAVPScr_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Application_ID	No	Yes	Integer Range: 0–4294967295 Standard Application-IDs: 0–16777215 Vendor specific Application-IDs: 16777216–4294967294 Relay: 4294967295 Default: N/A	Application-ID is used to identify a specific Diameter Application.
Command_Code	No	Yes	Integer Range: 0–16777215 Default: N/A	Command-Code for the given Application-ID.
Message_Type	No	Yes	Enumerated Range: Request: 1 Answer: 2 Both: 3 Default: N/A	Message Type for which the configuration is applicable.
AVP_Name	No	Yes	UTF8String Range: 1–1279 characters Default: N/A	Name of the AVP as per Diameter AVP Dictionary. AVPs that are part of Grouped AVP can be defined along with its Parent AVP Names (Max 5 level) separated by ";". For example, BaseAVPName;SubAVPName;AVPName. Each AVP name cannot exceed 255 characters.
AVP_Value_Type	No	Yes	Enumerated Range: OctetString: 1 Integer32: 2 Integer64: 3 unsigned32: 4 unsigned64: 5 Float32: 6 Float64: 7 Address: 8 Time: 9 UTF8String: 10 DiameterIdentity: 11 DiameterURI: 12 Enumerated: 13 Default: N/A	Data type of the AVP value.

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
AVP_Value	No	Yes	UTF8String Range: 1–2048 characters Default: N/A	Value of the AVP that needs to be screened.

4.3.9.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Apart from that, below additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- If Specific AVP Screening (SpecAVPScr) countermeasure is provisioned in Security_Countermeasure_Config Table then this table cannot be empty. At least one entry needs be provisioned.
- An AVP Name can be configured only once for a given Application-ID, Command-Code, Message-Type and AVP Value combination. No Duplicate AVP Name is allowed.
- If an AVP-Name is configured with a given Application-ID, Command-Code and AVP Value combination with Message-Type as **Both**, then same combination cannot be configured again with Message_Type as **Request** or **Answer**.
- The allowable characters for an AVP Name are A–Z, a–z, 0–9, dash "–", underscore "_", parentheses "()", and dot "."
- The allowable separator for configuring Grouped AVP in **AVP_Name** is semicolon (;). For example:
  - Parent1AVPName;AVPName.
  - Parent1AVPName;Parent2AVPName;AVPName.
  - Parent1AVPName;Parent2AVPName;Parent3AVPName;Parent4AVPName;AVPName.
- A maximum of 5 level deep Grouped AVP is supported. I.e. a Grouped AVP can have at max four parents.
- The configured AVP Value should be in-line with the configured AVP data type. E.g. If the **AVP_Value_Type** is provisioned as **OctetString** then the value configured in **AVP_Value** must be of OctetString type.

4.3.10 AVPInstChk_Config Table

This table is used to customize the behavior of AVP Multiple Instance Check (AVPInstChk) countermeasure by using the following options.

Table 20. AVPInstChk_Config Fields

Field	Description
Application ID	Application_ID defines diameter Application-ID.
Command Code	Command_Code defines the supported Command-Codes for the given Application_ID.
Message Type	Defines the type of diameter message. If Message_Type is configured as **Request**, then the given configuration is applicable to only diameter Request messages. If Message_Type is configured as **Answer**, then the given configuration is applicable to only diameter Answer messages. If Message_Type is configured as **Both**, then the given configuration is applicable to both diameter Request and Answer messages.

Field	Description
AVP Name	AVP_Name defines the name of the AVP. This AVP Name should match exactly (case sensitive) with the Name configured in Diameter AVP dictionary (Refer to the SO GUI Main Menu **Diameter > AVP Dictionary > All-AVP Dictionary**). Grouped AVP name can be defined with its Parent AVP Names (Maximum of five (5) levels including the child AVP) separated by semicolon (;). For example: • Parent1AVPName;AVPName. • Parent1AVPName;Parent2AVPName;AVPName. • Parent1AVPName;Parent2AVPName;Parent3AVPName;Parent4AVPName;AVPName.
Minimum Number of Instance	Minimum_Instance defines the minimum number of instances of the AVP in the incoming diameter message.
Maximum Number of Instance	Maximum_Instance defines the maximum number of instances of the AVP in the incoming diameter message.

Table 21 describes the field details for the AVP Multiple Instance Check (AVPInstChk).

Table 21. Field Details for AVPInstChk_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Application_ID	No	Yes	Integer Range: 0–4294967295 Standard Application-IDs: 0–16777215 Vendor specific Application-IDs: 16777216–4294967294 Relay: 4294967295 Default: N/A	Application-ID is used to identify a specific Diameter Application.
Command_Code	No	Yes	Integer Range: 0–16777215 Default: N/A	Command-Code for the given Application-ID.
Message_Type	No	Yes	Enumerated Range: Request: 1 Answer: 2 Both: 3 Default: N/A	Message Type for which the configuration is applicable.
AVP_Name	No	Yes	UTF8String Range: 1–1279 characters Default: N/A	Name of the AVP as per Diameter AVP Dictionary. AVPs that are part of Grouped AVP can be defined along with its Parent AVP Names (Max 5 level) separated by ";". For example, BaseAVPName;SubAVPName;AVP Name. Each AVP name cannot exceed 255 characters.

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Minimum_Instance	No	Yes	Integer Range: 0–25 Default: N/A	Minimum allowed instances of the given AVP in the diameter message. 0 instance means the AVP should not present in the message.
Maximum_Instance	No	Yes	Integer Range: 0–25 Default: N/A	Maximum allowed instances of the given AVP in the diameter message. 0 instance means the AVP should not present in the message.

4.3.10.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- If AVP Multiple Instance Check (AVPInstChk) countermeasure is provisioned in Security_Countermeasure_Config Table then this table cannot be empty. At least one entry needs be provisioned.
- An AVP Name can be configured only once for a given Application-ID, Command-Code and Message-Type combination. No Duplicate AVP Name is allowed.
- If an AVP-Name is configured with a given Application-ID and Command-Code combination with Message-Type as **Both**, then same combination cannot be configured again with Message_Type as **Request** or **Answer**.
- The allowable characters for an AVP Name are A–Z, a–z, 0–9, dash "–", underscore "_", parentheses "()", and dot "."
- The allowable separator for configuring grouped AVP in **AVP_Name** is semicolon (;). For example:
  - Parent1AVPName;AVPName.
  - Parent1AVPName;Parent2AVPName;AVPName.
  - Parent1AVPName;Parent2AVPName;Parent3AVPName;Parent4AVPName;AVPName.
- A maximum of 5 level deep Grouped AVP is supported, for example, a Grouped AVP can have a maximum of four parents.
- The value configured in **Maximum_Instance** cannot be less than the value configured in **Minimum_Instance**.

4.3.11 TimeDistChk_Config Table

This table is used to customize the behavior of Time-Distance Check (TimeDistChk) countermeasure by using the following options.

Table 22. TimeDistChk_Config Fields

Field	Description
Source and Destination Node-IDs	Defines the two Node-ID values. Node-ID can be MCC or VPLMN-ID of any given network. Value of MCC_Or_VPLMNID configured in System_Config_Options Table determines that the configured Node-IDs is MCCs or VPLMN-IDs. If MCC_Or_VPLMNID is configured as **MCC_Based**, then the Node_ID_1 and Node_ID_2 are treated as MCC values. If MCC_Or_VPLMNID is configured as **VPLMNID_Based**, then the Node_ID_1 and Node_ID_2 are treated as VPLMN-ID values.
Minimum Transition Time	Defines the minimum transition time (in minutes) required to move between Node_ID_1 and Node_ID_2.

Table 23 describes the field details for the TimeDistChk_Config Table.

Table 23. Field Details for TimeDistChk_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Node_ID_1	No	Yes	UTF8String Range: 6 Integer: 3 digits OctetString: 6 digits Default: N/A	MCC in digits or VPLMN-ID in OctetString. MCC_Or_VPLMNID configured in System_Config_Options Table is used to determine configured Node_ID is MCCs or VPLMN-IDs.
Node_ID_2	No	Yes	UTF8String Range: 6 Integer: 3 digits OctetString: 6 digits Default: N/A	MCC in digits or VPLMN-ID in OctetString. MCC_Or_VPLMNID configured in System_Config_Options Table is used to determine configured Node_ID is MCCs or VPLMN-IDs.
Minimum_Transition_Time	No	Yes	Integer Range: 1–4320 Default: N/A	Minimum Transition time (in minutes) between the Node_id_1 and Node_id_2.

4.3.11.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Apart from that, below additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- If Time-Distance Check (TimeDistChk) countermeasure is provisioned in Security_Countermeasure_Config Table then this table cannot be empty. At least one entry needs be provisioned.
- Each **Node_ID_1** and **Node_ID_2** combination must be unique. No duplicate **Node_ID_1** and **Node_ID_2** combination is allowed. Not even by swapping **Node_ID_1** and **Node_ID_2** values.

  For example, an entry with Node_ID_1=10 and Node_ID_2=20 and another entry with Node_ID_1=20 and Node_ID_2=10 is not allowed.

- At any given point of time, all the **Node_ID_1** and **Node_ID_2** combinations can have either MCCs of networks or VPLMN-IDs of networks (determined by the value configured in **MCC_Or_VPLMNID** of System_Config_Options Table). But not both.
- Valid MCC value range for **Node_ID_1** and **Node_ID_2** is 100 to 999. This validation is performed when **MCC_Or_VPLMNID** is configured as **MCC_Based** in System_Config_Options Table.
- Valid VPLMN-ID value range for **Node_ID_1** and **Node_ID_2** is a 6-digit OctetString with allowed digits of 0–9 and **F**. This validation is performed when **MCC_Or_VPLMNID** is configured as **VPLMNID_Based** in System_Config_Options Table.

  Note: **F** is allowed to act as filler for 2 digits MNC. Therefore, if **F** is present, it must be in the 3rd byte string. Format of the Visited-PLMN-Id defined in TS 3GPP TS 29.272.

4.3.12 MsgRateMon_Config Table

This table is used to customize the behavior of Message Rate Monitoring (MsgRateMon) countermeasure by using the following options.

Table 24. MsgRateMon_Config Fields

Field	Description
Application ID	Application_ID defines diameter Application-ID.
Command Code	Command_Code defines the supported Command-Codes for the given Application_ID.
Message Threshold	Message_Threshold define the maximum allowable incoming diameter message Rate for the given Application_ID and Command_Code combination.

Table 25 describes the field details for the MsgRateMon_Config Table.

Table 25. Field Details for MsgRateMon_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Application_ID	No	Yes	Integer Range: 0–4294967295 Standard Application-IDs: 0–16777215 Vendor specific Application-IDs: 16777216–4294967294 Relay: 4294967295 Default: N/A	Application-ID is used to identify a specific Diameter Application.
Command_Code	No	Yes	Integer Range: 0–16777215 Default: N/A	Command-Code for the given Application-ID.
Message_Threshold	No	Yes	Integer Range: 1–50000 Default: 1000	The maximum threshold value to mark the message as vulnerable if the current ingress request rate for this Application-id/Command-Code combination exceeds the configured threshold value.

4.3.12.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Apart from that, below additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- An Application-ID and Command-Code combination can be configured only once. No Duplicate Application-ID and Command-Code combination is allowed.
- If Message Rate Monitoring (MsgRateMon) countermeasure is provisioned in Security_Countermeasure_Config Table, then this table cannot be empty. At least one entry needs to be provisioned.

4.3.13 AppCmdCst_Config Table

This table is used to customize the behavior of Application-ID and Command-Code Consistency Check (AppCmdCst) countermeasure by using the following options.

Table 26. AppCmdCst_Config Fields

Field	Description
Roamer Type	Defines the type of Roamer to which this configuration is applicable. If Roamer_Type is configured as **Outbound_Roamer**, then the given configuration is applicable to the Foreign network subscribers who are currently Roaming in this Home network. If Roamer_Type is configured as **Inbound_Roamer**, then the given configuration is applicable to the Home network subscribers who are currently Roaming in a Foreign network.
Application ID	Application_ID defines diameter Application-ID.
Command Code	Command_Code defines the supported Command-Codes for the given Application_ID.

Table 27 describes the field details for the AppCmdCst_Config Table.

Table 27. Field Details for AppCmdCst_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Roamer_Type	No	Yes	Enumerated Range: Inbound_Roamer: 1 Outbound_Roamer: 2 Default: N/A	Type of Roamer to which this configuration is applicable.
Application_ID	No	Yes	Integer Range: 0–4294967295 Standard Application-IDs: 0–16777215 Vendor specific Application-IDs: 16777216–4294967294 Relay: 4294967295 Default: N/A	Application-ID is used to identify a specific Diameter Application.

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Command_Codes	No	Yes	UTF8String Range: 1–2048 characters Valid Command-Code Range: 0–16777215 Default: N/A	List of Command-Codes supported for the given Application-ID (semicolon (;) separated).

4.3.13.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Additional validation is performed during DSA business logic script compilation. If validation fails, the compilation also fails and **Event #33309** is raised with appropriate error text. See the [3] DCA Programmer's Guide for script compilation.

- If Application-ID and Command-Code Consistency Check (AppCmdCst) countermeasure is provisioned in Security_Countermeasure_Config Table then this table cannot be empty. At least one entry needs be provisioned.
- An Application-ID can be configured only once for a given Roamer Type. No Duplicate Application-ID is allowed.
- The allowable separator for configuring multiple Command-Codes in **Command_Codes** is semicolon (;).
- Command-Codes configured in **Command_Codes** must be in valid format. Valid Command-Code range is 0–16777215.
- A Command-Code can be configured only once for a given Application-ID. No Duplicate Command-Code is allowed.

5. DSA MEALs

DSA MEALs defines various Measurements, SysMetric, and Alarms used for reporting the application behavior. All these DSA MEALs are defined using **DCA Custom MEAL Framework**.

5.1 Configure DSA MEALs

This procedure configures DSA MEALs.

DSA MEALs are pre-populated if DSA is configured using DSA JSON file. Refer to Configure DSA Business Logic and Database Schema.

Alternatively, DSA MEALs can be configured manually using the following steps. See *DCA Programmer's Guide* for detailed information

1. From the NO GUI main menu, navigate to **DCA Framework > Diameter Security Application > Custom MEALs**.
2. Click **Insert**.
3. Fill in the fields to define the MEAL.
4. Click **OK/Apply**.
5. Repeat steps 2 to 4 for each MEAL defined in Table 28, Table 29, Table 30, Table 31, Table 32, Table 33, Table 34 and Table 35.

5.2 Measurement

5.2.1 ProcessedBy<Countermeasure ShortName>

This Measurement is used to report the number of diameter messages screened by a countermeasure. Table 28 defines the list of Measurement name for each countermeasure type.

Table 28. ProcessedBy<Countermeasure ShortName> Measurement

Countermeasure Type	Measurement Names
Measurement Name	ProcessedByAppIdWL ProcessedByAppCmdCst ProcessedByRealmWLScr ProcessedByOhOrCstChk ProcessedByDrOrMatch ProcessedByVplmnORCst ProcessedByRealmIMSICst ProcessedBySubsIdenValid ProcessedBySpecAVPScr ProcessedByAVPInstChk ProcessedByMsgRateMon ProcessedByTimeDistChk ProcessedByPreLocChk ProcessedBySrcHostValHss ProcessedBySrcHostValMme
Template Type	Counter
Measurement Type	Scalar

5.2.2 DetectedBy<Countermeasure ShortName>

This Measurement is used to report number of diameter message found to be vulnerable by a countermeasure while the countermeasure operating in **Detection Only** mode. Below table defines the list of Measurement name for each countermeasure type.

Table 29. DetectedBy<Countermeasure ShortName> Measurement

Countermeasure Type	Measurement Names
Measurement Name	DetectedByAppIdWL DetectedByAppCmdCst DetectedByRealmWLScr DetectedByOhOrCstChk DetectedByDrOrMatch DetectedByVplmnORCst DetectedByRealmIMSICst DetectedBySubsIdenValid DetectedBySpecAVPScr DetectedByAVPInstChk DetectedByMsgRateMon

Countermeasure Type	Measurement Names
	DetectedByTimeDistChk DetectedByPreLocChk DetectedBySrcHostValHss DetectedBySrcHostValMme
Template Type	Counter
Measurement type	Scalar

5.2.3 DroppedBy<Countermeasure ShortName>

This Measurement is used to report number of diameter message found to be vulnerable by a countermeasure while the countermeasure operating in **Detection_And_Correction_By_Drop** mode. Below table defines the list of Measurement name for each countermeasure type.

Table 30. DroppedBy<Countermeasure ShortName> Measurement

Countermeasure Type	Measurement Names
Measurement Name	DroppedByAppIdWL DroppedByAppCmdCst DroppedByRealmWLScr DroppedByOhOrCstChk DroppedByDrOrMatch DroppedByVplmnORCst DroppedByRealmIMSICst DroppedBySubsIdenValid DroppedBySpecAVPScr DroppedByAVPInstChk DroppedByMsgRateMon DroppedByTimeDistChk DroppedByPreLocChk DroppedBySrcHostValHss DroppedBySrcHostValMme
Template Type	Counter
Measurement type	Scalar

5.2.4 RejectedBy<Countermeasure ShortName>

This Measurement is used to report number of diameter message found to be vulnerable by a countermeasure while the countermeasure operating in **Detection_And_Correction_By_Send_Answer** mode. Below table defines the list of Measurement name for each countermeasure type.

Table 31. RejectedBy<Countermeasure ShortName> Measurement

Countermeasure Type	Measurement Names
Measurement Name	RejectedByAppIdWL RejectedByAppCmdCst RejectedByRealmWLScr RejectedByOhOrCstChk RejectedByDrOrMatch RejectedByVplmnORCst RejectedByRealmIMSICst RejectedBySubsIdenValid RejectedBySpecAVPScr RejectedByAVPInstChk RejectedByMsgRateMon RejectedByTimeDistChk RejectedByPreLocChk RejectedBySrcHostValHss RejectedBySrcHostValMme
Template Type	Counter
Measurement type	Scalar

5.2.5 FailedExec<Countermeasure ShortName>

This Measurement is used to report number of diameter message failed to screen by a countermeasure due to error in executing the countermeasure's business logic. E.g. failure due to U-SBR DB not available, Runtime/Internal errors, Roamer type cannot be determined due to unavailability of User-Name AVP etc. Below table defines the list of Measurement name for each countermeasure type.

Table 32. FailedBy<Countermeasure ShortName> Measurement

Countermeasure Type	Measurement Names
Measurement Name	FailedExecAppIdWL FailedExecAppCmdCst FailedExecRealmWLScr FailedExecOhOrCstChk FailedExecDrOrMatch FailedExecVplmnORCst FailedExecRealmIMSICst FailedExecSubsIdenValid FailedExecSpecAVPScr FailedExecAVPInstChk FailedExecMsgRateMon FailedExecTimeDistChk FailedExecPreLocChk FailedExecSrcHostValHss FailedExecSrcHostValMme

Countermeasure Type	Measurement Names
Template Type	Counter
Measurement type	Scalar

5.3 SysMetric

5.3.1 VulnerableBy<Countermeasure ShortName>

This SysMetric is used to report the vulnerable message rate detected by a countermeasure. Depending upon the configured threshold value, Critical, Major, or Minor alarm are also raised. Below table defines the list of Sysmetric name for each countermeasure type.

Table 33. VulnerableBy<Countermeasure ShortName> SysMetric

Countermeasure Type	Measurement Names
Measurement Name	VulnerableByAppIdWL VulnerableByAppCmdCst VulnerableByRealmWLScr VulnerableByOhOrCstChk VulnerableByDrOrMatch VulnerableByVplmnORCst VulnerableByRealmIMSICst VulnerableBySubsIdenValid VulnerableBySpecAVPScr VulnerableByAVPInstChk VulnerableByMsgRateMon VulnerableByTimeDistChk VulnerableByPreLocChk VulnerableBySrcHostValHss VulnerableBySrcHostValMme
Template Type	Rate
Measurement type	Scalar
KPI Description	Average number of vulnerable messages detected by <Countermeasure LongName>
Generate Alarm	Yes
Alarm Description	The Number of vulnerable messages detected by <Countermeasure LongName> is approaching its maximum Threshold.
100% Threshold Value	10000
Alarm Minor Set Threshold	40
Alarm Minor Clear Threshold	30
Alarm Major Set Threshold	60

Countermeasure Type	Measurement Names
Alarm Major Clear Threshold	50
Alarm Critical Set Threshold	80
Alarm Critical Clear Threshold	70
Template Type	Rate
Measurement type	Scalar
KPI Description	Average number of vulnerable messages detected by <Countermeasure LongName>
Generate Alarm	Yes
Alarm Description	The Number of vulnerable messages detected by <Countermeasure LongName> is approaching its maximum Threshold.

5.3.2 MsgRatePerPeer

This SysMetric is used to internally by Message Rate Monitoring (MsgRateMon) countermeasure to compute the Rate at which ingress diameter request message is received (for each ingress peer, Application-ID and Command-Code combination). See 2.5.1 for more details.

Table 34. MsgRatePerPeer SysMetric

Countermeasure Type	Measurement Names
Measurement Name	MsgRatePerPeer
Template Type	Rate
Measurement type	Arrayed
KPI Description	Rate (Indexed by per ingress peer, Application-ID and Command-Code combination) at which ingress diameter request messages are getting processed by Message Rate Monitoring (MsgRateMon) countermeasure.
Generate Alarm	No

5.4 <Countermeasure ShortName>ExecFailed Alarm

This Alarm is used to report any failure occur in countermeasure's business logic execution which may result in traffic loss. The Alarm is auto cleared in 90 second, if the problem still persists after 90 second, alarm is raised again. Below table defines the list of Alarm name for each countermeasure type.

Note: These alarms represent DSA Custom MEALS that are referred by logical names assigned to them and not by any event or alarm number. Append "Alrm" as a suffix to each measurement in the following table.

Table 35. <Countermeasure ShortName>ExecFailed Alarm

Countermeasure Type	Measurement Names
Measurement Name	AppIdWLExecFailed<Alrm as a suffix> AppCmdCstExecFailed<Alrm as a suffix> RealmWLScrExecFailed<Alrm as a suffix> OhOrCstChkExecFailed<Alrm as a suffix> DrOrMatchExecFailed<Alrm as a suffix> VplmnORCstExecFailed<Alrm as a suffix> RealmIMSICstExecFailed<Alrm as a suffix> SubsIdenValidExecFailed<Alrm as a suffix> SpecAVPScrExecFailed<Alrm as a suffix> AVPInstChkExecFailed<Alrm as a suffix> MsgRateMonExecFailed<Alrm as a suffix> TimeDistChkExecFailed<Alrm as a suffix> PreLocChkExecFailed<Alrm as a suffix> SrcHostValHssExecFailed<Alrm as a suffix> SrcHostValMmeExecFailed<Alrm as a suffix>
Template Type	Event
Alarm Description	Failed executing <Countermeasure LongName> business logic. Disable the countermeasure until the problem is resolved.
Alarm Autoclear Interval	180
Alarm Throttling Interval	60

6. DSA Vulnerable Message Logs

Option has been provided to log vulnerable message details into a log file on MPs. The active SO collects these log files from the MPs and dumps it in SO.

MPs creates the file containing vulnerable message details on **/var/TKLC/db/filemgmt/dca_logs**

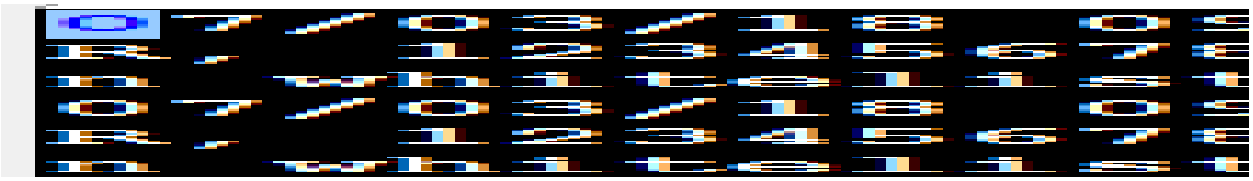
- Each vulnerable message detail can be of maximum of 2000 characters.
- Each log file can contain a maximum of 30000 vulnerable message details. Also each log file is open for a maximum of 1 hour for logging. Once the maximum number of entries is logged into a log file or on the expiry of the 1 hour timeout, the file gets closed for logging and a new log file is created for subsequent logs.
- MPs suspends logging if the available disk space of /var/TKLC/db/filemgmt/dca_logs on MP is less than 30%. The logging resumes again once the available disk space increases.
- MPs also suspends logging if the vulnerable message logging rate is above 25000 per second. The logging resumes again once the vulnerable message logging rate decreases.

- An Alarm is raised to notify the user if the logging is suspended on the MP(s). The alarm gets cleared once the logging resumes.
- Naming Convention of Log File on DAMP is:

  [DCA AppShort Name] + [Task Id] + "_" + [start time] + "-" + [End Time]+"_"+ "_logs.csv"

  For example : "DSA4_1527243681-1527247282_logs.csv"

  The snapshot of a sample log file:



The active SO collects the closed log file from the MP and saves them on **/var/TKLC/db/filemgmt/export/dsa**

- The active SO suspends collecting the logs from MP if the available disk space of /var/TKLC/db/filemgmt/export/dsa on active SO is less than 30%. The collection resumes again once the available disk space increases.
- The active SO also suspends collecting the logs from MP if any error occurs during the log collection process. The collection resumes again once the error is resolved.
- An alarm is raised to notify the user if log collection is suspended on SO due to any error. The alarm gets cleared once the error is resolved.
- The log file shall have the value of "Time Stamp, Application-Id, Command-Code, Origin-Realm, Origin-Host, Destination-Realm, Destination-Host, Applied Action (Discarded/Rejected/Detected), Message Type (Request or Response), Applied CounterMeasure Name, Session-Id, Ingress Peer name and Subscriber- Type" in comma separated format. The message shall contain only field value and no field name.
- Naming Convention of Log File on Active SOAM is:

  [DAMP Server Name] + [Time Stamp]+ "_dsa.tar.gz"

6.1 Configure Vulnerable Message Logging

By default, logging of vulnerable message logging is disabled (Refer to Enable Tracing option of System_Config_Options Table). Before enabling logging, the below steps needs to be performed on all the Server under the SO Server Group (active/standby/spare) where DSA is running.

1. Log into the SO server using SSH as **admusr**.
2. Create a directory and copy the below list of files into it.
   - fetchLogDsa.sh
   - fetchLogDsa.ini
   - configureScriptAndCronJob.sh
   - dsa_application.cron
   - dsa_application_log_rotate
3. Change the permission of the file as below

   command: chmod 744 configureScriptAndCronJob.sh dsa_application.cron dsa_application_log_rotate fetchLogDsa.ini fetchLogDsa.sh

4. Execute the script configureScriptAndCronJob.sh

   command: sh configureScriptAndCronJob.sh

   This script sets up a cron job task that runs periodically to fetch the log files from MPs and dumps it the SO.

5. Execute step 1 to 4 on the active, standby, and spare SO where DSA is running.

Note: The active SO collects the closed log files and save them at /var/TKLC/db/filemgmt/export/dsa. The user needs to move the old log files to a different server to free up space for new log files. Alternatively the **Data Export** feature can be used to regularly transfer these log files out of the file management area to a remote server using rsyn

This procedure configures a **Data Export Job**. See **DSR Online Help** for more details.

1. From the SO GUI main menu, navigate to **Administration > Remote Servers > Data Export**.
2. Click **Insert**.
3. Enter a **Task Name**.
4. Enter a **Task Description**.
5. Enter a **Remote Server** Name, IPv4, or IPv6 address.
6. Enter the **User name** of the Remote Server.
7. Enter the **Directory on Export Server** (the target directory path on the Remote Server).
8. Enter the **Path to Rsync** on the remote server (Optional).
9. Enter **/export/dsa/*** as the **Files to Transfer**.
10. Select the appropriate **Upload Frequency**.
11. Click **OK** to apply the changes.

Table 36. fetchLogDsa.ini File Configuration Options

Field	Description
destinationPath	The destination path to which the script[fetchLogDsa.sh] copies the log files from the MPs to the active SO on which it is running. Default: /var/TKLC/db/filemgmt/export/dsa
logDirectory	The directory where the log files of the script are going to be stored. This file can be used for debugging the script, that is, if something goes wrong in the script. Default: /var/TKLC/log
freeSpace	The required free space (in %). To run this script, the SO must have that much % free space in that file system. Default: 30
maxLogFilesToTar	Maximum number of files that can be compressed at a time. Default: 1500

6.2 dsa_application.cron File Script and Log

Run the script /var/TKLC/db/filemgmt/dca/fetchLogDsa.sh every 5 minutes.

***/5 * * * * admusr /var/TKLC/db/filemgmt/dca/fetchLogDsa.sh**

Run the rotate logger once a day.

*** */24 * * * root /usr/sbin/logrotate /etc/logrotate.d/dsa_application_log_rotate**

Note: Oracle recommends not changing any fields in fetchLogDsa.ini and dsa_application.cron files since these are set optimally.

Appendix A. General Recommendations

While configuring the DSA, consider the following:

1. Increase the resource allocation to achieve desired throughput. Details for increasing the resource allocation is provided in section 3.2.
2. Ensure that after enabling a countermeasure, its related configuration tables are configured properly for countermeasure to take effect. In the case of no configuration or invalid configuration, countermeasure do not have any effect. Table 37 provides the configuration tables associated with countermeasures (Refer Section 2.4 and Section 2.5 for more details).

Table 37. Contermeasure Configuration

Countermeasure Name	Configuration Table
Origin Realm and Destination Realm Whitelist Screening Countermeasure	Realm_List
Application ID Whitelist Screening Countermeasure	AppIdWL_Config
Application ID and Command Code Consistency Check Countermeasure	AppCmdCst_Config
AVP Instance Check Countermeasure	AVPInstChk_Config
VPLMN ID and Origin Realm Consistency Check Countermeasure	VplmnORCst_Config
Specific AVP Screening Countermeasure	SpecAVPScr_Config
Time Distance Countermeasure	TimeDistChk_Config
Measure Rate Monitoring Countermeasure	MsgRateMon_Config

3. For validating the configurations, set the **Operating Mode** parameter in Security_Countermeasure_Config table as **Detection_Only**. Once configurations are validated, then the **Operating Mode** parameter can be changed as desired.
4. For stateful countermeasures, set the **Operating Mode** parameter in Security_Countermeasure_Config table as **Detection_Only** for at least the first 24 hours. This allows the security application to learn about any subscribers who are already roaming in partner networks without impacting their service. The operating mode can be changed to **Detection and Correction** after that period, if desired by the operator.
5. Set the value for the **Error Action if U-SBR Failure** parameter (in the System_Config_Options table) as **Continue Processing** to ensure the requests are not dropped and roaming subscribers continue to receive service in case of any U-SBR error (though it is a rare occurrence). Also change the Operating mode for any enabled stateful countermeasures (in the Security_Countermeasure_Config table) to **Detection_Only** for 24 hours (revert to original after 24 hours) if U-SBR errors are observed.
6. To share the common U-SBR database, between the DSA of different sites, the SOs need to be under the same NO.

Appendix B. My Oracle Support (MOS)

My Oracle Support

MOS (https://support.oracle.com) is your initial point of contact for all product support and training needs. A representative at Customer Access Support (CAS) can assist you with MOS registration.

Call the CAS main number at **1-800-223-1711** (toll-free in the US), or call the Oracle Support hotline for your local country from the list at http://www.oracle.com/us/support/contact/index.html. When calling, make the selections in the sequence shown on the Support telephone menu:

1. Select **2** for New Service Request.
2. Select **3** for Hardware, Networking and Solaris Operating System Support.
3. Select one of the following options:

   For technical issues such as creating a new Service Request (SR), select **1**.

   For non-technical issues such as registration or assistance with MOS, select **2**.

You are connected to a live agent who can assist you with MOS registration and opening a support ticket. MOS is available 24 hours a day, 7 days a week, 365 days a year.

Emergency Response

In the event of a critical service situation, emergency response is offered by the CAS main number at 1-800-223-1711 (toll-free in the US), or by calling the Oracle Support hotline for your local country from the list at http://www.oracle.com/us/support/contact/index.html. The emergency response provides immediate coverage, automatic escalation, and other features to ensure that the critical situation is resolved as rapidly as possible.

A critical situation is defined as a problem with the installed equipment that severely affects service, traffic, or maintenance capabilities, and requires immediate corrective action. Critical situations affect service and/or system operation resulting in one or several of these situations:

- A total system failure that results in loss of all transaction processing capability
- Significant reduction in system capacity or traffic handling capability
- Loss of the system's ability to perform automatic system reconfiguration
- Inability to restart a processor or the system
- Corruption of system databases that requires service affecting corrective actions
- Loss of access for maintenance or recovery operations
- Loss of the system ability to provide any required critical or major trouble notification

Any other problem severely affecting service, capacity/traffic, billing, and maintenance capabilities may be defined as critical by prior discussion and agreement with Oracle.

Locate Product Documentation on the Oracle Help Center

Oracle Communications customer documentation is available on the web at the Oracle Help Center (OHC) site, http://docs.oracle.com. You do not have to register to access these documents. Viewing these files requires Adobe Acrobat Reader, which can be downloaded at http://www.adobe.com.

1. Access the **Oracle Help Center** site at http://docs.oracle.com.
2. Click **Industries**.
3. Under the Oracle Communications subheading, click the **Oracle Communications** documentation link. The Communications Documentation page appears. Most products covered by these documentation sets display under the headings Network Session Delivery and Control Infrastructure or Platforms.
4. Click on your **Product** and then the Release Number. A list of the entire documentation set for the selected product and release displays. To download a file to your location, right-click the PDF link, select **Save target as** (or similar command based on your browser), and save to a local folder.