Oracle® Communications Diameter Signaling Router

Diameter Security Application User Guide with UDR

Release 9.0.0.0.0
F74160-02
October 2023

Oracle Communications Diameter Signaling Router Diameter Security Application User Guide with UDR, Release 9.0.0.0.0

F74160-02

Contents

My Oracle Support

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

Call the Customer Access Support 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 below 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 My Oracle Support, select **2**.

You are connected to a live agent who can assist you with My Oracle Support registration and opening a support ticket.

My Oracle Support is available 24 hours a day, 7 days a week, 365 days a year.

What's New in This Guide

This section introduces the documentation updates made in Oracle® Communications Diameter Signaling Router Diameter Security Application User Guide with UDR.

Release 9.0.0.0.0 - F74160-02, July 2023

- Updated description for Application_ID field in AppCmdCst_Config Table.
- Provided further information on SIVC CM under configuration for Application Route Table in DSA Mandatory Configuration.
- Updated the Application_ID condition in Application-ID Whitelist Screening (AppIdWL).

Release 9.0.0.0.0 - F74160-01, April 2023

- Added the following sections as part of DSA Low Level Filtering feature update:
  - AVP Whitelist Screening (AVPWLScr)
  - Origin Host and Origin Realm Format Check (OhOrFrmChk)
  - Session Id Validation Check (SesIdValChk)
  - AVPWLScr_Config Table
- Added the following three stateful countermeasures in the Understanding DSA Functionality and DSA Logic Process sections:
  - AVP Whitelist Screening
  - Origin Host and Origin Realm Format Check
  - Session Id Validation Check
- Added `AVPWLScr_Config Table` in the list of tables mentioned in Configuring DSA section.
- Added `AVP Whitelist Screening (AVPWLScr)` under the list of Stateless countermeasures in the ART Configuration for DSA section.
- Added `AVPWLScr_Config` table in the DSA Tables section.
- Added the following measurements in the ProcessedBy<Countermeasure ShortName> section:
  - ProcessedByAVPWLScr
  - ProcessedByOhOrFrmChk
  - ProcessedBySesIdValChk
- Added the following measurements in the DetectedBy<Countermeasure ShortName> section:
  - DetectedByAVPWLScr
  - DetectedByOhOrFrmChk
  - DetectedBySesIdValChk
- Added the following measurements in the DroppedBy<Countermeasure ShortName> section:

  - DroppedByAVPWLScr
  - DroppedByOhOrFrmChk
  - DroppedBySesIdValChk
- Added the following measurements in the RejectedBy<Countermeasure ShortName> section:
  - RejetedByAVPWLScr
  - RejetedByOhOrFrmChk
  - RejetedBySesIdValChk
- Added the following measurements in the FailedExec<Countermeasure ShortName> section:
  - FailedExecAVPWLScr
  - FailedExecOhOrFrmChk
  - FailedExecSesIdValChk
- Added the following measurements in the VulnerableBy<Countermeasure ShortName> section:
  - VulnerableByAVPWLScr
  - VulnerableByOhOrFrmChk
  - VulnerableBySesIdValChk
- Added the following measurements in the <Countermeasure ShortName>ExecFailed Alarm section:
  - AVPWLScrExecFailed<Alrm as a suffix>
  - OhOrFrmChkExecFailed<Alrm as a suffix>
  - SesIdValChkExecFailed<Alrm as a suffix>
- Added a new table TimeDistChk_Exception_List Table
- Updated the AVP Name description in AVPInstChk_Config Table
- Updated the field values in DSA Vulnerable Message Logging Details.

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

- Diameter Custom Applications Feature Activation Guide
- Diameter User's Guide
- DCA Programmer'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 UDR. 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

The following table provides information about the acronyms and the terminology used in the document:

Table 1-1 Acronyms

Acronym	Description
AIR/A	Authentication-Information Request/Answer
ART	Application Routing Table
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, http://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.

Table 1-1 (Cont.) Acronyms

Acronym	Description
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.
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
SIVC	Session Integrity Validation Check
SOAM	System Operations, Administration, and Maintenance

Table 1-1 (Cont.) Acronyms

Acronym	Description
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
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.
UDR	Unified Data Repository

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, UDR 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 UDR 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. For instructions about how to activate the DCA framework, refer to the *Diameter Custom Applications Feature Activation Guide*.

When DSA is initially installed, it is disabled, and you must manually enable it by navigating to **Diameter**, and then **Maintenance**, and then **Applications** and enable the application for every DMAP using DSA.

If DSA is in the DCA framework GUI menu, it indicates that the application is already enabled, but that does not guarantee if it is provisioned. You can also disable DSA from **Diameter**, and then **Maintenance**, and then **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 the following 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. UDR 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
- Session Integrity Validation Check

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 UDR for performing validation procedure. A list of stateless 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
- AVP Whitelist Screening
- Origin Host and Origin Realm Format Check
- Session Id Validation Check

Counter measures based on fs19 category are as follows:

- Category 0:
  - Origin Realm and Destination Realm Whitelist Screening
  - Destination-Realm and Origin-Realm Match Check
  - AVP Multiple Instance Check

  - AVP Whitelist Screening
  - Origin Host Origin Realm Format check
  - Session Id validation check
  - Destination Host Destination Realm Format Check
- Category 1:
  - Application-ID Whitelist Screening
  - Application-ID and Command-Code Consistency Check
- Category 2:
  - Subscriber Identity Validation
  - Visited-PLMN-ID and Origin-Realm Consistency Check
  - Specific AVP Screening
  - Origin host and Origin Realm Consistency Check
  - Realm and IMSI Consistency Check
- Category 3:
  - Message Rate Monitoring
  - Time-Distance Check
  - Previous Location Check
  - Source Host Validation HSS
  - Source Host Validation MME
  - Session Integrity validation 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.

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

- DSA starts executing the provisioned countermeasures, which are enabled, in a predefined sequence irrespective of the countermeasure's provisioning sequence.
- Each countermeasure can be enabled or disabled independently for screening the message for vulnerability.
- The stateless countermeasures are performed first followed by stateful countermeasures for better efficiency. The stateless countermeasures are processed in the following 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)
11. AVP Whitelist Screening (AVPWLScr)
12. Origin Host and Origin Realm Format Check (OhOrFrmChk)
13. Session Id Validation Check (SesIdValChk)

The stateful countermeasures are processed in the following 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)
6. Session Integrity Validation Check (SesIntValChk)

This countermeasure screens S6a/d ULR and AIR messages of Outbound Roaming Subscribers are 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 are 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 geographical coordinate (Latitude/Longitude) of the capital city of each country (MCC) used by this countermeasure for screening. This configuration is already pre-configured with geographical coordinate (Latitude/Longitude) of the capital city of all the countries. The option is also available to update/insert the geographical coordinate’s details for any missing country.

The option is also available to consider the S6a/d ULR and AIR messages as vulnerable if the geographical coordinates of the country for the received message 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 geographical coordinates for both the countries is configured, but the actual transit time is less than the calculated minimum transit time (calculated using geo-coordinates of the two countries).
- The geographical coordinates for either of the countries is not configured and the configuration says to mark the message as vulnerable, if matching configuration not found.
- This countermeasure also provide exception list of neighboring countries for each country to exempt S6a/d ULR and AIR messages from screening.

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.
- Transit time between two geo-coordinates point is calculated using distance (between two geo-coordinate points) and speed (user configured in the System_Config_Options table, default: 700 km/hr).

Apart from the mandatory configurations, configure the DSA tables for this countermeasure.

2.3.1 DSA Mandatory Configuration

To screen the incoming message for vulnerability, DSA uses various values provisioned in DSA tables 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 need 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 with 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 UDR 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.

- Application Route Table need to be provisioned with two rules for SIVC CM if it is enabled.
  These provisioned ART rules have the conditions to route all 3GPP S6a and 3GPP Gx CCR-I messages to DSA application. If SIVC CM is not enabled, then the Application Route Table must be provisioned with one rule which has conditions to route all 3GPP S6a messages to DSA application.

2.4 DSA Stateless Countermeasure Logic

Stateless countermeasures do not require maintenance of any State-Data (in UDR) 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 Configuration Set Name containing the peer from which the diameter message is received is not configured in the Foreign_WL_Peer_Cfg_Set of AppIdWL_Config Table.
  Apart from the mandatory configuration in DSA Mandatory Configuration, configure AppIdWL_Config Table for configuring allowed 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 DSA Mandatory Configuration, configure 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 Destination-Realm of the ingress diameter message is not configured as Home network’s Realm.
- 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 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 the following:
  - 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 the following 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 in DSA Mandatory Configuration, configure 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 the Origin-Realm and Destination-Realm of the ingress diameter request have the same value.

Apart from the mandatory configuration in 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.

The pre-conditions for executing this countermeasure are stated as follows. 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 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 the fields are of 3 digits. If the MNC of the PLMN is of 2 digits, then add a zero at 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 in DSA Mandatory Configuration, configure VPLMN_ID_Exception_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.

The pre-conditions for executing this countermeasure are as follows. 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.
- 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 the fields are of 3 digits. If the MNC of the PLMN is of 2 digits, then add a zero at 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 in DSA Mandatory Configuration, Realm_IMSI_Cst_Config table has 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 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 in DSA Mandatory Configuration, configure 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 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. For more information, refer to ART Configuration for DSA.

Apart from the mandatory configuration in DSA Mandatory Configuration, configure 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. For more information, refer to ART Configuration for DSA.

Apart from the mandatory configuration in DSA Mandatory Configuration, configure 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.

2.4.11 AVP Whitelist Screening (AVPWLScr)

This countermeasure screens the ingress diameter request/answer message for whitelist AVP(s) screening.

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 any of these conditions are true:

- Any AVP present in diameter message is not needed by technical specifications (AVP whitelist screening).
- **Nesting level of grouped AVPs:** Control of maximum nesting level of grouped AVPs over interconnection interfaces (maximum Nesting Depth should be 8).
- **Encoding risks of AVPs:** If an AVP has been defined as UTF8 String, OctetString, and DiameterIdentity and/or if an address format purposely contains manipulated contents with the objective to introduce unintended behavior.

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. For more information, refer to ART Configuration for DSA.

Apart from the mandatory configuration in DSA Mandatory Configuration, configure the AVPWLScr_Config Table for configuring values for AVP(s) used by this countermeasure for screening. The AVPWLScr_Config Table contains list of AVPs with AVP_Name, AVP_Code, AVP_DataType, Vendor_Id, Command_Code_List, Message_Type, and Diameter_Version.

2.4.12 Origin Host and Origin Realm Format Check (OhOrFrmChk)

This countermeasure screens the ingress diameter request/answer message for occurrence of Origin Host and Origin Realm AVPs in incoming Request and answer message.

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

- If count of AVPs in message is greater than one.
- If format of both the AVPs is not correct.

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. For more information, refer to ART Configuration for DSA.

2.4.13 Session Id Validation Check (SesIdValChk)

This countermeasure screens the ingress diameter request/answer message for Session Id AVP as first AVP in diameter message.

This countermeasure considers the ingress diameter request/answer message as vulnerable, if the Session Id AVP is not the first AVP in diameter message.

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. For more information, refer to ART Configuration for DSA.

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. UDR 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:

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 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 in DSA Mandatory Configuration, configure the tables outlined in this section.

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 geographical coordinate (Latitude/Longitude) of the capital city of each country (MCC) used by this countermeasure for screening. This configuration is already pre-configured with geographical coordinate (Latitude/Longitude) of the capital city of all the countries. The option is also available to update/insert the geographical coordinate's details for any missing country.

The option is also available to consider the S6a/d ULR and AIR messages as vulnerable if the geographical coordinates of the country for the received message 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 geographical coordinates for both the countries is configured, but the actual transit time is less than the calculated minimum transit time (calculated using geo-coordinates of the two countries).
- The geographical coordinates for either of the countries is not configured and the configuration says to mark the message as vulnerable, if matching configuration not found.
- This countermeasure also provide exception list of neighboring countries for each country to exempt S6a/d ULR and AIR messages from screening.

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.
- Transit time between two geo-coordinates point is calculated using distance (between two geo-coordinate points) and speed (user configured in the System_Config_Options table, default: 700 km/hr).

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

- TimeDistChk_Config Table: For configuring geographical coordinates (Latitude/ Longitude) for each country used by this countermeasure for screening.
- System_Config_Options Table: Option to indicate the average flight speed to consider for calculating the transit time. Option to define the behavior when no matching Source and Destination location is configured. Option to enable the behavior for neighboring country exemption list screening.
- TimeDistChk_Exception_List: For configuring List of neighboring countries MCC for which Time Distance Check screening will not be applied.
- TimeDistChk_MCC_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.

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.

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 CLR can be processed by this countermeasure. For more information, refer to ART Configuration for DSA.

System_Config_Options Table: Check the **Process_Foreign_RSR_Msg** field, if RSR message needs to be processed 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.

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. For more information refer to the ART Configuration for DSA.

2.5.6 Session Integrity Validation Check (SesIntValChk)

Session Integrity Validation Check [SesIntValChk] facilitates in GTP-C signaling fraud detection based on subscriber location information for an outbound roaming subscriber. This countermeasure screens 3GPP-Gx-CCR-I message of Outbound Roaming Subscribers to check that if a ULR message corresponding to this CCR-I message is already present in UDR DB or not.

This CM consider that ULR message are validated with other countermeasure run and store key/value pair into UDR DB. SIVC validate the CCR-I message against ULR message received, validated and found key/value into UDR DB.

To enable Session Integrity Validation Check Countermeasure, User has to enable below two stateful countermeasures first for ULR message validation followed by SIVC CM to validate CCR_I message:

- Time Distance Check Countermeasure
- Source Host Validation MME Countermeasure

This countermeasure considers the ingress GX CCR-I message as vulnerable if any of these conditions are true:

- The IMSI value of Gx-CCR-I message is not found in UDR DB. That means we have not received any ULR message corresponding to this CCR-I message.
- The MCCMNC value from the Gx-CCR-I message is not matching with the MCCMNC value of ULR message stored in UDR DB with same IMSI.

Appropriate ART configuration needs to be done for routing the Gx-CCR-I messages generated from outbound roamers towards DSA Application so that it can be processed by this countermeasure. For more information, refer to the ART Configuration for DSA.

3 Configuring Comagent with UDR as Remote Server

Comagent Configuration with UDR DB will be NOAM Level Configuration.

3.1 ComAgent Configuration on DSR

For Comagent configuration, go to the Communication agent tab on Active DSR NO GUI and configure UDB DB Server IMI IP as remote server.

Note:

- If DSR and UDR deployment are in the same network, use UDR IMI IP as Comagent Remote Server Configuration.
- If DSR and UDR deployment are in a different network, use UDR XSI IP as Comagent Remote Server configuration.

For this, add new XSI Interface on both DSR and UDR side for Comagent Communication. Ensure that newly added XSI interface are Desktop routable and are accessible from both the sides.

Note:

Do not use DSR signaling Interface (XSI Interface) for comagent communication.

- **Remote Server Configuration**: Configure UDR DB as Remote Server.
- **Connection Group configuration**: Add previously configured Remote Server to the **STPSvcGroup** Connection Group.

  Note:

  This "STPSvcGroup" routed service is common for DCA and vSTP application.

  Note:

  Restart MP servers to make the Comagent service/connection up.

- **Steps to Restart the MPs Server**: Go to the Active DSR NOAM status & Manage section, select the MP server and restart the MP server by clicking the **Restart** button.

3.2 Comagent Configuration on UDR

For Comagent configuration, go to the Communication agent tab on Active UDR NO GUI and configure all the DSR MP IMI IP as client.

> **Note:**
>
> Refer to the ComAgent Configuration on DSR section for configuring the Interface IP as client.

- **Remote Server Configuration**: Configure DSR MPs IMI IP as Client.

  > **Note:**
  >
  > Reboot the Active UDR NOAM Server to make the Comagent service/ connection up.

- **Steps to Reboot the MPs Server**: Go to the Active UDR NOAM status & Manage section, select the Active NOAM server and reboot the Active NOAM server by clicking the **Reboot** button.

3.3 Comagent Connection Status Validation

- **Comagent Connection status check on DSR NO Server**: For Connection status check, go to the Communication agent Maintenance tab on DSR NO GUI.
- **Routed Service status check on DSR NO Server**: For routed service status check, go to the Communication agent Maintenance tab on DSR NO GUI.
- **HA Service status check on DSR NO Server**: For HA Service status check, go to the Communication agent Maintenance tab on DSR NO GUI.
- **Comagent Connection status check on UDR NO Server**: For Connection status check, go to the Communication agent Maintenance tab on UDR NO GUI.
- **Routed Service status check on UDR NO Server**: For routed service status check, go to the Communication agent Maintenance tab on UDR NO GUI.
- **HA Service status check on UDR NO Server**: For HA Service status check, go to the Communication agent Maintenance tab on UDR NO GUI.

4 UDR Configuration

This section provides information about enabling the security profile and configuring the Audit Time.

4.1 Enabling Security Profile on Active UDR NOAM for DSA Application

1. Log in to Active NOAM Server through putty session and run the `enableSecurityApp` loader.
2. Go to this path: `/usr/TKLC/udr/prod/maint/loaders/upgrade`.
3. Run the `enableSecurityApp` script.
4. Reboot both the UDR NOAM servers.

4.2 Auditing Time Configuration on Active UDR NOAM

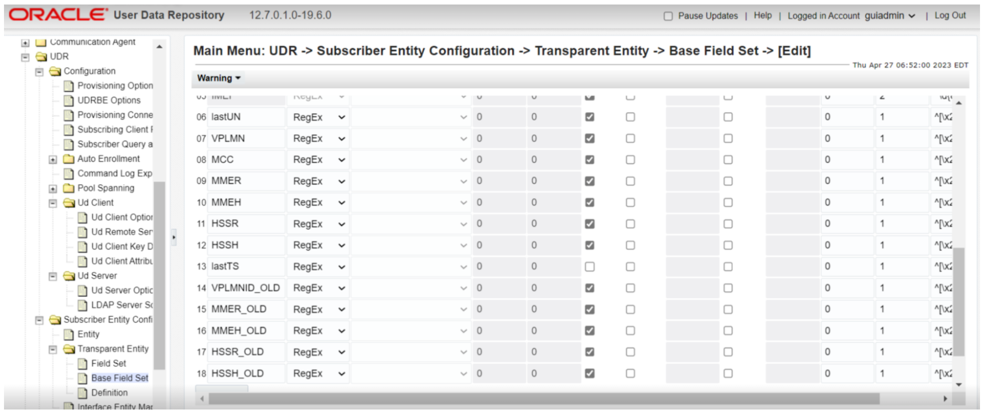
By default, the Audit Time configuration is disabled (unchecked), for example, no record is cleaned up on the UDR server.

- To clean up the old records on UDR, do the following:
  1. Select the `Cleanup Inactive Security App Subscriber Enabled` option.
  2. Set the value of `Security App SDO Audit Interval` to 10.

     The system removes all the records after 10 seconds.

4.3 Configure DSA Fields in UDR

1. From the main Menu, navigate to **UDR**, then select **Subscriber Entity Configuration**, then select **Transparent Entity**, select **Base Field Set**.
2. Click **SprProfileBFS**.



User Data Repository 12.7.0.1.0-19.6.0

Pause Updates | Help | Logged in Account guiadmin | Log Out

Main Menu: UDR -> Subscriber Entity Configuration -> Transparent Entity -> Base Field Set -> [Edit]

Thu Apr 27 06:52:00 2023 EDT

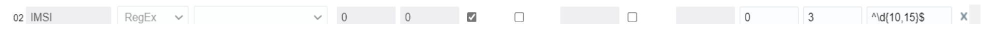
06	lastUN	RegEx		0	0	✓	☐		☐		0	1	^[\x2
07	VPLMN	RegEx		0	0	✓	☐		☐		0	1	^[\x2
08	MCC	RegEx		0	0	✓	☐		☐		0	1	^[\x2
09	MMER	RegEx		0	0	✓	☐		☐		0	1	^[\x2
10	MMEH	RegEx		0	0	✓	☐		☐		0	1	^[\x2
11	HSSR	RegEx		0	0	✓	☐		☐		0	1	^[\x2
12	HSSH	RegEx		0	0	✓	☐		☐		0	1	^[\x2
13	lastTS	RegEx		0	0	☐	☐		☐		0	1	^[\x2
14	VPLMNID_OLD	RegEx		0	0	✓	☐		☐		0	1	^[\x2
15	MMER_OLD	RegEx		0	0	✓	☐		☐		0	1	^[\x2
16	MMEH_OLD	RegEx		0	0	✓	☐		☐		0	1	^[\x2
17	HSSR_OLD	RegEx		0	0	✓	☐		☐		0	1	^[\x2
18	HSSH_OLD	RegEx		0	0	✓	☐		☐		0	1	^[\x2

Figure 4-1 SprProfileBFS screen

3. Click **Edit**.
4. Add the following fields.

```
IMSI Regular expression: ^\d{10,15}$
lastUN Regular expression: ^[\x20-\x7e]{0,255}$
VPLMN Regular expression: ^[\x20-\x7e]{0,255}$
MCC Regular expression: ^[\x20-\x7e]{0,255}$
MMEH Regular expression: ^[\x20-\x7e]{0,255}$
MMER Regular expression: ^[\x20-\x7e]{0,255}$
HSSR Regular expression: ^[\x20-\x7e]{0,255}$
HSSH Regular expression: ^[\x20-\x7e]{0,255}$
lastTS Regular expression: ^[\x20-\x7e]{0,255}$
VPLMNID_OLD Regular expression: ^[\x20-\x7e]{0,255}$
MMER_OLD Regular expression: ^[\x20-\x7e]{0,255}$
MMEH_OLD Regular expression: ^[\x20-\x7e]{0,255}$
HSSR_OLD Regular expression: ^[\x20-\x7e]{0,255}$
HSSH_OLD Regular expression: ^[\x20-\x7e]{0,255}$
```

Figure 4-2 Configuration Fields



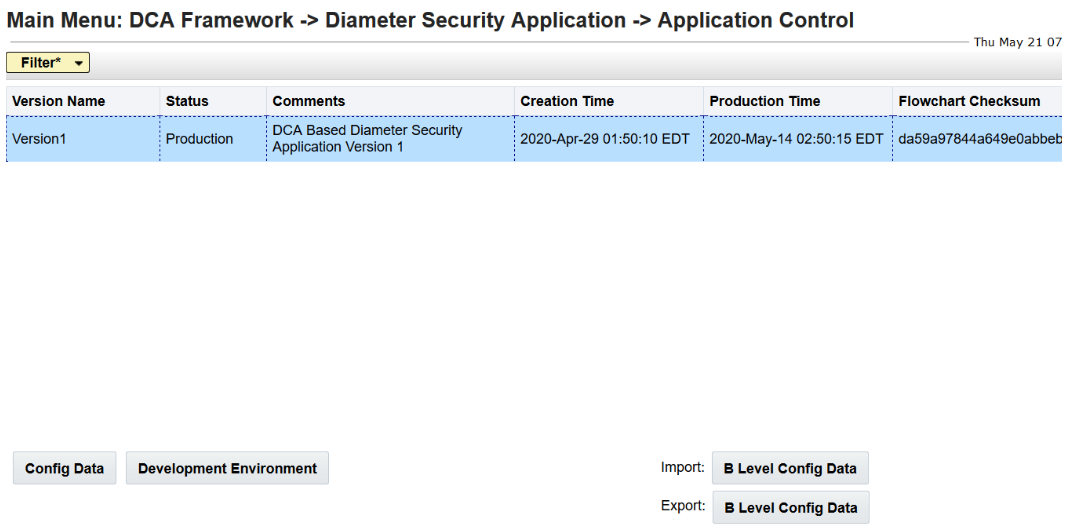
02	IMSI	RegEx		0	0	✓	☐		☐		0	3	^\d{10,15}$	X

5 Upgrade

The upgrade from DSA with U-SBR to DSA with UDR DSR Release is not supported, The upgrade from DSA with UDR to DSA with UDR DSR Release is supported.
Perform the following procedure to migrate the configuration data.

1. Export the SOAM configuration data on the previous release setup.
   a. Log in to the SOAM GUI.
   b. Export the B Level configuration data.
   c. Export the SOAM configuration data.
2. Click the B level configuration data and save it on the local system.

Figure 5-1 DSA SOAM Level Configuration Export



Main Menu: DCA Framework -> Diameter Security Application -> Application Control

Thu May 21 07

Version Name	Status	Comments	Creation Time	Production Time	Flowchart Checksum
Version1	Production	DCA Based Diameter Security Application Version 1	2020-Apr-29 01:50:10 EDT	2020-May-14 02:50:15 EDT	da59a97844a649e0abbeb

3. To import the configuration data to a new release setup, the system should fresh install on a new release.
4. To import the SOAM configuration data, click **B Level Config Data**, select the "file", and then click **Import**.

6
Configuring DSA

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

For holding configuration values, DSA uses the following tables as described in DSA Tables:

- Security_Countermeasure_Config Table
- Foreign_WL_Peers_Cfg_Sets Table
- System_Config_Options Table
- MCC_MNC_List 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
- AVPWLScr_Config Table

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

6.1 DSA Pre-Activation Activities

Before activating DSA as a DCA application, DCA framework must be activated on the NO. For more information, refer to the *Diameter Custom Applications Feature Activation Guide*.

Note:

After DSA is activated, by default the application is in the disabled state. While disabled, no diameter traffic is delivered to DSA. For the procedure to enable an application, refer to the *Diameter User's Guide*.
DSA's operational status is unavailable until a successful compiled version (production or trial version) of DSA is configured.

6.2 Activating DSA

More information about this procedure is available in the *Diameter Custom Applications Feature Activation Guide*.

1. Ensure that the DCA framework is activated by referring to the *Diameter Custom Applications Feature Activation Guide*.
2. Activate DSA using the DCA Application Activate procedure as described in the *Diameter Custom Applications Feature Activation Guide*.
   - Recommended DCA Short Name: DSA
   - Recommended DCA Long Name: Diameter Security Application
3. Post DSA activation, check the visibility of DSA subtree in the main menu **DCA Framework**, and then **Diameter Security Application**.
4. To verify whether DSA is activated before enabling DSA and performing provisioning activities, do the following:
   a. Confirm the DSA folder is visible on the GUI under the main menu: **DCA Framework**.
   b. Ensure that all measurements and KPIs associated with the DCA framework are visible on **Measurements**, and then **Report and Status & Manage**, and then **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.
5. To set the DSA resource allocation, do the following:
   a. Log in to the active SO server using SSH as `admusr`.
   b. Run the `update_dca_thread_count_damp_profile.sh` script.
   c. Select **1** to increase thread counts.
   d. Restart the DAMPs hosting DSA under this SO.

6.3 Configuring DSA Business Logic and Database Schema

Perform this procedure to import DSA business logic and the configuration database schema using the `DSA JSON` file.

DSA NO JSON file name: `Diameter_Security_Application-Version1.json`.

1. From the NO GUI main menu, navigate to **DCA Framework**, and then **Diameter Security Application**, and then **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. Select the **Abort on first error** check box to abort the import process in case of error.
6. Click **Import** to start the import process.
7. To verify whether DSA JSON is imported before enabling DSA and performing provisioning activities, do the following:
   a. From the NO GUI main menu, navigate to **DCA Framework**, and then **Diameter Security Application**, and then **Application Control**, and ensure that an entry is added in DCA application version details table.
   b. Select the newly added version and click **Config Tables and Data**.
   c. Ensure that all DSA configuration tables are listed.
   d. Select the newly added version and click **Development Environment**.
   e. Ensure that DSA Perl business logic is present.

6.4 Configuring DSA Mandatory Options

1. To increase the maximum supported State-Data size, do the following:
   a. From the NO GUI main menu, navigate to **DCA Framework**, and then **Configuration**.
   b. Set the **Maximum Size of Application State** to **4800**.
   c. Click **Apply**.
2. To configure general options, do the following:
   a. From the NO GUI main menu, navigate to **DCA Framework**, and then **Diameter Security Application**, and then **General Options**.
   b. Update **Perl Subroutine for Diameter Request** to **process_request**.
   c. Update **Perl Subroutine for Diameter Answer** to **process_answer**.
   d. Update **Max. UDR Queries per Message** to **10**.
   e. Clear the **Enable Opcodes Accounting** option to disable opcode accounting.
   f. Click **Apply**.

6.5 ART Configuration for DSA

DSA processes ingress diameter messages received from other 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:

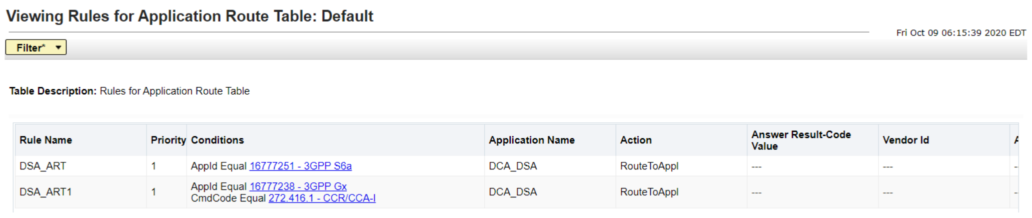
- Stateless countermeasures:
  - Origin Realm and Destination Realm Whitelist Screening (RealmWLScr)
  - Specific AVP Screening (SpecAVPScr)
  - AVP Multiple Instance Check (AVPInstChk)
  - AVP Whitelist Screening (AVPWLScr)
- Stateful countermeasures:
  - Previous Location Check (PreLocChk)
  - Source Host Validation HSS (SrcHostValHss)
  - Source Host Validation MME (SrcHostValMme)

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.

If you want to screen the diameter message with SIVC CM, then you need to create below two ART rules to route all 3GPP S6a and 3GPP Gx CCR-I messages to DSA application:

- Create an ART rule with condition of AppId equal to 3GPP S6a and routes to DSA application.
- Create an ART rule with conditions of AppId equal to 3GPP Gx and CmdCode equals to CCR/CCA-I and routes to DSA application.

Figure 6-1   ART Rules Configured for SIVC CM



Viewing Rules for Application Route Table: Default

Fri Oct 09 06:15:39 2020 EDT

Filter*

Table Description: Rules for Application Route Table

Rule Name	Priority	Conditions	Application Name	Action	Answer Result-Code Value	Vendor Id
DSA_ART	1	AppId Equal 16777251 - 3GPP S6a	DCA_DSA	RouteToAppl	---	---
DSA_ART1	1	AppId Equal 16777238 - 3GPP Gx CmdCode Equal 272.416.1 - CCR/CCA-I	DCA_DSA	RouteToAppl	---	---

6.6 Enabling and Disabling DSA

Perform the following procedure to enable and disable DSA on the SO.

1. To enable DSA, do the following:
   a. Navigate to **Diameter**, and then **Maintenance**, and then **Applications**.

   b. Select DCA_DSA entries and click **Enable**.
2. To disable DSA, do the following:
   a. Navigate to **Diameter**, and then **Maintenance**, and then **Applications**.
   b. Select DCA_DSA entries and click **Disable**.

6.7 Deactivating DSA

Perform this procedure to deactivate DSA. You cannot deactivate DSA while a version of the respective application is still in the production or trial state.

Disable DSA on all MPs in the network and no ART rules should refer to DSA.

For more information, refer to the *Diameter Custom Applications Feature Activation Guide*.

1. To disable DSA for all the MPs from the SO GUI main menu, navigate to **Diameter**, and then **Maintenance**, and then **Applications**.
2. Delete ART rules referring to DSA.
3. Deactivate DSA using DCA Application Activate procedure as described in the *Diameter Custom Applications Feature Activation Guide*.

7

DSA Tables

DSA database schema defines various tables used to define and customize the application behavior.

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 UDR Generic State database indexed by the subscriber's IMSI.

Table 7-1 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 7-2
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 7-5
System_Config_Options	This table contains common configurable options required to process various countermeasure business logic.	Yes	SO	Table 7-7
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 7-11
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 7-13

Table 7-1 (Cont.) DSA Configuration Tables

Table Name	Description	Single Row Indicator	Table Level	Table Fields
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 7-15
VplmnORCst_Config	This table defines the list of Application-ID and its Supported Command-Code combinations.	No	SO	Table 7-17
SpecAVPScr_Config	This table defines the list of AVP's that needs to be screened in the incoming messages.	No	SO	Table 7-19
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 7-21
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	#unique_83/ unique_83_Connect_42_TABLE_P33_HTW_V4B
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 7-25
AppCmdCst_Config	This table defines the Application-ID and supported Command-Codes for a given Roamer Type.	No	SO	Table 7-27
AVPWLScr_Config	This table defines the AVP Whitelist screening config countermeasure.	No	SO	Table 7-44Table 7-44

7.1 Configuring DSA Tables

DSA configuration tables are pre-populated if DSA is configured using DSA JSON file. For more information, refer to Configuring DSA Business Logic and Database Schema.

Alternatively, DSA configuration tables can be configured manually using the following procedure. For more information, refer to the *DCA Programmer's Guide*.

1. From the NO GUI main menu, navigate to **DCA Framework**, and then **Diameter Security Application**, and then **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 in the fields to define the table.
6. Click **Add** to add multiple Table Fields.
7. Click **OK** or **Apply**.
8. Repeat Step 4 to 7 for each table listed in Table 7-1.

7.2 Provisioning DSA Tables

Perform this procedure to import DSA default provisioning data using the DSA JSON file.

DSA SO JSON file name: `Diameter_Security_Application-Version1_Default_Config.json`. For more information, refer to the *DCA Programmer's Guide*.

1. From the SO GUI main menu, navigate to **DCA Framework**, and then **Diameter Security Application**, and then **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. Select the **Abort on first error** check box to abort the import process in case of error.
6. Click **Import** to start the import process.
7. To complete the additional provisioning manually, do the following:
   a. From the SO GUI main menu, navigate to **DCA Framework**, and then **Diameter Security Application**, and then **Application Control**.
   b. Select the newly added **DSA Version Name**.
   c. Click **Config Data**.

      If DSA JSON is not used to import DSA business logic and the configuration database schema, then the configured table list is empty.
   d. Select the table that needs to be provisioned.
   e. Click **Provision Table**.
   f. Click **Insert**.
   g. Fill in the values for required fields of the table.
   h. Click **OK** or **Apply**.

7.3 DSA Table Details

7.3.1 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 7-2 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 rejected 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 discarded by DSR by sending an Error Answer and is not processed/relayed any further.
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.

Note:

Upon enabling a new countermeasure, ensure that 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 7-3 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 Session_Integrity_Validation_Check_SesIntValChk: 16 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.

Table 7-3 (Cont.) Field Details for Security_Countermeasure_Config

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
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.

Table 7-3 (Cont.) Field Details for Security_Countermeasure_Config

Field Name	Unique	Mandatory	Data type, Range, and Default Value	Description
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.

7.3.1.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Apart from that, 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. For script compilation, refer to the *DCA Programmer's Guide*.

- 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.

7.3.2 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 following 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 7-4 Foreign_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**.

The following table describes the field details for the Foreign_WL_Peers_Cfg_Sets Table.

Table 7-5 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.

7.3.2.1 Additional Provisioning Rules

Basic input data validation is done using the DCA Framework's Configuration Data Provisioning GUI. Apart from that, 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. For script compilation, refer to the *DCA Programmer's Guide*.

- 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**, and then **Configuration**, and then **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>

7.3.3 System_Config_Options Table

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

Table 7-6 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 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.

Table 7-6 (Cont.) System_Config_Options Fields

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 UDR Failure	Defines the action performed if a UDR failure occurs while executing the business logic of a Stateful countermeasure. If Error_Action_for_UDR_Failure is configured as **Continue_Processing**, thenthe 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_UDR_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**, thenthe 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**, thenthe 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.

Table 7-6 (Cont.) System_Config_Options Fields

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.
TDC_Chk_For_First_ULR_AIR_Msg	If checked, the DSA Application will screen first ULR/AIR message for vulnerability by Time Distance Check Countermeasure.
Error_Action_For_CASM_Failure	Defines the action performed if a CreateAndSendMsg request failure occurs while executing the business logic of a Stateful countermeasure. If Error_Action_for_CASM_Failure is configured as **Continue_Processing**, thenthe 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_UDR_Failure is configured as **Drop**, then the message is discarded at DSR and is not processed/relayed any further.
Avg_Flight_Velocity	Defines the Average Flight speed considered to calculate the Distance between two points using latitude and longitude for Time Distance Check CM. [Velocity in km].
TDC_Chk_For_Neighbour_Country	To decide whether Time Distance Check CM should be exempted for neighboring countries.
Max_Tuple_For_SrcHostValHss	(Bug#30133341) Defines the Max tuple to be stored in the UDR Db for Source Host Validation HSS CM for each subscriber. Either of 'Maximum Size of Application State' or Max_Tuple_For_SrcHostValHss ' is reached the limit, Oldest Tuple in UDR State Data will be popped off to store the latest tuple.
CounterMeasure_Exception_Chk	To decide whether to Enable or Disable the Security Exception function for the CounterMeasure.
MCCMNC_AVP	To decide from which AVP, MCCMNC value is to be fetched for Session Integrity validation Check [SesIntValChk] Countermeasure.

This table describes the field details for the System_Config_Options Table.

Note:

While the failure of a UDR is rare, loss of connectivity to a remote UDR can sometimes occur due to network fluctuations. Loss of connectivity is also treated by the DSA as a UDR failure and it is therefore desirable to set the value for the **Error Action if UDR 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 UDR 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 7-7 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_TimeNotConfigured	N/A	N/A	Boolean Range: Yes/No Default: No	To decide whether mark the message as vulnerable by 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
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.

Table 7-7 (Cont.) Field Details for System_Config_Options

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Error_Action_for_UDR_Failure	Yes	Yes	Enumerated Range: Continue_Processing: 1 Drop: 2 Default: Continue_Processing	Error action performed if UDR 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
TDC_Chk_For_First_ULR_AIR_Msg	N/A	N/A	Boolean Range: Yes/No Default: No	To decide whether to screen first ULR/AIR for vulnerability by Time Distance Check CM. Yes: Check first ULR/AIR for Vulnerability
Error_Action_For_CASM_Failure	Yes	Yes	Enumerated Range: Continue_Processing: 1 Drop: 2 Default: Continue_Processing	[CASM:- CreateAndSendMsg] Perform Error Action when CreateAndSendMsg gets failed. Continue_Processing: The message will be treated as nonVulnerable and will be processed further. Drop: The message will be treated as Vulnerable and will be dropped.

Table 7-7 (Cont.) Field Details for System_Config_Options

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
TDC_Chk_For_Continent	N/A	N/A	Boolean Range: Yes/No Default:Yes	To decide whether to screen ULR/AIR message for Continent check by Time Distance Check CM. Yes: Apply Continent check on AIR/ULR message for Vulnerability No: Don't Apply Continent check on AIR/ULR message for Vulnerability
MCCMNC_AVP	N/A	N/A	Enumerated: 3GPP_SGSN_MCC_MNC:1, 3GPP_User_Location_Info:2	To decide from which AVP, MCCMNC value is to be fetched for Session Integrity validation Check [SesIntValChk] Countermeasure.

7.3.3.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- 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

7.3.4 TimeDistChk_Exception_List Table

This table is used to configuring List of neighboring countries MCC for which time distance check screeing will not be applied.

Table 7-8 TimeDistChk_Exception_List Table

Field	Description
MCC	MCC in digits Range - 3 digits Integer

Table 7-8 (Cont.) TimeDistChk_Exception_List Table

Field	Description
Exception_List_For_Country	List of neighboring countries MCC(in Valid format) separated by semicolon (;) for which Time Distance Check will not be checked.

Table 7-9 Field Details for TimeDistChk_Exception_List

Field name	Unique	Mandatory	Data Type, Range, and Default Value	Description
MCC	Yes	Yes	Integer Range: 100-999 Default: NA	MCC in digits Range: 3 digits Integer
MNC	No	Yes	UTF8string Range: 1- 2048 Default: NA	List of neighboring countries MCC (in Valid format) separated by (;) for which Time Distance check will not be checked

Additional Provisioning Rules

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

- If time distance check counter measure is enabled and TDC_Chk_For_Neighbour_Country configured in System_Config_Options table, then this table cannot be empty. Atleast one record should be provisioned here.
- MCC is unique and no duplicate MCC are allowed.

7.3.5 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 7-10 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**)

This table describes the field details for the MCC_MNC_List Table.

Table 7-11 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.

Table 7-11 (Cont.) Field Details for MCC_MNC_List

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
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.

7.3.5.1 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. For script compilation, refer to the *DCA Programmer’s Guide*.

- 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.

7.3.6 AppIdWL_Config Table

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

Table 7-12 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.

This table describes the field details for the AppIdWL_Config Table.

Table 7-13 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.

7.3.6.1 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. For script compilation, refer to the *DCA Programmer’s Guide*.

- 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 setnameneeds 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**.

7.3.7 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 7-14 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.

This table describes the field details for the Realm_List Table.

Table 7-15 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.

7.3.7.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- 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 setnameneeds 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**.

7.3.8 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 7-16 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.

This table describes the field details for the VplmnORCst_Config Table.

Table 7-17 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.

Table 7-17 (Cont.) Field Details for VplmnORCst_Config

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).

7.3.8.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- 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.

7.3.9 SpecAVPScr_Config Table

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

Table 7-18 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.

Table 7-18 (Cont.) SpecAVPScr_Config Fields

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**, and then **AVP Dictionary**, and then **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. 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** , and then **Mediation**, and then **Enumerations**).

This table describes the field details for the SpecAVPScr_Config Table.

Table 7-19 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.

Table 7-19 (Cont.) Field Details for SpecAVPScr_Config

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

7.3.9.1 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. For script compilation, refer to the *DCA Programmer’s Guide*.

- 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.

7.3.10 AVPInstChk_Config Table

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

Table 7-20 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.
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**, and then **AVP Dictionary**, and then **All-AVP Dictionary**). Grouped AVP name can be defined with its Parent AVP Names (Maximum of eight (8) 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.

Table 7-20 (Cont.) AVPInstChk_Config Fields

Field	Description
Maximum Number of Instance	Maximum_Instance defines the maximum number of instances of the AVP in the incoming diameter message.

This table describes the field details for the AVP Multiple Instance Check (AVPInstChk).

Table 7-21 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 8 level) separated by ";". For example, BaseAVPName;SubAVPName;AVPName. Each AVP name cannot exceed 255 characters.
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.

7.3.10.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- 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**.

7.3.11 TimeDistChk_Continent_Config Table

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

Table 7-22 TimeDistChk_Continent_Config

Field	Description
Source and Destination Node-IDs	Defines the two Continent values. Value of TDC_Chk_For_Continent flag in System_Config_Options Table determines the configured Continent values. If TDC_Chk_For_Continent flag in System_Config_Options Table is set to YES, then the Continent_1 and Continent_2 from TimeDistChk_Continent_Config table are used to filter the vulnerable messages at first level.
Minimum Transition Time	Defines the minimum transition time (in minutes) required to move between Continent_1 and Continent_2.

This table describes the field details for the TimeDistChk_Continent_Config Table.

Table 7-23 Field Details for TimeDistChk_Continent_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Continent_1	No	Yes	Enum Range: EUROPE:2,NORTH_AMERICA_AND_THE_CARIBBEAN:3,ASIA_AND_MIDDLE_EAST:4,OCEANIA:5,AFRICA:6,SOUTH_AND_CENTRAL_AMERICA:7Default: N/A	List of various supported Continents.
Continent_2	No	Yes	Enum Range: EUROPE:2,NORTH_AMERICA_AND_THE_CARIBBEAN:3,ASIA_AND_MIDDLE_EAST:4,OCEANIA:5,AFRICA:6,SOUTH_AND_CENTRAL_AMERICA:7Default: N/A	List of various supported Continents.
Minimum_Transition_Time	No	Yes	Integer Range: 1–720 Default: N/A	Minimum Transition time [in Minutes] between the Continent_1 and Continent_2. [Range = 0 – 720]

7.3.11.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- If Time-Distance Check (TimeDistChk) countermeasure is provisioned in Security_Countermeasure_Config Table and TDC_Chk_For_Continent flag is set **YES** in System_Config_Options Table then this table will be used. All the Entries are pre-configured.
- Each **Continent_1** and **Continent_2** combination must be unique. No duplicate **Continent _1** and **Continent _2** combination is allowed. Not even by swapping **Continent _1** and **Continent _2** values.
  For example, an entry with Continent_1= EUROPE and Continent_2= AFRICA and another entry with Continent_1=AFRICA and Continent_2=EUROPE is not allowed.
- Valid Continents value range for **Continent _1** and **Continent _2** are EUROPE, NORTH_AMERICA_AND_THE_CARIBBEAN, ASIA_AND_MIDDLE_EAST, OCEANIA, AFRICA, SOUTH_AND_CENTRAL_AMERICA. This validation is performed TDC_Chk_For_Continent flag is set **YES** in System_Config_Options Table.

7.3.12 MsgRateMon_Config Table

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

Table 7-24 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.

This table describes the field details for the MsgRateMon_Config table.

Table 7-25 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.

7.3.12.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- 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.

7.3.13 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 7-26 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.

This table describes the field details for the AppCmdCst_Config Table.

Table 7-27 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. Since DSA for this CM supports only S6a, the expected value for application-id is 16777251.
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). Configure the valid command code values, for 16777251 S6a application id.

7.3.13.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- 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.

7.3.14 CreateAndSendMsg_Config Table

This table is used to customize the behavior of stateful countermeasures by using the following options.

Any Stateful CounterMeasure which is using CreateAndSendMsg Feature, They have to specify the Origin-Host/Realm and Destination-Host/Realm.

Table 7-28 CreateAndSendMsg_Config Fields

Field	Description
CounterMeasure_ Type	List of Countermeasures which are going to use CreateAndSendMsg Feature.
Origin Host/Realm and Destination Host/ Realm	Defines the Origin Host/Ream and Destination Host/Realm to be used for DSA App generated messages.

This table describes the field details for the CreateAndSendMsg_Config Table.

Table 7-29 Field Details for CreateAndSendMsg_Config Table

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
CounterMeasure_Type	Yes	Yes	Enumerated, Time_Distance_Check_TimeDistChk:12 Default=N/A Note: Currently, only Time Distance Check CM is supporting CreateAndSendMsg Feature.	List of CounterMeasures, which are going to use CreateAndSendMsg Feature.

Table 7-29 (Cont.) Field Details for CreateAndSendMsg_Config Table

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Origin_Host	No	Yes	DiameterIdentity Default: n/a,	Origin-Host value to be used during Creating message from DSA Business Logic.
Origin_Realm	No	Yes	DiameterIdentity Default: n/a,	Origin-Realm value to be used during Creating message from DSA Business Logic.
Destination_Host	No	Yes	DiameterIdentity Default: n/a,	Destination-Host value to be used during Creating message from DSA Business Logic
Destination_Realm	No	Yes	DiameterIdentity Default: n/a,	Destination-Realm value to be used during Creating message from DSA Business Logic.

7.3.14.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- If Time-Distance Check (TimeDistChk) countermeasure is provisioned in Security_Countermeasure_Config Table and TDC_Chk_For_First_ULR_AIR_Msg flag is set to YES in System_Config_Options Table then this table cannot be empty. At least one entry needs be provisioned.
- Each record should be unique. No duplicate countermeasure configuration is allowed.
- 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.

7.3.15 Exception_Rule_Config Table

This table is used to configure priorities for exception types for various countermeasures. It allows to customize the countermeasure behavior using the following options.

Table 7-30 Exception_Rule_Config Fields

Field	Description
CounterMeasure Type	CounterMeasure_Type lists the countermeasure name (suffixed with their short-names).
IMSI_EX_Type	Priority of execution for IMSI Exception Type.
Realm_EX_Type	Priority of execution for Realm Exception Type.
MCC_MNC_EX_Type	Priority of execution for MCC-MNC Exception Type.

This table describes the field details for the Exception_Rule_Config Table.

Table 7-31 Field Details for Exception_Rule_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 Session_Integrity_Validation_Check_SesIntValChk: 16 Default: N/A	List of various supported CounterMeasures.
IMSI_EX_Type	No	Yes	Enumerated Range: 1,2,3 Default: 1	Priority of execution for IMSI Exception Type.
Realm_EX_Type	No	Yes	Enumerated Range: 1,2,3 Default: 2	Priority of execution for Realm Exception Type.

Table 7-31 (Cont.) Field Details for Exception_Rule_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
MCC_MNC_EX_Type	No	Yes	Enumerated Range: 1,2,3 Default: 3	Priority of execution for MCC-MNC Exception Type.

7.3.15.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- If 'CounterMeasure_Exception_Chk' flag is checked in System_Config_Options Table then this table cannot be empty. At least one entry needs be provisioned.
- For any countermeasure same priority cannot be provisioned for multiple Exception types.
- The allowable priority values are 1, 2 and 3.

7.3.16 IMSI_Exception_Config Table

This table is used to configure the list of IMSI range/value to be exempted from Countermeasure business logic execution. It allows to customize the countermeasure behavior using the following options.

Table 7-32 IMSI_Exception_Config Fields

Field	Description
Start_Address	Start Address of the range
End_Address	End Address of the range
AppCmdCst	To decide whether to Exempt IMSI for ApplicationId_And_Command_Code_Consistency_Check countermeasure
RealmWLScr	To decide whether to Exempt IMSI for Origin_Realm_And_Destination_Realm_Whitelist_Screening countermeasure
SubsIdenValid	To decide whether to Exempt IMSI for Subscriber_Identity_Validation countermeasure
SpecAVPScr	To decide whether to Exempt IMSI for Specific_Avp_Screening countermeasure
OhOrCstChk	To decide whether to Exempt IMSI for Origin_Host_And_Origin_Realm_Consistency_Check countermeasure
VplmnORCst	To decide whether to Exempt IMSI for Visited-PLMN-ID and Origin-Realm Consistency Check countermeasure
RealmIMSICst	To decide whether to Exempt IMSI for Realm_And_IMSI_Consistency_Check countermeasure

Table 7-32 (Cont.) IMSI_Exception_Config Fields

Field	Description
DrOrMatch	To decide whether to Exempt IMSI for Destination_Realm_And_Origin_Realm_Match_Check countermeasure
AVPInstChk	To decide whether to Exempt IMSI for Avp_Multiple_Instance_Check countermeasure
AppIdWL	To decide whether to Exempt IMSI for ApplicationID_Whitelist_Screening countermeasure
PreLocChk	To decide whether to Exempt IMSI for Previous_Location_Check countermeasure
TimeDistChk	To decide whether to Exempt IMSI for Time_Distance_Check countermeasure
SrcHostValMme	To decide whether to Exempt IMSI for Source_Host_Validation_MME countermeasure
MsgRateMon	To decide whether to Exempt IMSI for Message_Rate_Monitoring countermeasure
SrcHostValHss	To decide whether to Exempt IMSI for Source_Host_Validation_HSS countermeasure
SesIntValChk	To decide whether to Exempt IMSI for Session_Integrity_Validation_Check countermeasure

This table describes the field details for the IMSI_Exception_Config Table.

Table 7-33 Field Details for IMSI_Exception_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Start_Address	No	Yes	UTF8String Range: 15 digit string. Valid digits are 0 – 9 Default: n/a	Start Address of the range. IMSI: [Default=n/a; Range = A 15 digit string. Valid digits are 0 - 9].
End_Address	No	Yes	UTF8String Range: 15 digit string. Valid digits are 0 – 9 Default: n/a	End Address of the range. IMSI: [Default=n/a; Range = A 15 digit string. Valid digits are 0 - 9].
AppCmdCst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: ApplicationId_And_Command_Code_Consistency_Check To decide whether to Exempt IMSI for CM
RealmWLScr	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Origin_Realm_And_Destination_Realm_Whitelist_Screening To decide whether to Exempt IMSI for CM.

Table 7-33 (Cont.) Field Details for IMSI_Exception_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
SubsIdenValid	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Subscriber_Identity_Validation To decide whether to Exempt IMSI for CM.
SpecAVPScr	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Specific_Avp_Screening To decide whether to Exempt IMSI for CM
OhOrCstChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Origin_Host_And_Origin_Realm_Consistency_Check To decide whether to Exempt IMSI for CM
VplmnORCst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Visited-PLMN-ID and Origin-Realm Consistency Check To decide whether to Exempt IMSI for CM
RealmIMSICst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Realm_And_IMSI_Consistency_Check To decide whether to Exempt IMSI for CM, Apply: Exempt the IMSI for CM
DrOrMatch	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Destination_Realm_And_Origin_Realm_Match_Check To decide whether to Exempt IMSI for CM
AVPInstChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Avp_Multiple_Instance_Check To decide whether to Exempt IMSI for CM
AppIdWL	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: ApplicationID_Whitelist_Screening. To decide whether to Exempt IMSI for CM
PreLocChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Previous_Location_Check To decide whether to Exempt IMSI for CM
TimeDistChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Time_Distance_Check To decide whether to Exempt IMSI for CM

Table 7-33 (Cont.) Field Details for IMSI_Exception_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
SrcHostValMme	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Source_Host_Validation_MME To decide whether to Exempt IMSI for CM
MsgRateMon	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Message_Rate_Monitoring To decide whether to Exempt IMSI for CM
SrcHostValHss	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Source_Host_Validation_HSS To decide whether to Exempt IMSI for CM
SesIntValChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Session_Integrity_Validation_Check To decide whether to Exempt IMSI for CM

7.3.16.1 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. For script compilation, refer to the *DCA Programmer’s Guide*.

- If At least one record is configured in Exception_Rule_Config Table then this table cannot be empty. At least one entry needs be provisioned.
- For IMSI range, value of Start_Range should be always lesser than or equal to End_Range.
- An Individual IMSI value can be configured by specifying only in Start_Address or by specifying same value in both Start_Address and End_Address.
- An IMSI value can be configured only once. No duplicate IMSI is allowed.
- IMSI value/range configured in **Start_Address/End_Address** must be in valid format. Valid IMSI value/range is 15 digit string, valid digits are 0 – 9.

7.3.17 MCC_MNC_Exception_Config Table

This table is used to configure the list of MCC_MNC value to be exempted from Countermeasure business logic execution. It allows to customize the countermeasure behavior using the following options.

Table 7-34 MCC_MNC_Exception_Config Fields

Field	Description
MCC_MNC	Defines MCC+MNC of the network.
AppCmdCst	To decide whether to Exempt MCC_MNC for ApplicationId_And_Command_Code_Consistency_Check countermeasure
RealmWLScr	To decide whether to Exempt IMSI for Origin_Realm_And_Destination_Realm_Whitelist_Screening countermeasure
SubsIdenValid	To decide whether to Exempt MCC_MNC for Subscriber_Identity_Validation countermeasure
SpecAVPScr	To decide whether to Exempt MCC_MNC for Specific_Avp_Screening countermeasure
OhOrCstChk	To decide whether to Exempt MCC_MNC for Origin_Host_And_Origin_Realm_Consistency_Check countermeasure
VplmnORCst	To decide whether to Exempt MCC_MNC for Visited-PLMN-ID and Origin-Realm Consistency Check countermeasure
RealmIMSICst	To decide whether to Exempt MCC_MNC for Realm_And_IMSI_Consistency_Check countermeasure
DrOrMatch	To decide whether to Exempt MCC_MNC for Destination_Realm_And_Origin_Realm_Match_Check countermeasure
AVPInstChk	To decide whether to Exempt MCC_MNC for Avp_Multiple_Instance_Check countermeasure
AppIdWL	To decide whether to Exempt MCC_MNC for ApplicationID_Whitelist_Screening countermeasure
PreLocChk	To decide whether to Exempt MCC_MNC for Previous_Location_Check countermeasure
TimeDistChk	To decide whether to Exempt MCC_MNC for Time_Distance_Check countermeasure
SrcHostValMme	To decide whether to Exempt MCC_MNC for Source_Host_Validation_MME countermeasure
MsgRateMon	To decide whether to Exempt MCC_MNC for Message_Rate_Monitoring countermeasure
SrcHostValHss	To decide whether to Exempt MCC_MNC for Source_Host_Validation_HSS countermeasure
SesIntValChk	To decide whether to Exempt MCC_MNC for Session_Integrity_Validation_Check countermeasure

This table describes the field details for the MCC_MNC_Exception_Config Table.

Table 7-35 Field Details for MCC_MNC_Exception_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
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. E.g. 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. [Range: 10000 - 999999]
AppCmdCst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: ApplicationId_And_Command_Code_Consistency_Check To decide whether to Exempt MCC_MNC for CM
RealmWLScr	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Origin_Realm_And_Destination_Realm_Whitelist_Screening To decide whether to Exempt MCC_MNC for CM.
SubsIdenValid	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Subscriber_Identity_Validation To decide whether to Exempt MCC_MNC for CM.
SpecAVPScr	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Specific_Avp_Screening To decide whether to Exempt MCC_MNC for CM
OhOrCstChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Origin_Host_And_Origin_Realm_Consistency_Check To decide whether to Exempt MCC_MNC for CM
VplmnORCst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Visited-PLMN-ID and Origin-Realm Consistency Check To decide whether to Exempt MCC_MNC for CM
RealmIMSICst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Realm_And_IMSI_Consistency_Check To decide whether to Exempt MCC_MNC for CM.
DrOrMatch	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Destination_Realm_And_Origin_Realm_Match_Check To decide whether to Exempt MCC_MNC for CM

Table 7-35 (Cont.) Field Details for MCC_MNC_Exception_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
AVPInstChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Avp_Multiple_Instance_Check To decide whether to Exempt MCC_MNC for CM
AppIdWL	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: ApplicationID_Whitelist_Screening. To decide whether to Exempt IM MCC_MNC SI for CM
PreLocChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Previous_Location_Check To decide whether to Exempt MCC_MNC for CM
TimeDistChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Time_Distance_Check To decide whether to Exempt MCC_MNC for CM
SrcHostValMme	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Source_Host_Validation_MME To decide whether to Exempt MCC_MNC for CM
MsgRateMon	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Message_Rate_Monitoring To decide whether to Exempt MCC_MNC for CM
SrcHostValHss	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Source_Host_Validation_HSS To decide whether to Exempt MCC_MNC for CM
SesIntValChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Session_Integrity_Validation_Check To decide whether to Exempt MCC_MNC for CM

7.3.17.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- If At least one record is configured in Exception_Rule_Config Table then this table cannot be empty. At least one entry needs be provisioned.

- An unique MCC+MNC value should be configured for each record. Duplicate MCC+MNC value is not allowed.

7.3.18 Origin_Host_Exception_Config Table

This table is used to configure the list of Origin-Host to be exempted from Countermeasure business logic execution. It allows to customize the countermeasure behavior using the following options.

Table 7-36 Origin_Host_Exception_Config Fields

Field	Description
Origin_Host	Defined the Origin-Host (in valid format).
AppCmdCst	To decide whether to Exempt Origin-Host for ApplicationId_And_Command_Code_Consistency_Check countermeasure
RealmWLScr	To decide whether to Exempt Origin-Host for Origin_Realm_And_Destination_Realm_Whitelist_Screening countermeasure
SubsIdenValid	To decide whether to Exempt Origin-Host for Subscriber_Identity_Validation Countermeasure
SpecAVPScr	To decide whether to Exempt Origin-Host for Specific_Avp_Screening countermeasure
OhOrCstChk	To decide whether to Exempt Origin-Host for Origin_Host_And_Origin_Realm_Consistency_Check countermeasure
VplmnORCst	To decide whether to Exempt Origin-Host for Visited-PLMN-ID and Origin-Realm Consistency Check countermeasure
RealmIMSICst	To decide whether to Exempt Origin-Host for Realm_And_IMSI_Consistency_Check countermeasure
DrOrMatch	To decide whether to Exempt Origin-Host for Destination_Realm_And_Origin_Realm_Match_Check countermeasure
AVPInstChk	To decide whether to Exempt Origin-Host for Avp_Multiple_Instance_Check countermeasure
AppIdWL	To decide whether to Exempt Origin-Host for ApplicationID_Whitelist_Screening countermeasure
PreLocChk	To decide whether to Exempt Origin-Host for Previous_Location_Check countermeasure
TimeDistChk	To decide whether to Exempt Origin-Host for Time_Distance_Check countermeasure
SrcHostValMme	To decide whether to Exempt Origin-Host for Source_Host_Validation_MME countermeasure
MsgRateMon	To decide whether to Exempt Origin-Host for Message_Rate_Monitoring countermeasure
SrcHostValHss	To decide whether to Exempt Origin-Host for Source_Host_Validation_HSS countermeasure
SesIntValChk	To decide whether to Exempt Origin-Host for Session_Integrity_Validation_Check countermeasure

This table describes the field details for the Origin_Host_Exception_Config Table.

Table 7-37 Field Details for Origin_Host_Exception_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Origin_Host	Yes	Yes	UTF8String Range: 1-255 characters Default: n/a	Origin-Host (in valid format). Origin-Host 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 "-" & "_". [Range: 1 - 255]
AppCmdCst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: ApplicationId_And_Command_Code_Consistency_Check To decide whether to Exempt Origin-Host for CM
RealmWLScr	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Origin_Realm_And_Destination_Realm_Whitelist_Screening To decide whether to Exempt Origin-Host for CM.
SubsIdenValid	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Subscriber_Identity_Validation To decide whether to Exempt Origin-Host for CM.
SpecAVPScr	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Specific_Avp_Screening To decide whether to Exempt Origin-Host for CM
OhOrCstChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Origin_Host_And_Origin_Realm_Consistency_Check To decide whether to Exempt Origin-Host for CM
VplmnORCst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Visited-PLMN-ID and Origin-Realm Consistency Check To decide whether to Exempt Origin-Host for CM
RealmIMSICst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Realm_And_IMSI_Consistency_Check To decide whether to Exempt Origin-Host for CM
DrOrMatch	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Destination_Realm_And_Origin_Realm_Match_Check To decide whether to Exempt Origin-Host for CM

Table 7-37 (Cont.) Field Details for Origin_Host_Exception_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
AVPInstChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Avp_Multiple_Instance_Check To decide whether to Exempt Origin-Host for CM
AppIdWL	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: ApplicationID_Whitelist_Screening. To decide whether to Exempt Origin-Host for CM
PreLocChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Previous_Location_Check To decide whether to Exempt Origin-Host for CM
TimeDistChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Time_Distance_Check To decide whether to Exempt Origin-Host for CM
SrcHostValMme	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Source_Host_Validation_MME To decide whether to Exempt Origin-Host for CM
MsgRateMon	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Message_Rate_Monitoring To decide whether to Exempt Origin-Host for CM
SrcHostValHss	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Source_Host_Validation_HSS To decide whether to Exempt Origin-Host for CM
SesIntValChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Session_Integrity_Validation_Check To decide whether to Exempt Origin-Host for CM

7.3.18.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- If at least one record is configured in Exception_Rule_Config Table, then this table cannot be empty. At least one entry needs be provisioned.

- An unique Origin_Host value should be configured for each record. Duplicate Origin_Host value is not allowed.

7.3.19 Realm_Exception_Config Table

This table is used to configure the list of realms to be exempted from Countermeasure business logic execution. It allows to customize the countermeasure behavior using the following options.

Table 7-38 Realm_Exception_Config Fields

Field	Description
Realm	Defines the Realm.
AppCmdCst	To decide whether to Exempt Realm for ApplicationId_And_Command_Code_Consistency_Check countermeasure
RealmWLScr	To decide whether to Exempt Realm for Origin_Realm_And_Destination_Realm_Whitelist_Screening countermeasure
SubsIdenValid	To decide whether to Exempt Realm for Subscriber_Identity_Validation countermeasure
SpecAVPScr	To decide whether to Exempt Realm for Specific_Avp_Screening countermeasure
OhOrCstChk	To decide whether to Exempt Realm for Origin_Host_And_Origin_Realm_Consistency_Check countermeasure
VplmnORCst	To decide whether to Exempt Realm for Visited-PLMN-ID and Origin-Realm Consistency Check countermeasure
RealmIMSICst	To decide whether to Exempt Realm for Realm_And_IMSI_Consistency_Check countermeasure
DrOrMatch	To decide whether to Exempt Realm for Destination_Realm_And_Origin_Realm_Match_Check countermeasure
AVPInstChk	To decide whether to Exempt Realm for Avp_Multiple_Instance_Check countermeasure
AppIdWL	To decide whether to Exempt Realm for ApplicationID_Whitelist_Screening countermeasure
PreLocChk	To decide whether to Exempt Realm for Previous_Location_Check countermeasure
TimeDistChk	To decide whether to Exempt Realm for Time_Distance_Check countermeasure
SrcHostValMme	To decide whether to Exempt Realm for Source_Host_Validation_MME countermeasure
MsgRateMon	To decide whether to Exempt Realm for Message_Rate_Monitoring countermeasure
SrcHostValHss	To decide whether to Exempt Realm for Source_Host_Validation_HSS countermeasure
SesIntValChk	To decide whether to Exempt Realm for Session_Integrity_Validation_Check countermeasure

This table describes the field details for the Realm_Exception_Config Table.

Table 7-39 Field Details for Realm_Exception_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
Realm	Yes	Yes	UTF8String Range: 1–255 characters Default: N/A	Realm (in valid format). Exact realm value is required.
AppCmdCst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: ApplicationId_And_Command_Code_Consistency_Check To decide whether to Exempt Realm for CM
RealmWLScr	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Origin_Realm_And_Destination_Realm_Whitelist_Screening To decide whether to Exempt Realm for CM.
SubsIdenValidNo	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Subscriber_Identity_Validation To decide whether to Exempt Realm for CM.
SpecAVPScr	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Specific_Avp_Screening To decide whether to Exempt Realm for CM
OhOrCstChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Origin_Host_And_Origin_Realm_Consistency_Check To decide whether to Exempt Realm for CM
VplmnORCst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Visited-PLMN-ID and Origin-Realm Consistency Check To decide whether to Exempt Realm for CM
RealmIMSICst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Realm_And_IMSI_Consistency_Check To decide whether to Exempt Realm for CM.
DrOrMatch	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Destination_Realm_And_Origin_Realm_Match_Check To decide whether to Exempt Realm for CM
AVPInstChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Avp_Multiple_Instance_Check To decide whether to Exempt Realm for CM

Table 7-39 (Cont.) Field Details for Realm_Exception_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
AppIdWL	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: ApplicationID_Whitelist_Screening. To decide whether to Exempt Realm for CM
PreLocChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Previous_Location_Check To decide whether to Exempt Realm for CM
TimeDistChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Time_Distance_Check To decide whether to Exempt Realm for CM
SrcHostValMme	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Source_Host_Validation_MME To decide whether to Exempt Realm for CM
MsgRateMon	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Message_Rate_Monitoring To decide whether to Exempt Realm for CM
SrcHostValHss	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Source_Host_Validation_HSS To decide whether to Exempt Realm for CM
SesIntValChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Session_Integrity_Validation_Check To decide whether to Exempt Realm for CM

7.3.19.1 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. For script compilation, refer to the *DCA Programmer’s Guide* .

- If at least one record is configured in Exception_Rule_Config Table, then this table cannot be empty. At least one entry needs be provisioned.
- An unique Origin/Destination realm value should be configured for each record. Duplicate Realm value is not allowed.

7.3.20 VPLMN_ID_Exception_Config Table

This table is used to configure the list of VPLMN-ID's to be exempted from Countermeasure business logic execution. It allows to customize the countermeasure behavior using the following options.

Table 7-40 VPLMN_ID_Exception_Config Fields

Field	Description
VPLMN_ID	Defines the VPLMN-ID of the network
AppCmdCst	To decide whether to Exempt IMSI for ApplicationId_And_Command_Code_Consistency_Check countermeasure
RealmWLScr	To decide whether to Exempt IMSI for Origin_Realm_And_Destination_Realm_Whitelist_Screening countermeasure
SubsIdenValid	To decide whether to Exempt IMSI for Subscriber_Identity_Validation countermeasure
SpecAVPScr	To decide whether to Exempt VPLMN-ID for Specific_Avp_Screening countermeasure
OhOrCstChk	To decide whether to Exempt VPLMN-ID for Origin_Host_And_Origin_Realm_Consistency_Check countermeasure
VplmnORCst	To decide whether to Exempt VPLMN-ID for Visited-PLMN-ID and Origin-Realm Consistency Check countermeasure
RealmIMSICst	To decide whether to Exempt VPLMN-ID for Realm_And_IMSI_Consistency_Check countermeasure
DrOrMatch	To decide whether to Exempt VPLMN-ID for Destination_Realm_And_Origin_Realm_Match_Check countermeasure
AVPInstChk	To decide whether to Exempt VPLMN-ID for Avp_Multiple_Instance_Check countermeasure
AppIdWL	To decide whether to Exempt VPLMN-ID for ApplicationID_Whitelist_Screening countermeasure
PreLocChk	To decide whether to Exempt VPLMN-ID for Previous_Location_Check countermeasure
TimeDistChk	To decide whether to Exempt VPLMN-ID for Time_Distance_Check countermeasure
SrcHostValMme	To decide whether to Exempt VPLMN-ID for Source_Host_Validation_MME countermeasure
MsgRateMon	To decide whether to Exempt VPLMN-ID for Message_Rate_Monitoring countermeasure
SrcHostValHss	To decide whether to Exempt VPLMN-ID for Source_Host_Validation_HSS countermeasure
SesIntValChk	To decide whether to Exempt VPLMN-ID for Session_Integrity_Validation_Check countermeasure

This table describes the field details for the VPLMN_ID_Exception_Config Table.

Table 7-41 Field Details for VPLMN_ID_Exception_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
VPLMN_ID	Yes	Yes	UTF8String Range: 6 digit octet string. Default: n/a	The VPLMN-ID valid value will be 6 digit long OctetString with only allowed digits are 0-9 and "F". "F" is allowed to act as filler for 2 digits MNC. So if "F" is present, it must be the 3rd byte string.
AppCmdCst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: ApplicationId_And_Command_Code_Consistency_Check To decide whether to Exempt VPLMN-ID for CM
RealmWLScr	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply
SubsIdenValid	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Subscriber_Identity_Validation To decide whether to Exempt VPLMN-ID for CM.
SpecAVPScr	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Specific_Avp_Screening To decide whether to Exempt VPLMN-ID for CM
OhOrCstChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Origin_Host_And_Origin_Realm_Consistency_Check To decide whether to Exempt VPLMN-ID for CM
VplmnORCst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Visited-PLMN-ID and Origin-Realm Consistency Check To decide whether to Exempt VPLMN-ID for CM
RealmIMSICst	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Realm_And_IMSI_Consistency_Check To decide whether to Exempt VPLMN-ID for CM
DrOrMatch	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Destination_Realm_And_Origin_Realm_Match_Check To decide whether to Exempt VPLMN-ID for CM

Table 7-41 (Cont.) Field Details for VPLMN_ID_Exception_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
AVPInstChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Avp_Multiple_Instance_Check To decide whether to Exempt VPLMN-ID for CM
AppIdWL	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: ApplicationID_Whitelist_Screening. To decide whether to Exempt VPLMN-ID for CM
PreLocChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Previous_Location_Check To decide whether to Exempt VPLMN-ID for CM
TimeDistChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Time_Distance_Check To decide whether to Exempt IMSI for CM
SrcHostValMme	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Source_Host_Validation_MME To decide whether to Exempt VPLMN-ID for CM
MsgRateMon	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Message_Rate_Monitoring To decide whether to Exempt VPLMN-ID for CM
SrcHostValHss	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Source_Host_Validation_HSS To decide whether to Exempt VPLMN-ID for CM
SesIntValChk	No	Yes	Enumerated Range: Do_Not_Apply: 1 Apply: 2, Not_Supported: 3 Default: Do_Not_Apply	CM Type: Session_Integrity_Validation_Check To decide whether to Exempt VPLMN-ID for CM

7.3.20.1 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. For script compilation, refer to the *DCA Programmer's Guide*.

- If at least one record is configured in Exception_Rule_Config Table, then this table cannot be empty. At least one entry needs be provisioned.

- An unique VPLMN-ID value should be configured for each record. Duplicate VPLMN-ID value is not allowed.

7.3.21 RealmIMSICst_Config Table

This table is used to configure the MNC's(3-digit MNC with leading '0') for all the operators around the world. This configuration is used by Realm and IMSI Consistency Check CM. It allows to customize the countermeasure behavior using the following options.

Table 7-42 RealmIMSICst_Config Fields

Field	Description
MCC	Defines Mobile country Code in digits.
MNC_List	List of MNC's (only 3 digit MNC with leading zero) for the MCC. List of MNC's[semicolon (“;”) separated]

This table describes the field details for the Field Details for RealmIMSICst_Config.

Table 7-43 Field Details for RealmIMSICst_Config

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
MCC	Yes	Yes	Integer Range: 100 - 999 Default: n/a	Mobile country Code in digits. [Range = 3 digits Integer]
MNC_List	No	Yes	UTF8String Range: 2048 characters Default: n/a	List of MNC's (only 3 digit MNC with leading zero) for the MCC. List of MNC's[semicolon (“;”) separated]

7.3.21.1 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. For script compilation, refer to the *DCA Programmer’s Guide*.

- RealmIMSICst_Config Table is preconfigured with list of every MCC in the world. This table should be updated eventually with latest MNC added to the network.
- If Realm and IMSI Consistency Check (RealmIMSICst) Countermeasure is enabled, At least one entry needs be provisioned in RealmIMSICst_Config Table.
- This table should be populated with value of MNC.

7.3.22 AVPWLScr_Config Table

This table is used to customize the behavior of `AVP Whitelist Screening (AVPWLScr)` countermeasure by using the following options.

Table 7-44 AVPWLScr_Config Table

Field Name	Unique	Mandatory	Data Type, Range, and Default Value	Description
AVP_Name	Yes	Yes	Data Type: String [UTF8String, Unique name of the Table Field [Default = n/a; Range = A 32-character string. AVP name supports value starting with a digit and it supports value with all numeric characters.]	Name of the AVP as per Diameter AVP Dictionary. AVPs which are part of Grouped AVP can be defined along with its Parent AVP Names (Max 8 level) separated by ";". E.g. BaseAVPName;SubAVPName;AVPName. Each AVP name can't exceed 255 chars.[Range: 1 - 1279]
AVP_Code	No	Yes	Data Type : Integer Range : 1-1677721 Default : NA	AVP Code
AVP_Data_Type	No	Yes	Enumerated Possible Values: 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, Grouped:14, IPFilterRule:15, QoSFilterRule:16 Default: NA	AVP Data Type
Vendor_Id	No	Yes	Data Type : Integer Range : 0- 4294967295 Default : NA	AVP Vendor Id
Command_Code_List	No	Yes	Data Type : UTF8String Range: 2048 Default: NA	List of command code
Diameter_Version	No	Yes	Data Type : Enumerated Possible Values: V1:1,V2:2,V1_V2:3 Default : NA	Version of diameter message
Message_Type	No	Yes	Data Type : Enumerated Possible Values: Request:1,Answer:2,Both:3 Default : NA	Type of Diameter Message

8
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.

8.1 Configuring DSA MEALs

DSA MEALs are pre-populated if DSA is configured using DSA JSON file. For more information, refer to Configuring DSA Business Logic and Database Schema. Alternatively, DSA MEALs can be configured manually using the following procedure.

1. From the NO GUI main menu, navigate to **DCA Framework**, and then **Diameter Security Application**, and then **Custom MEALs**.
2. Click **Insert**.
3. Fill in the fields to define the MEAL.
4. Click **OK** or **Apply**.
5. Repeat Step 2 to 4 for each MEAL defined in the following tables:
   - Table 8-1
   - Table 8-2
   - Table 8-3
   - Table 8-4
   - Table 8-5
   - Table 8-6
   - Table 8-7
   - Table 8-8

8.2 Measurement

8.2.1 ProcessedBy<Countermeasure ShortName>

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

Table 8-1 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 ProcessedBySesIntValChk ProcessedByAVPWLScr ProcessedByOhOrFrmChk ProcessedBySesIdValChk
Template Type	Counter
Measurement Type	Scalar

8.2.2 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. The following table defines the list of Measurement name for each countermeasure type.

Table 8-2 DetectedBy<Countermeasure ShortName> Measurement

Countermeasure Type	Measurement Names
Measurement Name	DetectedByApplIdWL DetectedByAppCmdCst DetectedByRealmWLScr DetectedByOhOrCstChk DetectedByDrOrMatch DetectedByVplmnORCst DetectedByRealmIMSICst DetectedBySubsIdenValid DetectedBySpecAVPScr DetectedByAVPInstChk DetectedByMsgRateMon DetectedByTimeDistChk DetectedByPreLocChk DetectedBySrcHostValHss DetectedBySrcHostValMme DetectedBySesIntValChk DetectedByAVPWLScr DetectedByOhOrFrmChk DetectedBySesIdValChk
Template Type	Counter
Measurement Type	Scalar

8.2.3 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. The following table defines the list of Measurement name for each countermeasure type.

Table 8-3 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 DroppedBySesIntValChk DroppedByAVPWLScr DroppedByOhOrFrmChk DroppedBySesIdValChk
Template Type	Counter
Measurement Type	Scalar

8.2.4 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. The following table defines the list of Measurement name for each countermeasure type.

Table 8-4 RejectedBy<Countermeasure ShortName> Measurement

Countermeasure Type	Measurement Names
Measurement Name	RejectedByAppldWL RejectedByAppCmdCst RejectedByRealmWLScr RejectedByOhOrCstChk RejectedByDrOrMatch RejectedByVplmnORCst RejectedByRealmIMSICst RejectedBySubsIdenValid RejectedBySpecAVPScr RejectedByAVPInstChk RejectedByMsgRateMon RejectedByTimeDistChk RejectedByPreLocChk RejectedBySrcHostValHss RejectedBySrcHostValMme RejectedBySesIntValChk RejetedByAVPWLScr RejetedByOhOrFrmChk RejetedBySesIdValChk
Template Type	Counter
Measurement Type	Scalar

8.2.5 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. For example, failure due to UDR DB not available, Runtime/Internal errors, Roamer type cannot be determined due to unavailability of User-Name AVP etc. The following table defines the list of Measurement name for each countermeasure type.

Table 8-5 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 FailedExecSesIntValChk FailedExecAVPWLScr FailedExecOhOrFrmChk FailedExecSesIdValChk
Template Type	Counter
Measurement Type	Scalar

8.2.6 CreateAndSendMsg

This measurement is used to report number of new diameter request messages created and sent by the DSA application.

Table 8-6 CreateAndSendMsgReqCnt Measurement

Countermeasure Type	Measurement Names
Measurement Name	CreateAndSendMsgReqCnt
Template Type	Counter
Measurement Type	Scalar

This measurement is used to report number of diameter answer messages received (for the request message generated and sent) by the DSA application.

Table 8-7 CreateAndSendMsgAnsCnt Measurement

Countermeasure Type	Measurement Names
Measurement Name	CreateAndSendMsgAnsCnt
Template Type	Counter
Measurement Type	Scalar

This measurement is used to report number of diameter request messages failed during creating/sending by the DSA application.

Table 8-8 CreateAndSendMsgReqFailedCnt Measurement

Countermeasure Type	Measurement Names
Measurement Name	CreateAndSendMsgReqFailedCnt
Template Type	Counter
Measurement Type	Scalar

8.3 SysMetric

8.3.1 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. The following table defines the list of Sysmetric name for each countermeasure type.

Table 8-9 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 VulnerableBySesIntValChk VulnerableByAVPWLScr VulnerableByOhOrFrmChk VulnerableBySesIdValChk
Template Type	Rate
Measurement type	Scalar
KPI Description	Average number of vulnerable messages detected by <Countermeasure LongName>
Generate Alarm	Yes

Table 8-9 (Cont.) VulnerableBy<Countermeasure ShortName> SysMetric

Countermeasure Type	Measurement Names
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
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.

8.3.2 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).

Table 8-10 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

8.4 <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 180 seconds, if the problem still persists after 180 second, alarm is raised again. The following 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 8-11 <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> SesIntValChkExecFailed<Alrm as a suffix> AVPWLScrExecFailed<Alrm as a suffix> OhOrFrmChkExecFailed<Alrm as a suffix> SesIdValChkExecFailed<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

9 Support for Visualization of DSA Vulnerable Message Logs

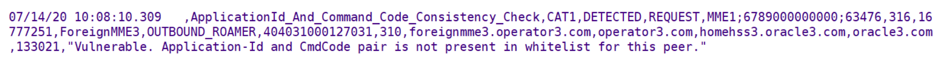
9.1 DSA Vulnerable Message Logging Details

Enable or disable option has been provided to log vulnerable message details into a log file on MPs. For more information, refer to Enable tracing option in System_Config_Options Table. After enabling logging, active SO collects these log files from the MPs and exports them to the SO path `/var/TKLC/db/filemgmt/export/SecurityLogs/dca_logs`.

MPs create the file containing vulnerable message details at `/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. When 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 when 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 when 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 log file has the value of "Timestamp, Applied CounterMeasure Name, category, Applied Action (Discarded/Rejected/Detected), Message Type (Request or Response), Session id, command code, Application id, peer name, Subscriber- Type, Imsi/User-name, MCC, ORIG_HOST, ORIG_REALM, DEST_HOST, DEST_REALM, VPLMNID, and Error text." 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"
  - The snapshot of a sample logs:

Figure 9-1 Sample Log



```
07/14/20 10:08:10.309   ,ApplicationId_And_Command_Code_Consistency_Check,CAT1,DETECTED,REQUEST,MME1;6789000000000;63476,316,16
777251,ForeignMME3,OUTBOUND_ROAMER,404031000127031,310,foreignmme3.operator3.com,operator3.com,homehss3.oracle3.com,oracle3.com
,133021,"Vulnerable. Application-Id and CmdCode pair is not present in whitelist for this peer."
```

The active SO suspends collecting the logs from MP if the available disk space of `/var/TKLC/db/filemgmt/export/SecurityLogs/dca_logs` 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.

9.2 Configuring the Visualization Server for Vulnerable Message Logging

By default, logging of vulnerable message logging is disabled. For more information, refer to Enable Tracing option of System_Config_Options Table. Before enabling logging, the following procedure must be performed on Active SO Server where DSA is running.

This procedure configures a Data Export Job. For more information, refer to the *DSR Online Help*.

1. From the SO GUI main menu, navigate to **Diameter Common**, and then **Visualization Server**.
2. Click **Insert**.
3. Enter a Task Name.
4. Enter a Hostname List as IPV4 addresses.
5. Enter the User name of the Visualization Servers.
6. Enter the Remote Directory Path (the target directory path on the Remote Server).
7. Select the source directory as DSA.
8. Click **OK** to apply the changes.
9. After insert operation, do the ssh key exchange.

Note:

Refer to the Configuring Visualization Server section for LogServer Configuration.

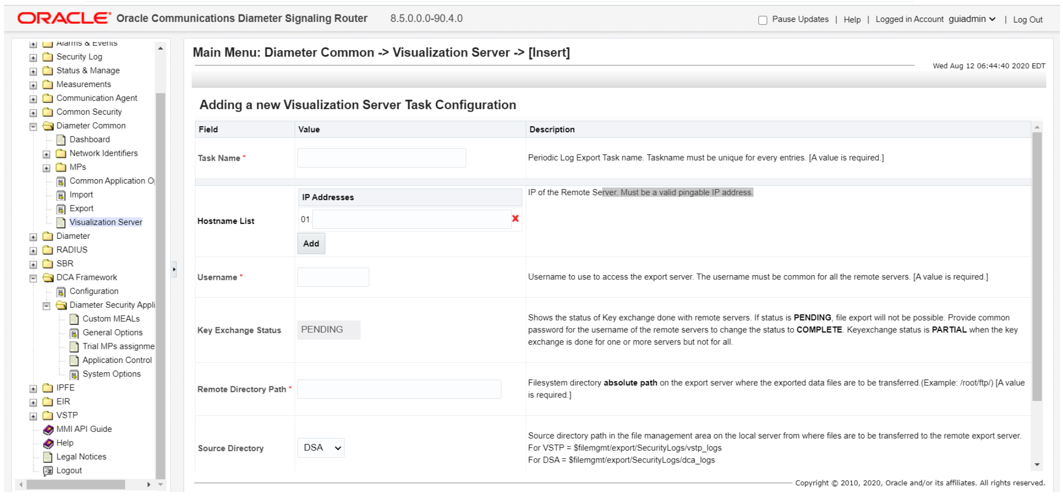
Note:

DSA uses “id_dsa.pub” keys instead of “id_rsa.pub” keys for SSH key exchange. Ensure that the remote server accepts “dsa public keys” for SSH. In the latest version of openssh, DSA is not accepted in the “DEFAULT” mode. Run these commands on a remote server:

- `/usr/bin/update-crypto-policies --set LEGACY`
- `service sshd restart`

GUI configuration on active SOAM server:

Figure 9-2 GUI Configuration 1



Main Menu: Diameter Common -> Visualization Server -> [Insert]

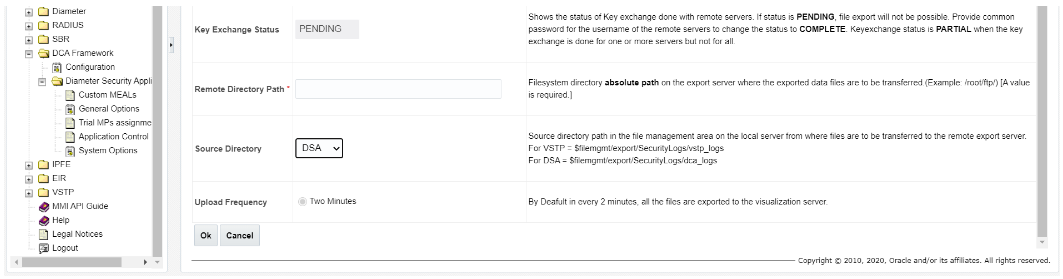
Wed Aug 12 06:44:40 2020 EDT

Adding a new Visualization Server Task Configuration

Field	Value	Description
Task Name *		Periodic Log Export Task name. Taskname must be unique for every entries. [A value is required.]
Hostname List	IP Addresses 01 Add	IP of the Remote Server. Must be a valid pingable IP address.
Username *		Username to use to access the export server. The username must be common for all the remote servers. [A value is required.]
Key Exchange Status	PENDING	Shows the status of Key exchange done with remote servers. If status is **PENDING**, file export will not be possible. Provide common password for the username of the remote servers to change the status to **COMPLETE**. Keyexchange status is **PARTIAL** when the key exchange is done for one or more servers but not for all.
Remote Directory Path *		Filesystem directory **absolute path** on the export server where the exported data files are to be transferred.(Example: /root/ftp/) [A value is required.]
Source Directory	DSA	Source directory path in the file management area on the local server from where files are to be transferred to the remote export server. For VSTP = $filemgmt/export/SecurityLogs/vstp_logs For DSA = $filemgmt/export/SecurityLogs/dca_logs

Copyright © 2010, 2020, Oracle and/or its affiliates. All rights reserved.

Figure 9-3 GUI Configuration 1



Key Exchange Status	PENDING	Shows the status of Key exchange done with remote servers. If status is **PENDING**, file export will not be possible. Provide common password for the username of the remote servers to change the status to **COMPLETE**. Keyexchange status is **PARTIAL** when the key exchange is done for one or more servers but not for all.
Remote Directory Path *		Filesystem directory **absolute path** on the export server where the exported data files are to be transferred.(Example: /root/ftp/) [A value is required.]
Source Directory	DSA	Source directory path in the file management area on the local server from where files are to be transferred to the remote export server. For VSTP = $filemgmt/export/SecurityLogs/vstp_logs For DSA = $filemgmt/export/SecurityLogs/dca_logs
Upload Frequency	Two Minutes	By Deafult in every 2 minutes, all the files are exported to the visualization server.

Ok Cancel

Copyright © 2010, 2020, Oracle and/or its affiliates. All rights reserved.

10

Security Exception Function for CounterMeasure

An option has been provided to define an exception list for each countermeasure to bypass the incoming message for Vulnerability check.

- Exception Function can be enabled or disabled with flag 'CounterMeasure_Exception_Chk' provided in the System_Config_Options Table.
- Exception List can be defined for the following parameters for each Countermeasure:
  - IMSI
  - MCC_MNC
  - REALM (Origin/Destination)
  - ORIGIN-HOST
  - VPLMN-ID
- Exception Function starts executing for the provisioned countermeasure in Exception_Rule_Config Table as per the defined priority sequence of Exception types in the table.
- Execution priority can be configured for only following Exception types IMSI, MCC_MNC and REALM. Remaining exception types ORIGIN-HOST and VPLMN-ID are internally executed along with the REALM exception Type.
- For REALM exception along with Realm in the Realm_Exception_Config Table add either VPLMN-ID in the VPLMN_ID_Exception_Config Table or Origin-Host in the Origin_Host_Exception_Config Table to get it exempted.
- For MCC_MNC exception along with MCC_MNC in the MCC_MNC_Exception_Config Table we have to add VPLMN-ID in the VPLMN_ID_Exception_Config Table to get it exempted.

  Note:

  In some messages, CLR does not have `VPLMN_ID` in the message. In such cases, the `Realm_Exception` config message (where Realm and `VPLMN-ID` are configured on the exception table) and `MCC_MNC_Exception` configs do not get exempted. Here, to get such messages exempt, use IMSI_Exception_Config.

- Each of the exception parameters mentioned above has a separate table to configure the list of values to be bypassed for the countermeasure.
- Following is the mapping of exception parameters and their corresponding tables to configure exception list:
  - IMSI:IMSI_Exception_Config Table
  - MCC_MNC: MCC_MNC_Exception_Config Table

  - REALM: Realm_Exception_Config Table
  - ORIGIN-HOST: Origin_Host_Exception_Config Table
  - VPLMN-ID: VPLMN_ID_Exception_Config Table
- Each exception list table will be used to configure the list of values for which the incoming traffic should bypass Vulnerability check and also configure the countermeasures for which this value should be applied.
- If there is a match found in exception list table for the priority_1 exception type, traffic will be bypass current countermeasure execution and subsequent exception type's check will be skipped.
- If there is no match found in exception list table for the priority_1 exception type, then the subsequent exception type configured as priority_2 will be executed traffic will be bypass current countermeasure execution if match found.
- If no match is found for any exception types configured for a Countermeasure, then current countermeasure will be executed and traffic will not be bypassed.

10.1 Configuring Security Exception List

By default, the security exception function is disabled. Before enabling the exception function, below steps need to be performed on the active SOAM GUI DSA Config tables and Data where DSA is running.

1. Log in to the active SO GUI.
2. From the SO GUI main menu, navigate to **DCA Framework** , and then **Diameter Security Applciation**, and then **Application Control**, and then **Config Tables and Data**.
3. Configure the Exception_Rule_Config Table with the countermeasure for which exception function to be checked and the corresponding exception type priorities.
4. Configure all the Exception list tables with list of values for which the traffic should be bypassed and corresponding Countermeasures for which it should be applied
5. After completing the configuration mentioned in the above steps, Enable the exception function **CounterMeasure_Exception_Chk' flag** in System_Config_Options Table.

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 Activating DSA.
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. The following table provides the configuration tables associated with countermeasures.

   Table A-1 Countermeasure 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 UDR 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 UDR 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 UDR errors are observed.
6. To share the common UDR database, between the DSA of different sites, the SOs need to be under the same NO.

B Configuring Visualization Server

Perform the following procedure to create the LogServer (logstorage, logparser, loggui) stack.

1. Install the following RPMs:
   - logstorage: On all the nodes
   - logparser: Only on master and data nodes
   - loggui: Only on ingestion Node
   - logstorage curator: Only on the master and data nodes
   - Rsync: Only on the master and data nodes
2. Update the `/etc/logstorage/logstorage.yml` configurations file for logstorage search:
   - cluster.name: Name of the stack
   - node.name: Hostname of the node
   - network.host: IPV4 address of the node
   - node.data: true if it’s a data node else false
   - node.master: true if it’s a master node else false
   - discovery.seed_hosts: contains the IPV4 address of all other nodes in the stack (Master node, data node and ingestion node)
   - cluster.initial_master_nodes: On ingestion node specify all the master node IP addresses
   - gateway.recover_after_nodes: Minimum number of master node should be available before processing

   ```
   Sample "/etc/logstorage/logstorage.yml"
   # ---------------------------------- Cluster
   -----------------------------------
   # Use a descriptive name for your cluster:
   cluster.name: vstp
   # ------------------------------------ Node
   ------------------------------------
   # Use a descriptive name for the node:
   node.name: node-1
   # ----------------------------------- Paths
   ------------------------------------
   # Path to directory where to store the data (separate multiple locations
   by comma):
   path.data: /var/lib/logstorage
   # Path to log files:
   path.logs: /var/log/logstorage
   # Set the bind address to a specific IP (IPv4 or IPv6):
   network.host: 10.75.219.169
   # Set a custom port for HTTP:
   ```

```
http.port: 9200
node.master: true
node.data: true
# --------------------------------- Discovery
----------------------------------
#
# Pass an initial list of hosts to perform discovery when this node
is started:
# The default list of hosts is ["127.0.0.1", "[::1]"]
#
#discovery.seed_hosts: ["host1", "host2"]
# Bootstrap the cluster using an initial set of master-eligible
nodes:
cluster.initial_master_nodes: ["node-1"]
# Block initial recovery after a full cluster restart until N nodes
are started:
#gateway.recover_after_nodes: 3
```

3. Update the `/etc/ logparser /conf.d/ logparser.conf` configurations file for logparser on the Master and data node.

```
input {
  file {
   mode => "read"
   path => "/var/log/dummy3/*.csv" □ path of the directory where
logs are present
   start_position => "beginning"
   codec => plain {
            charset => "ISO-8859-1"
         }
   file_completed_action => "delete"
   sincedb_path => "/dev/null"
  }
}
filter {
        if [message] =~ /^\s*$/ {
            drop { }
        }
mutate {
            gsub => ["message", "\t", ""]
       }
grok {
      match => {"message" => "%{GREEDYDATA:TIME},%{WORD:CM_NAME},%
{WORD:Cat},%{WORD:OPERMODE},%{WORD:MSGTYPE},%{NOTSPACE:SESSION_ID},%
{INT:CMD_CODE},%{INT:APP_ID},%{WORD:PEER_NAME},%
{WORD:SUBSCRIBERTYPE},%{INT:IMSI},%{INT:MCC},%{NOTSPACE:ORIG_HOST},%
{NOTSPACE:ORIG_REALM},%{NOTSPACE:DEST_HOST},%{NOTSPACE:DEST_REALM},%
{INT:PLMN_ID},%{GREEDYDATA:ERRORTEXT}"}

     }

 mutate
       {
                     remove_field => [ "message" ]
        }
if "_grokparsefailure" in [tags] {
    drop { }
  }
}
output {
        logstorage {
            hosts => [ "http://A:B:C:D:9200" ] □ Host IP addresses
                index => "dsa" □ Index where all the data will be
captured and can be used on loggui to get all the logs.
    }
}
```

> **Note:**
>
> Only path, index hosts and index field must be updated. Rest of the details will remain the same for DSA.

4. Update the following mandatory fields in `/etc/loggui/loggui.yml`.

   `server.host` is the IP address of the host.

   `logstorage.hosts` is the IP address of the host in which logstoragemodule is running. In our architecture, logstorage, loggui will be running on the same instance/VM.

   `logging.dest:` is used to redirect the log of loggui. “stdout” is the default option.

5. Follow these steps on the **loggui** GUI:
   a. By default, loggui runs on port 5601.
   b. Go to the loggui GUI and navigate to **Management**, and then **loggui**, and then **Create index pattern**.

      It will display all the existing index where data has been generated.
   c. Click **Next Step**, and then **@timestamp**.
   d. Click **Create Index pattern**.

      Now index has been generated and data can be seen in the **discover tab**.
   e. When the index is created, import the sample visualization first (visual_MCC_Cat.ndjson, visual_top_imsi.ndjson), and then import the sample dashboard from the DSA package.
   f. On loggui, navigate to **Management**, and then **Saved Objects**, and then **Import**.

6. Logstorage curator: Curator helps to clear the older logs for an index pattern.

```
command:
  curator /root/curator/delete.yaml --config /root/curator/curator.yml
Need to run a cron job to run curator periodically to clear the data.
crontab -e and add below command in that.
* */2 * * * /usr/bin/curator /root/curator/delete.yaml --config /root/
curator/curator.yml
Sample "/root/curator/delete.yaml" file:
actions:
  1:
    action: delete_indices
    description: >-
      Delete indices older than 30 days (based on index name), for
tomcat-
      prefixed indices. Ignore the error if the filter does not
result in an
      actionable list of indices (ignore_empty_list) and exit
cleanly.
    options:
      ignore_empty_list: True
      timeout_override:
      continue_if_exception: False
      disable_action: False
    filters:
    - filtertype: pattern
      kind: regex
      value: dsa  ----------> specify the regex of the index pattern
      exclude:
    - filtertype: age
      source: creation_date
      direction: older
      unit: days
      unit_count: 30
sample "/root/curator/curator.yml" file:
client:
  hosts:
    - A:B:C:D   IP address of the system.
  port: 9200
  url_prefix:
  use_ssl: False
  certificate:
  client_cert:
  client_key:
  ssl_no_validate: False
  http_auth:
  timeout: 30
  master_only: False
logging:
  loglevel: INFO
  logfile:
  logformat: default
blacklist: ['logstorage', 'urllib3']
```

7. Some recommendations to increase the performance of the server:
   - Use separate index name for each logparser
   - Index name should be of the form: visual_dsa*
   - Example- visual_dsa1, visual_dsa2 and so on
   - In logparser.yml configure pipeline.workers as 32 and pipeline.batch.size as 500. Here in our setup 16 vCPU is there.
   - In jvm.options of logparser increase the heap space:
     – Xms10g
     – Xmx10g

- In jvm.options of logstorage increase the heap size:
  - Xms6g
  - Xmx6g

Note:

After changing all the configuration files, services needs to be restart otherwise configurations will not get updated.

- systemctl restart logparser
- systemctl restart logstorage
- systemctl restart loggui

Note:

Get all the sample configuration files from DSA package.