Signaling Delivery Controller

Session Management

4.4

Catalog Number: FD-014-44-95 Ver. 2

Publication Date: May 2015

Legal Information

Copyright

Trademarks

Patents

Confidential and Proprietary

About F5 Networks

F5 Networks (NASDAQ: FFIV) makes the connected world run better. F5 helps organizations meet the demands and embrace the opportunities that come with the relentless growth of voice, data, and video traffic, mobile workers, and applications—in the data center, the network, and the cloud. The world's largest businesses, service providers, government entities, and consumer brands rely on F5's intelligent services framework to deliver and protect their applications and services while ensuring people stay connected. For more information, visit **www.F5.com**, or contact us at **Tfx_info@f5.com**.

About this Document

Document Name: F5 Signaling Delivery Controller Session Management
Catalog Number: FD-014-44-95 Ver. 2
Publication Date: May 2015

Document Objectives

This document provides an overview of the session management capabilities included in the F5 Signaling Delivery Controller (SDC) Release 4.4. This document describes the session management capabilities as they are configured with the session replication functionality using Tripo.

Document History

Revision Number	Change Description	Change Location
May 2015 – Ver. 2	Added note about session binding and multiple protocols. Added note about setStateless to true in Transformation. Trademark text changed	*See, Session Binding Between Different Sessions in Different Protocols, Configuring a Session Binding Action*

Conventions

The style conventions used in this document are detailed in Table 1.

Table 1: Conventions

Convention	Use
Normal Text	Regular text; style: F5_Normal
Normal Text Bold	Names of menus, commands, buttons, and other elements of the user interface; style: F5_Normal_Bold

Convention	Use
Normal Text Italic	Links to figures, tables, and sections in the document, as well as references to other documents; style: *F5_Normal_CrossRef*
`Script`	Language scripts; style: F5_Scripts
Calibri	File names; F5_Normal_FileName
Table Heading	Table Headings; style: F5_Table Header Text
Table Text	Table Text; style: F5_Table_Text
Note:	Notes which offer an additional explanation or a hint on how to overcome a common problem
Warning:	Warnings which indicate potentially damaging user operations and explain how to avoid them

Table of Contents

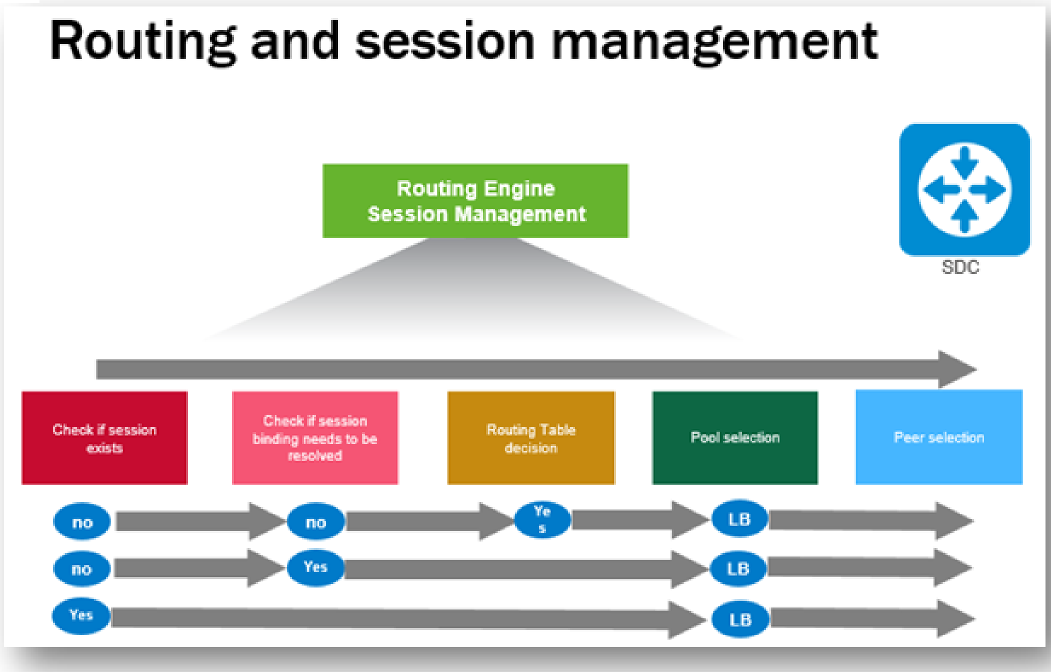
List of Figures

List of Tables

1. Introduction

Session Management refers to the F5® Traffix® Signaling Delivery Controller™ (SDC) abilities to manage routing decisions on a session and sub-session level and replicate session data between SDC sites. The session binding functionality defines the dependency between different sessions initiated from different Remote Peers which share common attributes. Bound sessions are handled as a session bundle composed of several sub-sessions. The session replication functionality defines the replication settings between SDC sites. This document describes the SDC session management functionality and architecture.

Figure 1: Routing and Session Management


Routing and session management
Routing Engine Session Management
SDC
Check if session exists
Check if session binding needs to be resolved
Routing Table decision
Pool selection
Peer selection
no
no
Yes
LB
no
Yes
LB
Yes
LB

2. Session Management and Session Binding

The following sections describe how the SDC manages sessions.

2.1 Session Lookup and Consistency

SDC manages a session throughout its life cycle to route it to its destination. Each generated session is associated with a unique session ID. To make sure a session destination is maintained during an entire session's life cycle, the session destination that is determined upon session initiation is stored in a session repository and is then used by all other transactions within the session. The destination is queried within the session repository using the session ID.

2.2 Session Binding

Session binding refers to SDC's Session Management capability of binding sessions of the same or different Diameter interfaces as well as binding between sessions with different protocols.

The session binding functionality defines the dependency between different sessions initiated from different remote peers which share common attributes. Bound sessions are handled as a session bundle composed of several sub-sessions.

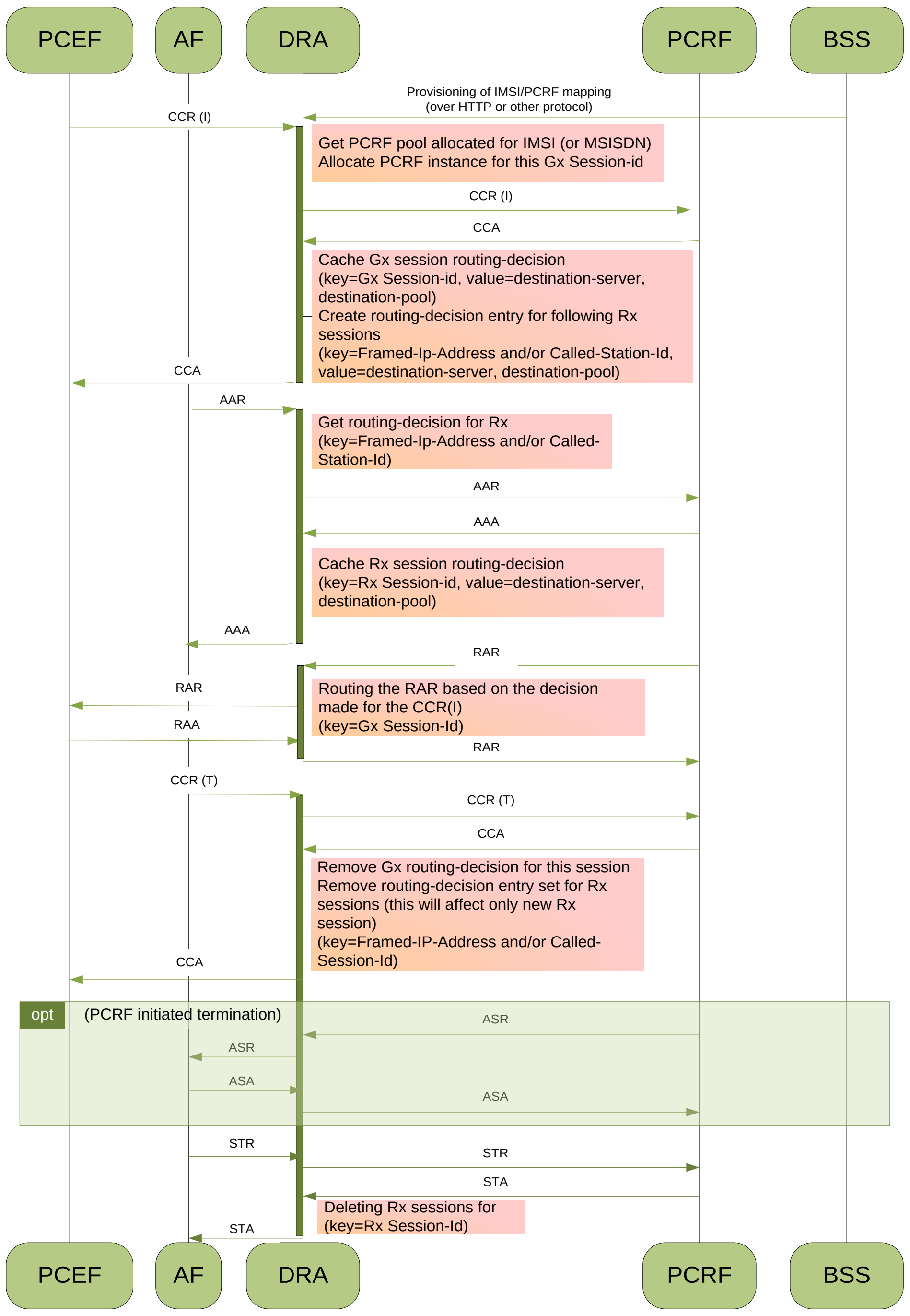
The master session is the session for which the routing selection is performed based on the routing rules or external decision. Slave sessions inherit their routing destination from their master session.

The session binding is performed using Binding Keys. Binding Keys are sets of values extracted from different attributes (e.g. AVPs or XML attributes) of the Master Session and used to bind several session identities.

Using the SDC Web UI, session binding methods may be applied to each specific type of session and special scripts may be composed and run upon each session type creation, session update, and session release. These scripts may be used to log specific transactions according to message content, for example.

For some Diameter reference points, there is a need to bind sessions that originated from different network elements and share common attributes. One such scenario is IP-CAN session binding, as described in 3GPP 29.213. IP-CAN session binding is required to associate between Rx and Gx sessions for the same UE. After the PCEF establishes a Gx session with the selected PCRF for a UE, all Rx sessions associated with the same UE should be routed to the same PCRF.

Figure 2: IP-CAB Session Binding


PCEF
AF
DRA
PCRF
BSS
Provisioning of IMSI/PCRF mapping (over HTTP or other protocol)
CCR (I)
Get PCRF pool allocated for IMSI (or MSISDN) Allocate PCRF instance for this Gx Session-id
CCR (I)
CCA
Cache Gx session routing-decision (key=Gx Session-id, value=destination-server, destination-pool) Create routing-decision entry for following Rx sessions (key=Framed-Ip-Address and/or Called-Station-Id, value=destination-server, destination-pool)
CCA
AAR
Get routing-decision for Rx (key=Framed-Ip-Address and/or Called-Station-Id)
AAR
AAA
Cache Rx session routing-decision (key=Rx Session-id, value=destination-server, destination-pool)
AAA
RAR
RAR
Routing the RAR based on the decision made for the CCR(I) (key=Gx Session-Id)
RAA
RAR
CCR (T)
CCR (T)
CCA
Remove Gx routing-decision for this session Remove routing-decision entry set for Rx sessions (this will affect only new Rx session) (key=Framed-IP-Address and/or Called-Session-Id)
CCA
opt
(PCRF initiated termination)
ASR
ASR
ASA
ASA
STR
STR
STA
Deleting Rx sessions for (key=Rx Session-Id)
STA

2.3 The Session Binding Rules

The session binding is performed according to a list of session binding AVP-based rules. The binding rules consist of parameters which are defined by the session binding attributes – a list of AVPs. Each AVP is assigned a type (i.e. data format: Boolean, regular expression, etc.) and a descriptive text.

2.4 The Session Binding Actions

The following table describes the available session binding actions.

Table 2: Session Binding Actions

Session Binding Method	Description
Cache	Indicates a master session and that the routing is performed based on the routing rule and the routing decision. It creates a binding record entry holding the relevant keys.
External	Indicates that the routing decision of this session creates a binding record holding the relevant keys. The destination is selected by performing a lookup in an external data source.
Resolve	Indicates a salve session and that cached routing decisions should be used for this session. Routing will not be performed if the cached routing decision does not exist.
External or Cache	Indicates a combination of the External and Cache options. If possible, the destination is selected by performing a lookup in the external data source. If this attempt fails, a cached routing decision is used.
Resolve or Cache	Indicates a combination of the Resolve and Cache options. If possible, a cached routing decision is used. If this attempt fails, routing is performed based on the routing rules.
No Binding	Indicates no binding. In this case only the Life-Cycle scripts are applied to a session that matches the session management rule.

2.5 Binding Keys

The session binding rules are applied using Binding Keys. A binding key is a set of values extracted from different attributes of the master session. Each key is comprised of a data record name and value used to mark master sessions and bind slave sessions to them. Multiple keys can be used.

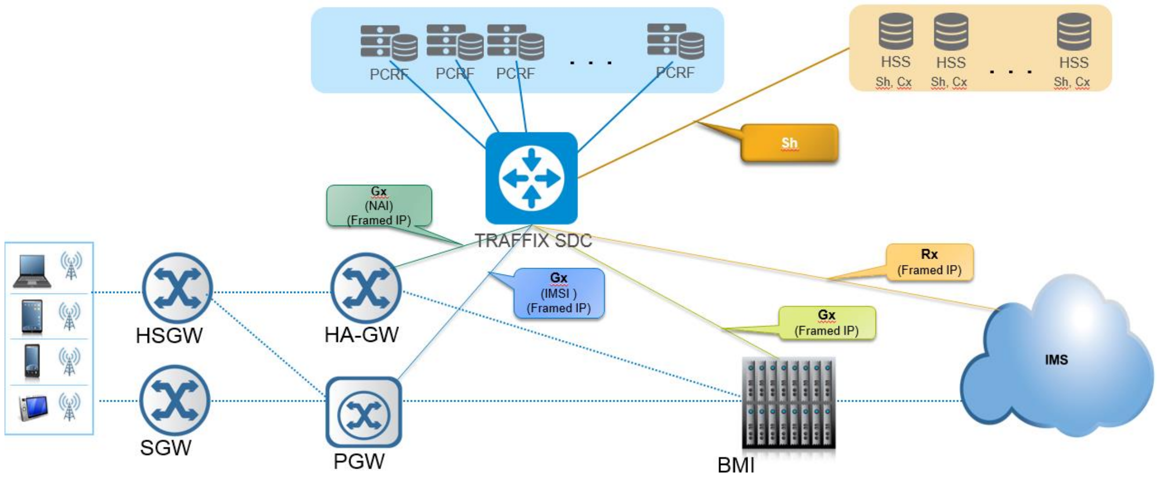
For each master session established, a binding key record is created. When a new session is cached, its keys and their values are cached along with them. The resolved sessions are compared against the selected binding keys. Sessions which share key values with the master session are bound to it as slave sessions.

2.6 Session Binding Between Different Sessions in Different Protocols

In addition to the routine session binding for same protocol, SDC also supports multiple-protocol session binding for linking destination server peers. When multiple-protocol sessions are defined by a master-slave relationship, their destination server peers can then be linked, based on a shared Binding Name. When sessions share a Binding Name, their destination server peers act as a cluster of servers in which each server handles its corresponding protocol session. This session binding is configured as part of the Peer Profile configuration, **Binding Name**. For more information about configuring a **Binding Name**, refer to the *F5 SDC User Guide*.

Note: When a slave-master relationship is defined with multiple protocols and a slave's destination peer (with the first protocol) is closed or out of service, another request is sent to another peer (within the same pool) with the same protocol, and if the second peer is open, then the request is forwarded to its matching peer (as defined by the Binding Name configuration) with a different protocol.

Figure 3: Session Binding Between Different Protocols



2.7 Session Termination and Expiration

When a session expires on one of the SDC sites, the session is queried in the Tripo on the second site. If the session timeout on the second site has not yet reached expiration, the session timeout on the first site is refreshed with the updated timeout and the session is kept alive. Once the sessions on both sites are refreshed, their timeout cannot be refreshed again for as long as the session is kept alive, unless the session is reused.

- **Session Timeout Units** defines the units of the timeout
- **Session Timeout** defines x units of that session that need to be saved
- **Idle Session Timeout** flag indicates that the session's inactivity period will be monitored, where each message received for that session resets the timer

The session expiration synchronization between SDC sites is performed only for sessions that are marked with "Persist and Replicate" session persistence policy.

For more information on how to configure session timeout, refer to the *F5 SDC User Guide.*

2.8 Session Routing When Destination is Unreachable

When a session cannot successfully reach its destination peer, because it cannot find or connect to it, the session will look for another peer in the pool. This session destination fallback is decided using a renewed load balancing mechanism that was assigned to the session, as the destination pool is saved in the session data.

2.9 Session Life Cycle Scripts

Life-Cycle scripts may run upon session creation, session update, and session release. These scripts may be used to log specific transactions according to message content, for example.

2.10 Session Routing during Server Overload Conditions

An overloaded destination server peer is another factor preventing a session to reach its destination. When a peer (or pool of server peers) nears or exceeds its configurable threshold, the Diameter peer server status is changed to “Out of Service” for a defined time interval. When the Remote Peer state is “Out of Service,” no further requests within the session are delivered to it and a new peer is selected from the assigned pool within the session. When a Remote Peer has an “Out of Service Partially” status, the processing of the existing sessions continues by this peer, while new sessions are routed to an alternate, not overloaded destination peer. For more information about Overload Control, refer to the *F5 SDC Overload Control Capabilities*.

2.11 Error handling of Transactions within a Session

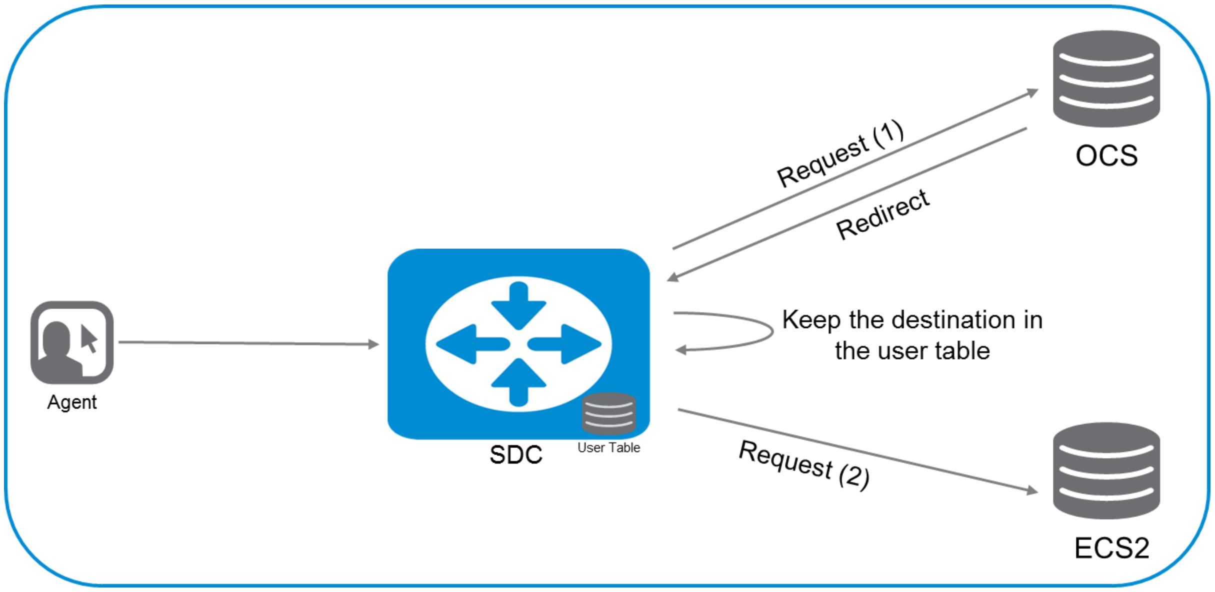
When a transaction within a session is ended with an error, the error handling of the transaction is done using “Check Error in Answer” scripts assigned to the routing rule ID maintained for each session within the session repository.

3. Using Session Repository User Tables

Sometimes clients routing to a dedicated destination based on the subscriber identifier or UE is required. For example, to implement shared data plans, there is a need to route subscribers to a dedicated OCS which manages their allowance bucket. The SDC allows saving routing decisions on a subscriber level.

The user tables enable saving routing decisions per subscriber identifier or UE. The mapping is saved in user tables on the application level. The user table can be created following routing decisions calculated using SDC routing rules, or alternatively, by querying an external data source. The user tables are maintained by the Session Repository and each table can have a maximum of five session keys.

Figure 4: Maintaining the Destination per Subscriber in a User Table



4. Session Repository

Tripo is the session repository that stores all the sessions and user tables.

4.1 Session Data Structure

Session Data is basically structured by the session keys and the session value. The session keys are used for lookup of the session data and session destination for ongoing transactions within the session, as well as, a lookup of the master session when a slave session arrives. Session data of each session includes all the relevant information about the session such as the session ID, Origin Host, Binding Keys, Session Destination. The maximum number of session keys is five, one of which is always the Session ID.

Note: Each Tripo entry is defined as a Key Value structure. The size of the key and the value is predefined as following:

Tripo Key Size: 128 bytes

Tripo Value Size: 1024 bytes

4.2 User Table API

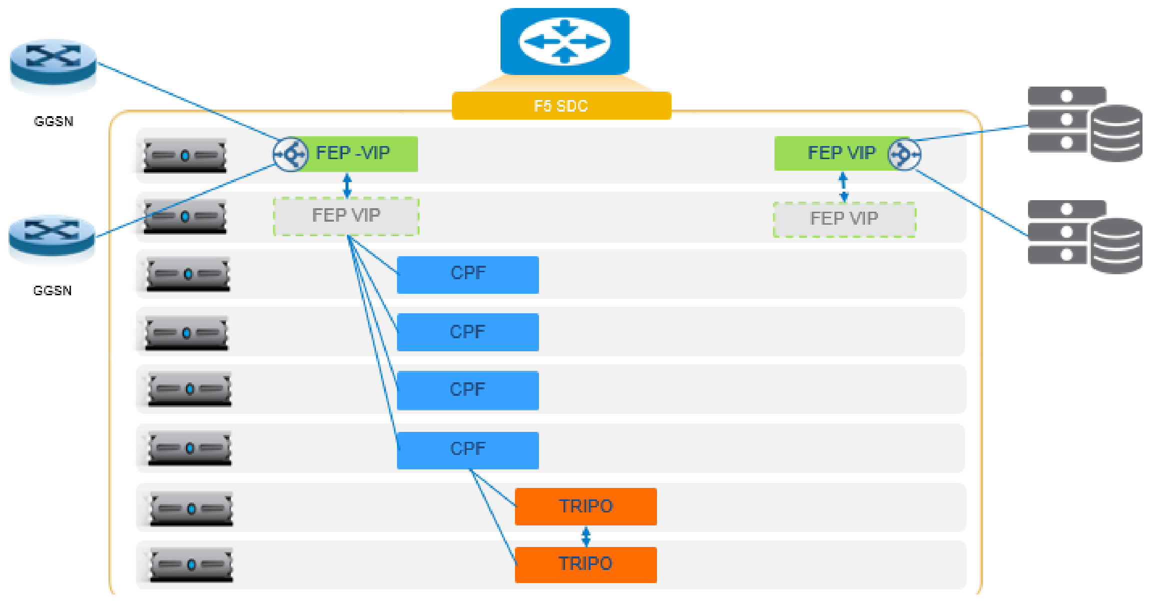
The user tables are created and accessed using scripts. The API for user table methods are:

- Create Table – method used to create a new table. During table creation user can decide if the table should be replicated to the mated SDC or not.
- Put – method that adds a record to the user table.
- Get – method that is used for retrieving the user table entry.

4.3 Communication between CPF and Session Repository

All CPFs are interconnected with the two Tripo instances concurrently. The Tripo can be co-located with other SDC components (CPF or FEP) or it can be configured on a dedicated node if it requires more capacity.

Figure 5: Communication between CPF and Session Repository



4.4 Session Proxy

In some cases, such as Tripo restart or SDC site restart, a session transaction may not find the relevant session in one of the local SDC Tripo instance. In such cases, the session can be configured to be proxied to the mated SDC site for processing.

The session proxy triggering cases:

1. Session binding is failed for sessions with “Persist and Replicate” Session Persistence Policy.
2. Sessions that are explicitly marked as “Proxiable” during incoming transformation and could not be found in the local Session Repository.
3. Site PROXY routing policy is chosen within the routing table for specific session.
4. Sessions are routed to the mated SDC site when it is defined as a destination peer within routing rule.

4.5 Persistency

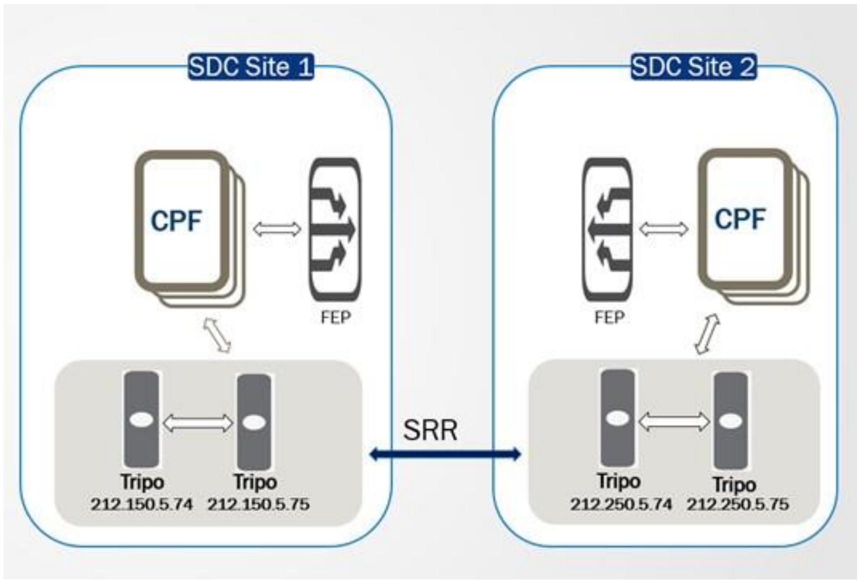
Session data is saved in data tables in Tripo once the session destination is determined. Whether or not a session is saved depends on which persistency policy (**Persist**, **Non Persistence**, **Persist and Replicate**) is configured per session. By saving (persisting) session data (i.e. Session ID, Destination, session stickiness) and Binding Keys in a repository, SDC can then query future incoming requests to see if there is a relevant existing session that meets the defined criteria, thereby allowing the request to be consistently routed to its destination peer.

To make sure the session is replicated to the mated SDC site, a **Persist and Replicate Session Persistence Policy** should be selected.

4.6 High Level Architecture

The Tripo session repository runs on two SDC nodes for high availability. Tripo instances can run on the same nodes as the CPF, or for large scale deployments, Tripo instances can run on dedicated SDC nodes. The Tripo high availability mechanism is Active-Active mode, and Tripo replicates all the sessions between these two instances. In addition to a local high availability, usually a geo redundancy is implemented by deploying two mated SDC sites with Tripo instances on each of the interconnected SDC sites.

Figure 6: Tripo Site Replication High Availability Architecture



In each SDC site, all CPFs are interconnected with the two Tripo instances within their own site that are running in active-active mode. Each new session created in Tripo is automatically replicated to the second Tripo instance in the local SDC site as well as to the mated SDC site.

4.7 High Availability

The Tripo session repository can run on one or two SDC sites. All CPFs in each SDC site are interconnected with two Tripo instances. All Tripo instances within an SDC site are implicitly synched all the time. When SDC is deployed as a mated pair of SDC sites, the synchronization is also performed between all of the mated SDC sites' Tripo instances. All these synchronization mechanisms are implemented to insure that the session data is not lost and the routing to the destination is not affected in the case that one of the Tripo instances is restarted.

A consistency of Tripo entries between mated SDC sites allows SDC peers to access the mated SDC site for session routing.

4.7.1 Communication between Tripo Instances within an SDC Site

All CPFs are interconnected with the two Tripo instances concurrently. All CPFs read and write to both Tripo instances (A/A). The reading and writing is based on a hash algorithm – the same session is written to the same Tripo. If any failure happens, the CPF tries the second Tripo. The timeout between the CPF and Tripo is configurable. Between the Tripo and CPF there is a heartbeat to monitor connection.

The two instances of Tripo are synchronized internally. Each entry has a cookie ID, for duplicate identification and Tripo synchronization.

Table 3: Communication between Tripo Instances within an SDC Site

Event	Action	Result
Ongoing Session Event arrives when one Tripo instance is down	CPF queries the active Tripo	Session handled as is by the active Tripo
Failed Tripo is recovered and second Tripo is up	• Tripo starts re-synching between them • CPF queries second Tripo if first Tripo query was unsuccessful	Session handled as is by the active Tripo

Note: If one of the Tripo instances goes down while the connection between two Tripo instances within a site is also down, all new sessions that were not yet added to the Tripo storage are lost. In addition, this situation can cause a lag in the replication mechanism. However, the system overcomes this situation through its session proxy mechanism. The maximum time period that sessions can potentially be lost in this situation is two seconds.

4.7.2 Communication between Tripo Instances between Mated SDC Sites

The method used for session replication between two mated SDC sites is SRR (Session Replication Request). Each session creation on one of the mated SDC sites leads to the replication of the session on the second SDC site, thus creating a synchronized session repository of the sessions. The SRR is initiated for Tripo entries that are marked with **Persist and Replicate** Session Persistence Policy.

SRRs are sent for each session initiation and termination. SRRs are also sent for session update requests when the session data is changed (e.g. destination of the session is updated).

SRR requests are implemented by Tripo. Upon a successful SRR, Tripo confirms that the SRR was received and that the session data was successfully replicated on the Tripo on the mated SDC site, and then it is erased from the replication queue. Upon an unanswered SRR, the SRR will be continuously resent from the replication queue until confirmation. In the event of a replication queue overflow, all SRRs in the replication queue are erased to free up the overloaded replication queue and then full replication of the Tripo is initiated.

SRR messages are implemented on top of the TCP transport layer and use TCP flow control.

Table 4: Communication between Tripo Instances between Mated SDC Sites

Event	Action	Result
New session cannot be replicated to the mated SDC site because connectivity is lost	Tripo stores the session in a replication queue till the connection is reestablished.	• Once a connection to a mated site is reestablished, the SRR backlog is replicated. • A CCR-U request is sent and the mated site is now able to query its session data to answer the CCR-U • A new session is written to the Tripo instance on the mated SDC site
Session data (i.e. updating session destination) cannot be replicated to the mated SDC site because connectivity is lost	Tripo stores the SRR in a replication queue and continues to send requests to replicate the updated session data to the mated SDC site	• Once a connection to the mated site is reestablished, the SRR backlog is replicated • A CCR-U request is sent and the mated site is now able to query its session data to answer the CCR-U

Event	Action	Result
		• Session data (i.e. new session destination) in the mated SDC site is updated
SRR replication queue is overloaded	Tripo erases the overloaded replication queue and a full replication of the Tripo instance between SDC sites is performed	Once a connection to a mated SDC site is reestablished, upon full replication, all Tripo entries are replicated to the mated SDC site
SDC site is recovered and the session repository (both Tripo instances) is empty	Full session repository synchronization mechanism starts	Session data in the recovered SDC site is updated
No entry found in both Tripo instances	CPF receives negative answers Session is proxied to the second SDC site, upon configuration	In case both sides do not have the entry, SDC reverts to routing table

4.8 Replication Queue

In some cases the communication between mated SDC sites is lost. The communication between SDC sites is monitored by a heartbeat on the application level. During communication loss, the session repositories in the mated SDCs are temporarily not synchronized. SDC supports a resynchronization mechanism, based on maintaining a backlog of SRRs. Tripo stores the backlog of the SRRs in a replication queue. Once the communication between SDC sites is lost, the replication queue begins to fill up with SRRs, and the connectivity is periodically checked by the Tripo. Once the connectivity is reestablished, the SRR backlog replication is started. In the case when the replication queue is overloaded before the connectivity is reestablished, the replication queue is deleted and full site replication is initiated once the connectivity between the sites is back.

4.9 Scalability

Tripo is scalable and has the ability to support a growing amount of entries corresponding to a growth in the number of sessions.

4.10 Session Repository Resources

The SDC session repository (Tripo) is designed to support local failovers. The Tripo is installed on two SDC nodes for local high availability. The basic configuration, which includes two SDC nodes, can support up to 30,000 TPS and up to 6M sessions.

In order to process more than 6M sessions there is a need to separate the servers that handle the sessions from the servers that handle the TPS.

Sessions Server HW Requirements:

- 2x Intel® Xeon® E5-2658 (2.10GHz/8-core/20MB/95W), 128 GB RAM Memory - can process up to 30,000,000 sessions
- 2x Intel® Xeon® E5-2658 (2.10GHz/8-core/20MB/95W), 256GB RAM Memory - can process up to 62,000,000 sessions
- 2x Intel® Xeon® E5-2658 (2.10GHz/8-core/20MB/95W), 512GB RAM Memory - can process up to 125,000,000 sessions

4.10.1 Bandwidth Calculation

During the installation of the SDC mated pair, a connection between SDCs should be established for session repository synchronization mechanism.

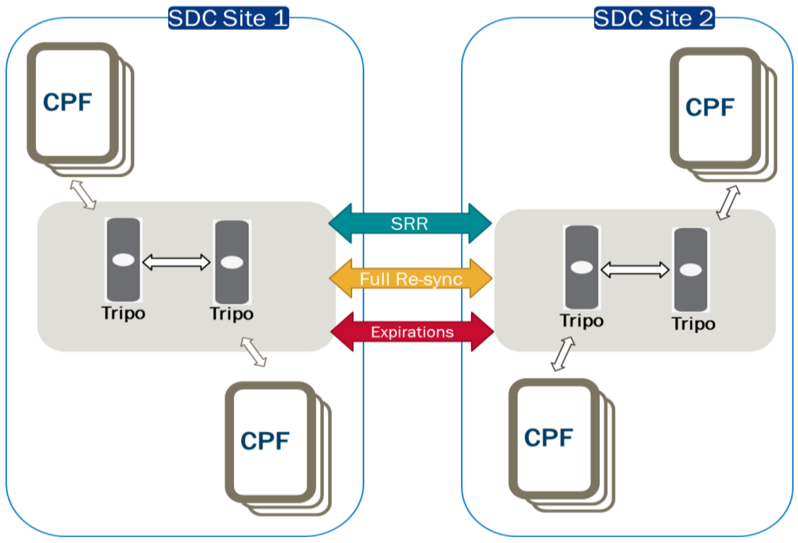
The session synchronization mechanism includes:

- Ongoing SRRs sent each time a Tripo entry is created, deleted or updated
- SRRs backlog sent following SDC reconnection
- Full Tripo synchronization sent when the site is restarted
- Session expiration

The required bandwidth is calculated based on max SRR rate between two mated SDC sites and the SRR message size.

- Maximum SRR Rate is 31K TPS
- SRR Message size and ACK Size is 3.6 KB

Figure 7: Session Synchronization Mechanism



The required Bandwidth is:

31,000 * 3.6KB(Message + Ack Size) * 8 bit * 1.3 (margin) ~= 1.1Gb (full duplex)

4.10.2 Session Repository Memory Allocation

The session repository memory allocation consists of the memory that is available after the memory has been allocated for the SDC and for the Operating System. The memory resources are calculated by the Installer during the installation/upgrade process.

4.11 Configuring a Session Binding Action

From the SDC Web UI, the user can configure a session binding action for a specific session type (i.e. master, slave) and compose special scripts that will run upon each session creation, session update and session release. The user may also create a rule-based session binding. The binding rules consist of parameters which are defined by the Session Binding Attributes – a list of AVPs or other request attributes.

For a description of the available session binding actions, refer to *The Session Binding Actions* section.

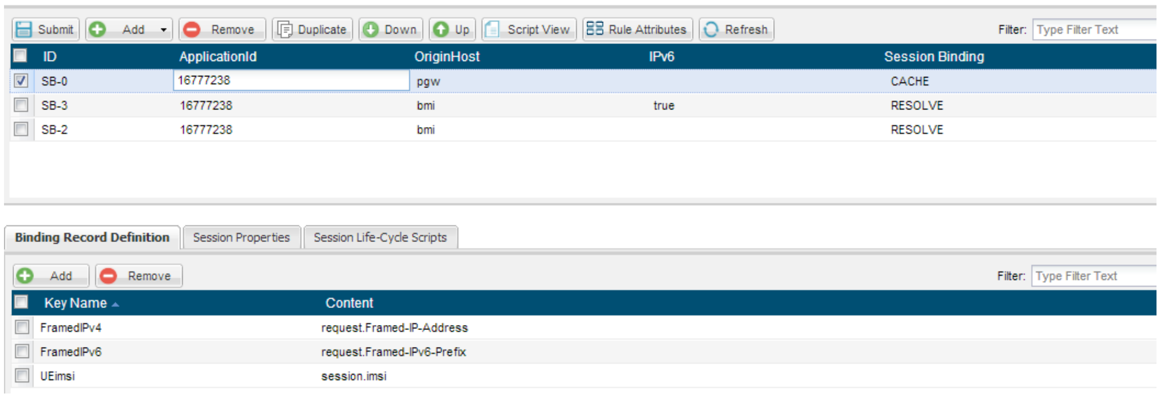
Note: In an SDC deployment without a central EMS configuration, all the session management configuration should be configured identically in both SDC mated sites to ensure session management and binding consistency.

From Routing>Transformation>Pre-routing, when executing the transactionEvent.setStateless(true) script on a resolve (or slave) session, it is considered stateless. This means that for each transaction, the system will always check Tripo for its master's state (based on its binding key) and never for the state of the session (based on its session ID).

4.11.1 Master (Cache) Session Configuration Example

The following example is a configuration for a master (Cache) session.

Figure 8: Session Binding: Cache Example



	ID	ApplicationId	OriginHost	IPv6	Session Binding
✓	SB-0	16777238	pgw		CACHE
	SB-3	16777238	bmi	true	RESOLVE
	SB-2	16777238	bmi		RESOLVE

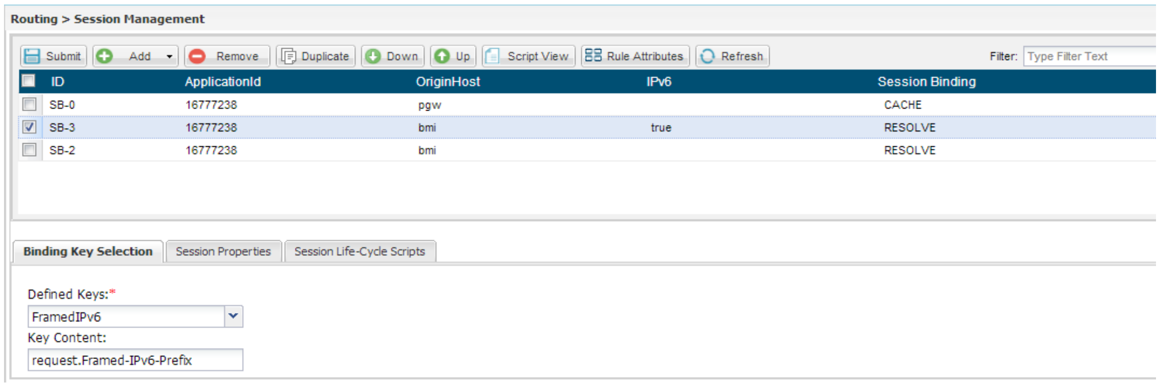
	Key Name	Content
	FramedIPv4	request.Framed-IP-Address
	FramedIPv6	request.Framed-IPv6-Prefix
	UEimsi	session.imsi

- First routing rule – SB-0
  - Is applicable to sessions initiated by PGW and applies the CACHE action where it learns and saves different keys
- The Binding Record definition has 3 keys that the SDC automatically retrieves and saves in its Tripo
  - FramedIPv4 is a key created for IPv4 matching, and it is retrieved from AVP “Framed-IP-Address”
  - FramedIPv6 is a key created for IPv6 matching, and it is retrieved from AVP “Framed-IPv6-prefix”
  - UEimsi is a key created for IMSI matching, and it is retrieved from Subscription-ID-Data grouped AVP.

4.11.2 Slave (Resolve) Session Configuration Example

The following example is a configuration for a slave (Resolve) session.

Figure 9: Session Binding: Resolve Example


Routing > Session Management

Filter: Type Filter Text

	ID	ApplicationId	OriginHost	IPv6	Session Binding
☐	SB-0	16777238	pgw		CACHE
☑	SB-3	16777238	bmi	true	RESOLVE
☐	SB-2	16777238	bmi		RESOLVE

Binding Key Selection | Session Properties | Session Life-Cycle Scripts

Defined Keys:* FramedIPv6

Key Content: request.Framed-IPv6-Prefix

- Second routing Rule –SB3
  - This rule is applicable for sessions initiated by BMI and have the IPv6 “Framed-IPv6-Prefix” AVP
  - The session action for this is RESOLVE – to find the correlated master session that have previously been cached
- The Binding Record definition has a single matching key
  - FramedIPv6 is a key that was defined when creating the “CACHE” action for the master session initiated by PGW
  - “Key Content” is where the SDC needs to retrieve the value of that binding key from the incoming message. In this case it is the “Framed-IPv6-Prefix” AVP

4.12 Configuring a Tripo Component during SDC Installation

During an SDC site installation, the memory needed for session repository resources is calculated and allocated by the Installer.

4.12.1 Enabling Tripo Site Replication between Mated SDC Sites

The Tripo Site Replication feature is disabled by default. Enabling this feature is performed post-installation by setting the **Site Replication** parameter to True. For more information on how to enable this feature, refer to *The F5 SDC Installation Guide*

5. Session Monitoring

The session repository status can be monitored using the session repository statistics in the SDC and EMS Web UI and the session repository SNMP traps in the Web UI and the MIB file. The EMS Web UI also presents Session Life Cycle and Session Management reports.

5.1 Statistics

Table 5 details the session management related statistics available in the SDC and EMS Web UI. For more information, refer to the *F5 SDC User Guide*.

Table 5: Session Management SDC and EMS Web UI Statistics

EMS Title	SDC Title	Statistic Definition
Session Expirations	Session Expirations	Counts the number of expired sessions per CPF
Session Bindings	Session Bindings	Counts the total number of successfully session bindings attempts counted per CPF
Session Binding Failures	Session Binding Failures	Counts the total number of failed session binding attempts counted per CPF
	Count of Proxy To Replicator Messages	Counts the number of proxied session events per CPF
Successful Tripo Queries		Counts the number of successful Tripo queries per Tripo instance
Failed Tripo Queries		Counts the number of failed Tripo queries (entry not found) per Tripo instance
Successfully Added Entries		Counts the number of successfully added Tripo entries per Tripo instance
Failed Addition Attempts (The entry is too long)		Counts the number of failed additional Tripo attempts as a

EMS Title	SDC Title	Statistic Definition
		result of the entry being too long.
Failed Addition Attempts (Tripo overflow)		Counts the number of failed additional Tripo attempts as a result of a Tripo storage overflow
Successfully Deleted Entries		Counts the number of successfully deleted Tripo entries per Tripo instance
Failed Deletion Attempts (entry not found)		Counts the number of failed deletion attempts per Tripo instance(as a result of the entry not being found)
Entry Expiration Events		Counts the number of Tripo entry expiration events per Tripo instance
Sent SRRs	Replication Sent Success Failed send attempts of SRRs	Counts the number of acknowledged/failed/expired SRRs sent to the Mated SDC site. The statistic is counted per Tripo instance that is sending the SRR
Sent SRRs during full SDC site replication	Successfully sent SRRs during full SDC site replication Failed send attempts of SRRs during full SDC site replication	Counts the number of acknowledged/failed/expired SRRs sent during full SDC site replication. The statistic is counted per Tripo instance that is sending the SRR
Sent SRRs during re-synchronization	Successfully sent SRRs during re-synchronization	Counts the number of acknowledged/ failed/expired SRRs sent during re-

EMS Title	SDC Title	Statistic Definition
	Failed send attempts of SRRs during re-synchronization	synchronization of the replication queue. The statistic is counted per Tripo instance that is sending the SRR
Received SRRs	Received SRRs Failed Received SRR attempts	Counts the number of received SRRs successful/failed attempts. The statistic is counted per Tripo instance that is receiving the SRR

5.2 SNMP Traps

The SDC sends the following session management related traps. For more information, refer to the *F5 SDC SNMP Guide*.

Table 6: Session Management Traps

Name	Description
tripoStartedSuccessfully	Indicates that the Tripo instance started successfully
tripoStoppedSuccessfully	Indicates that the Tripo instance was shut down
tripoMateConnectSuccessful	Indicates that the Tripo instance successfully connected to a second Tripo instance within the SDC site
tripoMateConnectPending	Indicates that the Tripo instance connection to a second Tripo instance within the SDC site is pending. Once the problem is solved, the tripoMateConnectSuccessful trap is sent.
tripoMateReplied	Indicates that the first message replication between two connected Tripo nodes succeeded
tripoStorageOverflow	Indicates that the Tripo storage overflowed and the additional entry attempt failed

Name	Description
tripoReplicationQOverflow	Indicates that the replication queue has overflowed.
tripoSRRQueueResyncStarted	Indicates that the resynchronization of the replication queue between the SDC mated sites has started
tripoSRRQueueResyncFinished	Indicates that the resynchronization of the replication queue between the SDC mated sites has finished
tripoFullResyncStarted	Indicates that the full Tripo resynchronization (recovery) between the SDC mated sites has started
tripoFullResyncFinished	Indicates that the full Tripo resynchronization (recovery) between the SDC mated sites has finished
tripoResyncFailed	Indicates that at least one of the entries failed to be synchronized during the last minute of full Tripo resynchronization between the SDC mated sites
tripoSrrEnabled	Indicates that replication between SDC sites by Tripo is enabled. This trap is activated upon Tripo starting or after the Tripo Site Replication parameter has been manually changed
tripoSrrDisabled	Indicates that replication between SDC sites by Tripo is disabled. This trap is activated upon Tripo starting or after the Tripo Site Replication parameter has been manually changed

5.3 Session Logging

The SDC and Tripo can be configured to generate logs for monitoring session life cycle events, session replication, and session errors. These logs can be used to help troubleshoot when stateful sessions fail to route or replicate. You can customize a session log by adding session attributes to a session log. For session life cycle events, logs are found in the CPF session logs and for session replication, logs are found in the Tripo session logs.

5.3.1 CPF Session Logs

The SDC maintains two log files: one for session events and one for session error events:

- */opt/traffix/sdc/logs/session_output*
- */opt/traffix/sdc/logs/session_error*

Note: By default, the CPF session log functionality is disabled and can be enabled and configured in the Web UI (**Administration** > **Specific Site Settings** > **Logging** > **Enable Session Log**).

5.3.1.1 Session Logging of Session Life Cycle Events

The following is a description of the parameters included in a session log for session created and removed events. In addition, session logs are also generated upon updating sessions.

Table 7: Session Created Log

Parameter	Definition
<Timestamp>	Timestamp
<Session ID>	Session ID
<Session Action>	Created Session are indicated with a "C" tag Updated Session are indicated with a "U" tag Removed Session are indicated with a "R" tag
<Origin Peer>	Name of the peer from which the session request originated
<Destination Pool>	Name of the pool that the session request is sent to
<Destination Peer>	Name of the peer that the session request is sent to
<Session Type>	Session types are indicated with an: "M" tag - for master and a "S" tag for slave
<Master session ID>	This is displayed for slave sessions only. There might be cases when a session is marked as a slave, but no master session ID is presented. This can happen when the slave session could not resolve its master session
< SM Row ID>	The matching row in the session management table.

Parameter	Definition
<Binding Keys>	The list of binding keys
<Session Source>	The initiation of the session indicator. An "L" tag is for a session that is created following session initiation request. "SRR" tag is for a session that is created following SRR received from mated SDC site
<Session Timeout>	Defines x units of that session that need to be saved

The following is an example of a log entry:

<Time Stamp>;<Session ID>;<Session Action<Origin Peer>;<Destination Pool>; <Destination Peer>; Session Type>;<Master session ID>, < SM Row ID>; <Binding Keys>;<Session Source>, <Timeout>

```
2015-04-2709:11:03,967
%98.1724%C%client_a%p_diameter_01%s_diameter_3868%S%98.15285%SB-
1%L%30000%;
```

Table 8: Session Removed Log

Parameter	Definition
<Timestamp>	Timestamp
<Session ID>	Session ID
<Session Action>	Created Session are indicated with a "C" tag Updated Session are indicated with a "U" tag Removed Session are indicated with a "R" tag
<Origin Peer>	Name of the peer from which the session request originated
<Destination Pool>	Name of the pool that the session request is sent to
<Destination Peer>	Name of the peer that the session request is sent to
<Session Type>	Session types are indicated with an: "M" tag - for master and a "S" tag for slave
<Session Release Reason>	Sessions released are indicated with a "RE" tag and expired sessions are indicated with an "EX" tag.

Parameter	Definition
<Session Source>	The initiation of the session indicator. An “L” tag is for a session that is created following session initiation request. “SRR” tag is for a session that is created following SRR received from mated SDC site.

The following is an example of a log entry:

<Time Stamp>;<Session ID>;<Removed Session>;<Origin Peer>;<Session Type;< Session Release Reason>;<Session Source>

```
2014-01-21 10:05:56,356 1.Session.master.xxx;R;origin_host_1;M;EX;SRR
```

5.3.1.2 Session Logging for Session Errors

The following is a description of the parameters included in a session logging error. Session error logs are created when a Tripo instance is down or a session cannot be found.

Table 9: Session Error Log

Parameter	Definition
<Timestamp>	Timestamp
<Session ID>	Session ID
<Origin Peer>	Name of the peer from which the session request originated
<Session Source>	The initiation of the session indicator. An “L” tag is for a session that is created following session initiation request. “SRR” tag is for a session that is created following SRR received from mated SDC site.
<Operation Type Models>	The following operation types are supported: GET, ADD, REMOVE
<Error type symbols >	The following error type symbols are supported: • TD - Tripo is down • NF – a session is neither found in a repository nor found in a session management table • IK- null binding key value in slave session • BF - binding failure

The following is an example of a log entry:

<Time Stamp>;<Session ID>;< Error type symbols>;<Origin Peer>;< Session Type;< Operation Type Models >

2014-01-20 12:08:03,807 9.Session.xxx;NF;origin_host_1;GET

5.3.2 Tripo Session Logs

The Tripo maintains the following log files:

- */opt/traffix/sdc/logs/session_output*
- */opt/traffix/sdc/logs/session_error*
- */opt/traffix/sdc/logs/session_mate_rep*
- */opt/traffix/sdc/logs/session_srr*

Each Tripo session log type contains different types of messages as follows:

- Session_output – contains incoming messages from the CPF and session expiration
- Session_mate_rep – contains mate replication information
- Session_srr – contains the response to the SRR message and incoming messages from the mated Tripo instance
- Session_error – contains all messages when the replication to the mated site or Tripo mate fail (after maximum allowed retries)

Note: By default, the Tripo session log functionality is disabled and can be enabled and configured from the UI_LogCfg utility. The log size is configurable and the system is configured to support rotation so that when the log size reaches the defined file size, it rotates the log file to a new file.

5.3.2.1 Tripo Session Log Levels

Each Tripo session log type is configurable by the following log level categories:

Note: An EMERGENCY log level is the default level for Session_output, Session_mate_rep and Session_srr.

- ALERT Level
  - Session_output log – consists of incoming messages from a CPF only for the primary key

    - Session_error log –consists of all session request errors (this is the default value)
- INFO Level
    - Session_output log–consists of:
        - incoming messages from a CPF for each binding key
        - expired sessions
    - Session mate_rep log – consist of mate replication responses
    - Session_srr log – consist of:
        - incoming messages from the mated site for each binding key
        - site replication response
- DEBUG Level
    - Session_output log – consists of:
        - incoming "Get" requests
        - responses for the "Get" request
    - Session_srr log – consists of the incoming requests from the mated site that were saved in the session repository

5.3.2.2 Tripo Session Log Messages

The following table describes the parameters included in a Tripo session log.

Table 10: Tripo Session Log Message Parameters

Parameter	Definition
<Timestamp>	Timestamp Note: The timestamp format is displayed in numeric time in seconds since the Epoch (January 1, 1970).
<CPF>	Source CPF IP and port
<ThrId>	Thread ID
<Origin>	Origin: CLIENT, MASTER, SLAVE, SERVER, SITE
<Opaque>	Opaque ID
<K0, K1, …>	Keys with their lengths and values in hexadecimal format
<MainKeyIdx>	Main key index

Parameter	Definition
<CurKeyIdx>	Current key index
<Ageout>	Time to live
<Flags>	Flags
<Required Action>	ADD, UPDATE, DELETE
<Status>	Received, Fail, Responded, Expiration
<Actual Action>	A – add, G – get, D – delete, U – update

The following is an example of a Tripo Session_output log record:

<Timestamp>, <CPF>, <ThrId>, <Origin>, < K0,>, <MainKeyIdx>, <CurKeyIdx>, <Ageout>, <Flags>, <Required Action>, <Status>, <Actual Action>

1428223766 CPF[127.0.0.1:34899] ThrId[10] Origin[CLIENT]; K0[(len=56) 00 01 02 00 00 4f 4d 41 48 4e 45 4c 56 2d 43 50 47 2d 30 31 2e 6c 74 65 2e 73 70 72 69 6e 74 2e 63 6f 6d 3b 31 35 38 38 32 31 3b 31 34 34 31 32 30 39 31 32 35 3b 33 32]; MainKeyIdx[0]; CurKeyIdx[0]; Ageout[60000 ms]; Flags[9];ADD -> Received; A

Glossary

The following table lists the terms and abbreviations used in this document.

Table 11: Terms and Abbreviations

Term	Definition
AAA	Authentication, Authorization and Accounting
ACL	Access Control List
AF	Application Function
Answer	A message sent from one Client/Server Peer to the other following a request message
API	Application Programming Interface
AVP	Attribute Value Pair
CLI	Command Line Interface
Client Peer	A physical or virtual addressable entity which consumes AAA services
CPF	Control Plane Function
Data Dictionary	Defines the format of a protocol’s message and its validation parameters: structure, number of fields, data format, etc.
DEA	Diameter Edge Agent
Destination Peer	The Client/Server peer to which the message is sent
DRA	Diameter Routing Agent
EMS Site	Element Management System Site
FEP-In	In-Front End Proxy
FEP-Out	Out-Front End Proxy
Geo Redundancy	A mode of operation in which more than one geographical location is used in case one site fails

Term	Definition
HA	High Availability
HSS	Home Subscriber Server
HTTP	Hypertext Transfer Protocol
IMS	IP Multimedia Subsystem
JMS	Java Message Service
KPI	Key Performance Indicator
LDAP	Lightweight Directory Access Protocol
LTE	Long Term Evolution
Master Session	The session for which the routing selection is performed based on the routing rules (Slave Sessions are applied with routing rules inherited from the Master Session)
MME	Mobility Management Entity
NGN	Next Generation Networking
Node	Physical or virtual addressable entity
OAM	Operation, Administration and Maintenance
OCS	Online Charging System
Origin Peer	The peer from which the message is received
PCEF	Policy and Charging Enforcement Function
PCRF	Policy and Charging Rules Function
PLMN	Public Land Mobile Network
Pool	A group of Server Peers
RADIUS	Remote Authentication Dial In User Service
Request	A message sent from one Client/Server peer to the other, followed by an answer message
SCCP	Signaling Connection Control Part

Term	Definition
SCTP	Stream Control Transmission Protocol
SDC	Signaling Delivery Controller
SDC Site	The entire list of entities working in a single site
Server Peer	A physical or virtual addressable entity which provides AAA services
Session	An interactive information interchange between entities
Slave (Bound) Session	A session which inherits properties from a master session
SNMP	Simple Network Management Protocol
SS7	Signaling System No. 7
TCP	Transmission Control Protocol
TLS	Transport Layer Security
Transaction	A request message followed by an answer message
Tripo	Session data repository
UDP	User Datagram Protocol
UE	User Equipment
URI	Universal Resource Identification.
Virtual Server	A binding point used by SDC to communicate with the Remote Peers (Clients and Servers)
VPLMN	Visited Public Land Mobile Network
Web UI	Web User Interface
WS	Web Service