# Implementing Data Plane Security

The data plane security (DPSec) feature prevents traffic injection from external sources into a LISP VPN. DPSec relies on the integrity of the routing locator (RLOC) network which is built using unicast reverse path forwarding (URPF) support.

In order to enable LISP shared mode segmentation without incurring the overhead of authentication and encryption, the DPSec feature uses a mechanism called Source RLOC Decapsulation Filtering that enforces URPF on the network. The URPF configured on the network disseminates lists of acceptable RLOCs, traffic from which will already have been proofed by URPF. This makes it impossible to spoof the source RLOC address of LISP control and data packets. DPSec feature uses the list of valid encapsulation sources for each EID instance to filter LISP data packets during decapsulation at xTRs and PxTRs.

### Feature History for Data Plane Security

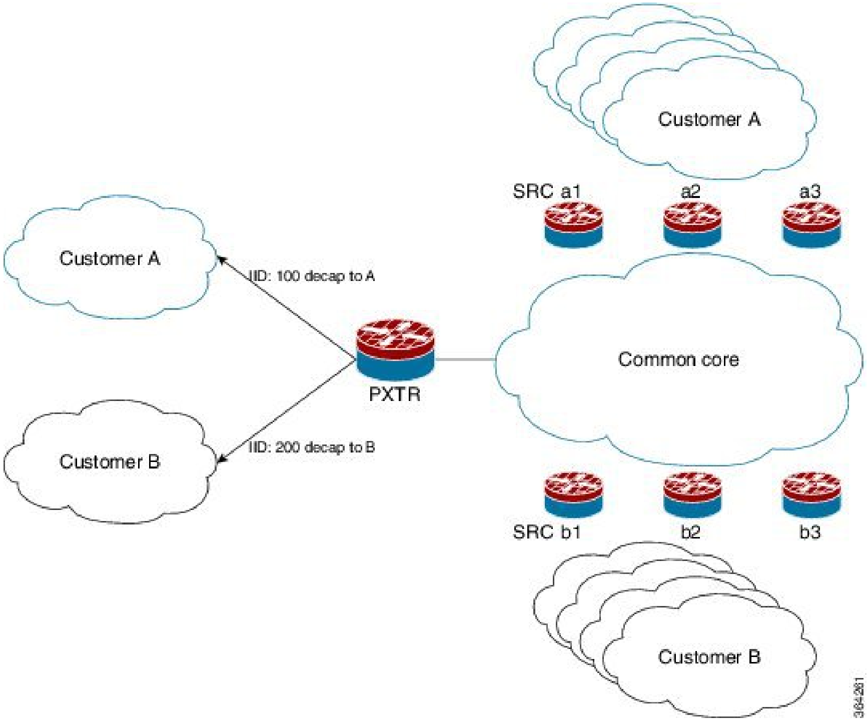
| | |
|---|---|
| Release 5.3.0 | This feature was introduced. |

# Information about Data Plane Security

The LISP Data Plane Security feature ensures that only traffic from within a LISP VPN can be decapsulated into the VPN. In order to understand data plane security you must be familiar with the following features and concepts it supports:

## Source RLOC Decapsulation Filtering

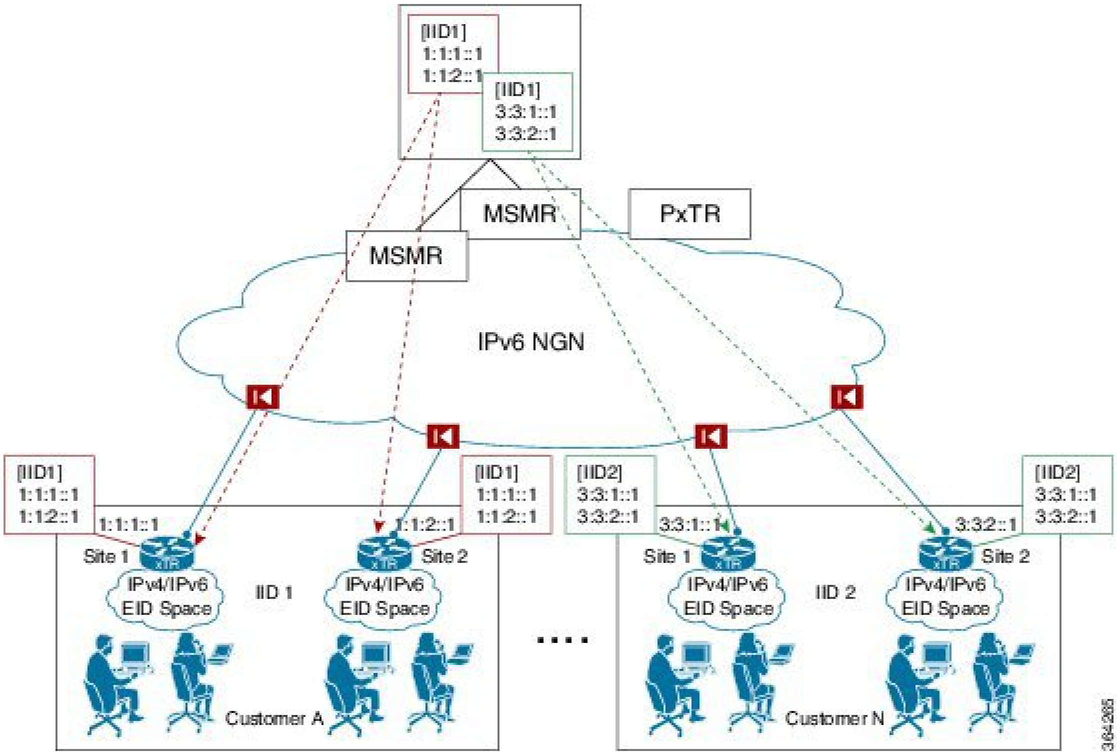
This illustration shows blue and black customer networks using LISP EID instance ID (IID) 100 and 200, respectively, over a shared common RLOC core. When decapsulating LISP data packets, the PxTR validates that packets carrying instance ID 100 have a source (SRC) RLOC in the encapsulation header of either a1, a2 or a3. Similarly, for instance ID 200 the PxTR validates that the RLOC source is b1, b2 or b3.



LISP encapsulated data packets that do not carry a valid source RLOC are dropped. The combination of RLOC space URPF enforcement and source RLOC-based decapsulation filtering ensures that it not possible for a source that is not member of a tenant VPN to inject traffic into the VPN.

# EID Instance Membership Distribution

To deploy the source RLOC filtering solution, an automated mechanism is required to push the list of valid RLOCs through the mapping system to the boxes performing decapsulation. This function is performed by the Map-Servers. The Map-Servers construct the EID instance ID RLOC membership list using the RLOC information in the received mapping records in Map-Register messages. The complete list is then pushed out to all the xTRs and PxTRs that must decapsulate packets for the VPN identified by the EID instance ID.



In the example, the Map-Servers build a separate VPN (EID instance) membership list for each customer and then push the contents of the list out. The two xTRs for customer A each register their site RLOCs. They each receive back from the Map-Server the complete list of RLOCs of all the xTRs for customer A. The received list is used to filter decapsulated traffic and enforce the data plane security.

When PxTRs are being used (for example to provide internet connectivity to the VPN) then the xTRs participating in the VPN must accept and decapsulate the LISP data packets sent by the PxTRs. The RLOC addresses used by the PxTRs have to be included in the EID instance membership list communicated to the xTRs by the Map-Server. The PxTRs do not register EID prefixes with the Map-Server that the Map-Server can use to discover the PxTR RLOCs. Those RLOCs will have to be manually configured on the Map-Server.

The EID instance membership lists built by Map-Servers are only useful to boxes participating in the VPN. As an added security measure, the Map-Server will only communicate the contents of the membership list for an EID instance to xTRs and PxTRs that are members of that VPN.

# Map-Server Membership Gleaning and Distribution

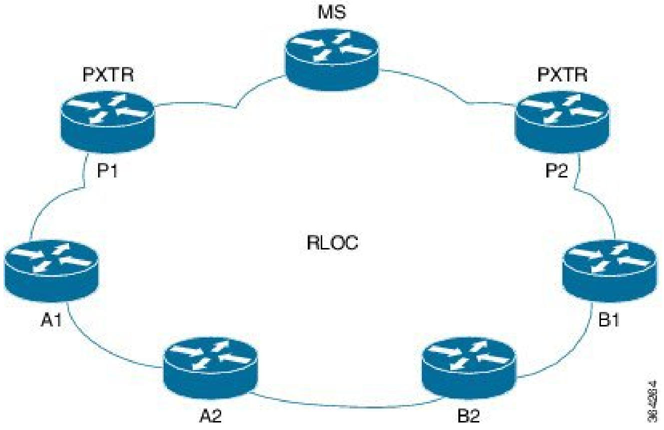
A LISP Map-Server is responsible for tracking the per EID instance membership and distributing it to (P)xTRs. Use the **map-server rloc members distribute** command to enable this functionality. The command configures the Map-Server to:

- Build a list of RLOC addresses using Map-Registrations and configuration from which to accept reliable transport sessions.
- Accept TCP connections from (P)xTRs in above list.
- Glean and maintain per EID instance RLOC membership from received Map-Register messages.

- Serve EID instance membership requests received over the reliable transport sessions from (P)xTRs and distribute membership information.

The per EID instance membership list that the MS gleans from received registrations can be extended or completely overridden through the **map-server rloc members {add | override}** configuration command. The command allows the user to extend the discovered xTR RLOC membership with PxTR RLOC addresses. The extended membership list is used to determine whether to allow a membership request that is received over a reliable transport session. Only requests from xTRs that have registrations in an EID instance are allowed. The extended membership list is then pushed to decapsulating devices implementing the data plane security feature that will then be able to accept encapsulated packets sent by both valid xTRs and PxTRs.

To prevent unauthorized attempts to establish TCP connections with the Map-Server, a list of allowed locators from which to accept connections is built. The list contains the RLOC addresses of the registering xTRs as well as the RLOC addresses configured in membership list extensions. Note that there is a single list from which to accept connections per RLOC address family (it is not EID instance specific).



As an example consider the network in the above figure with two VPNs. VPNs A and B each have two xTRs A1/A2 and B1/B2 respectively. The membership of VPN A is extended on the MS through the “map-server rloc members add …” configuration to include PxTR RLOC address P1. The membership of VPN B is extended to include PxTR RLOC address P2. The resulting lists maintained by the MS are:

- EID instance 1 (VPN A) membership: A1, A2, P1
- EID instance 2 (VPN B) membership: B1, B2, P2
- Locators from which to accept TCP session: A1, A2, P1, B1, B2, P2

The Map-Server may receive an EID instance membership request for one or more EID instances through each established reliable transport session. PxTRs will typically request the membership of multiple instances through the single session that they establish with the MS. The Map-Server must provide full membership refreshes and incremental updates for each of the accepted requests.

When a membership request is received by an MS and the peer (P)xTR originating the request is not a member of the EID instance to which the request pertains, then the MS will reject the request and return a membership NACK message to the (P)xTR. Note that such an event may occur during normal operation as the TCP session and membership request from an xTR may be received before the corresponding Map-Register message that places it in the EID instance membership. If after an EID instance membership request has been accepted by an MS, the requesting (P)xTR is removed from the EID instance membership because of a registration expiration or configuration change, then the MS will send a Membership-NACK message to the (P)xTR to indicate that it is no longer receiving membership updates for that instance.

When a Map-Server restarts, it must first discover and rebuild the EID instance membership lists before serving membership requests. Specifically the MS must hold off sending any membership refresh end messages for EID instances that do not have a complete membership list. On the MS the LISP control plane will wait to receive registrations before considering the membership list complete. The following conditions must be met:

- At least one registration period has elapsed (one minute) after the first registration was received and one of the following conditions holds:
    - No accept-more-specific site EID prefix configuration exists for the EID instance and registrations for all the configured EID prefixes have been received.
    - Three registration periods have elapsed from the time that the first registration was received.
    - No registrations have been received and three registration periods have elapsed from the time that the LISP control plane restarted.

You can manage membership distribution on the Map-Server using the **show lisp site rloc members** command in the EXEC configuration mode.

How to Implement Data Plane Security, on page 6 provides procedural details.

# Decapsulation Filtering on (P)xTRs

The source RLOC decapsulation RLOC filtering feature is enabled on a (P)xTR through the **decapsulation filter rloc source** command. After the feature is enabled, the (P)xTR only allows the decapsulation of LISP data packets carrying a source RLOC that is allowed by the filter. When the feature is first enabled, if the filter is based on the auto-discovery of the EID instance membership from the Map-Servers then traffic will be dropped until a reliable transport connection is established with the Map-Servers and the membership is received.

**(P)xTR Membership Discovery**

A (P)xTR that is configured for data plane source RLOC filtering with membership auto-discovery for one or more EID instances through the **decapsulation filter rloc source members** configuration, attempt to establish a reliable transport session with each of the configured Map-Servers for those instances. A single reliable transport session is initiated with each Map-Server over which the membership for one or more EID instances is communicated. The auto-discovered membership lists is extended to form the source rloc filter through the **locator-set** option of the **decapsulation filter rloc source** command. The membership lists for an EID instance discovered through each of the Map-Servers are merged together with the contents of the configured locator-set and used to define the data plane source RLOC . The Map-Server only accepts incoming reliable transport connections from RLOC addresses that have first successfully registered an EID prefix. An xTR only attempts to establish a connection after it receives a Map-Notify acknowledging that its registration was successful. In order to request EID instance membership services for a specific instance ID at least one EID prefix for that instance must have been successfully registered.

Once the connection with a Map-Server is established the (P)xTR sends a Membership-Request message for each of the EID instances that have the Map-Server in their configuration. Received Membership-Add and Membership-Delete messages update the EID instance membership database on the (P)xTR.

To rebuild its EID instance membership database, the (P)xTR issues a Membership-Refresh-Request message as soon as the Map-Server indicates that it is willing to provide membership services through a Membership-ACK message. The (P)xTR maintains an epoch for each discovered membership entry. When a Membership-Refresh-Start message is received from a Map-Server, the (P)xTR increments the epoch it maintains for the Map-Server and EID instance combination, thus flagging the existing membership state as stale. Subsequent Membership-Add messages received during the refresh update the epoch of the corresponding entries. When the Membership-Refresh-End message is received, the (P)xTR sweeps the membership entries for the EID instance received from the Map-Server deleting the ones carrying an old epoch that have not been updated during the refresh.

**Filter Communication to Forwarding**

The LISP control plane uses the RIB opaque facility for communicating information through the RIB, all the way to all FIB instances as part of table distribution. Messages are defined to:

- Convey the filter enablement state on a per RLOC AF and EID instance granularity
- Convey RLOC filter entries

## TCP-based Reliable Transport Sessions

LISP uses TCP-based sessions between the xTRs and Map-Servers for EID instance membership distribution. The reliable transport session supports (using TCP port 4342) establishing of an active or passive session, with the xTR taking the active role and the Map-Server the passive role. Sessions are accepted only from valid RLOCs from the Map-Server side based on source RLOC filtering. The number of concurrent TCP connections that can be supported varies on a per OS and platform basis. Some security considerations that you must be take into account:

- The number of xTRs that a Map-Server can cater for is limited by the number of TCP sessions that a platform can establish and maintain. This will determine the number of VPN customers that a Map-Server can host. Horizontal scaling is achieved by dividing VPN customers between multiple Map-Servers.
- All the xTRs belonging to the same VPN must register with the same Map-Server.You cannot have VPNs with a larger number of xTRs than the Map-Server TCP session scale limit.
- Session authentication of the initial deliverable relies on the integrity of the RLOC network and only filters TCP sessions using the source address of packets.

For additional details on TCP-based reliable transport session such as Session Establishment, Reliable Transport Message Format, Keep-alive Message, Error Notification Message, see http:// tools.ietf.org /id/draft-kouvelas-lisp-reliable-transport-00.txt.

# How to Implement Data Plane Security

This section contains the following procedures:

## Enable Source RLOC-based Decapsulation Filtering

To configure an xTR or Proxy-xTR to download decapsulation filter lists for source validation when decapsulating LISP packets, use the **decapsulation filter source** command in the lisp configuration mode.

When an (P)ETR decapsulates LISP packets, this occurs without consideration of the LISP packet outer header source address. In networking environments where the source address can be trusted, it may be desired to consider the source address of the LISP packet prior to decapsulation. By configuring the decapsulation filter source command on (P)xTRs, a device will establish a TCP-based reliable transport session with its

Map-Server(s) and download and use filter list(s) when decapsulating LISP packets. Either or both of the **members** or **locator-set** keywords must be specified.

When the **members** keyword is specified the xTR will attempt to establish a reliable transport (TCP) sessions with the configured map-servers to automatically obtain the registered RLOC membership list. When a **locator-set** is specified the filtering will be performed against the locators that are configured within the locator-set. When both the locator-set and the "members" keyword are specified then the configured locators and the automatically discovered ones will be merged and the resulting list used to filter decapsulated packets.

**Note**

- A (P)xTR normally communicates with multiple Map-Servers. However, in the event that all reliable transport session goes down, any existing (possibly stale) filter list will remain in use during a small window of time (several minutes), during which time the (P)xTR tries to re-establish the session(s) with the MS and refresh its membership.
- If no filter list can be downloaded, or the existing list times out, packets will be dropped. (fail closed.)
- If the xTR changes RLOCs (via DHCP for example), as soon as the RLOC is changed, the registration with the Map-Server is updated and the new registered RLOC is pushed to all “members” of this IID/VPN (event-driven).

### Before you begin

Ensure that the following pre-requisites are met:

- On an xTR, the TCP-based reliable transport session is established only after the UDP-based (normal) Map-Registration process successfully completes.
- On a PxTR, since this device does not (normally) register with a Map-Server, a 'stub' (fake) Map-Registration configuration must be added to allow the establishment of the reliable transport session and the download of any filter lists. The Map-Server requires the PETR RLOC(s) to be included in a map-server rloc members modify-discovered add command to permit this session establishment.

## SUMMARY STEPS

1. **configure**
2. **router lisp**
3. **exit**
4. **locator-set** *name IP_address*
5. **eid-table** { **default** | [ **vrf** *vrf_name*]} **instance-id** *instance_id*
6. **address-family** { **ipv4** | **ipv6** } **unicast**
7. **etr map-server** *IP_address* { **key** [ **clear** | **encrypted**] **LINE** | **proxy-reply** }
8. **itr map-resolver** *map-resolver-address*
9. **map-cache** *destination-EID-prefix / prefix-length* { **action** { **drop** | **map-request** | **native-forward**} | **locator** *locator-address* **priority** *priority_value* | **weight** *weight_value*
10. **database-mapping** *EID-prefix/prefixlength locator* **locator-set site priority** *priority* **weight** *weight*
11. **exit**
12. **decapsulation filter rloc source** [ **locator-set** *locator_set_name* ][ **members** ]
13. **locator-table** *name* [ **default** | **vrf** *vrf_name*]

14. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **router lisp**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# router lisp` | Enables LISP for the specified routing instance, and places the router in Locator and ID Separation Protocol (LISP) configuration mode. |
| **Step 3** | **exit**<br>**Example:**<br>`RP/0/RP0/CPU0:routerRP/0/0/CPU0:ios(config-lisp-afi)#exit` | Returns the router to LISP configuration mode. |
| **Step 4** | **locator-set** *name IP_address*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-lisp)#locator-set loc_sh1_vrf1 202.1.0.1` | Configure a named locator set site, and specifies the RLOC IP address of Loopback or other Egress Tunnel Router (ETR) interfaces. |
| **Step 5** | **eid-table** { **default** \| [ **vrf** *vrf_name*]} **instance-id** *instance_id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-lisp)#eid-table default instance-id <IID-A>` | Selects the default (global) routing table or the specified VRF table for association with the configured instance ID. |
| **Step 6** | **address-family** { **ipv4** \| **ipv6** } **unicast**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-lisp-afi)# address-family ipv4 unicast` | Specifies the IPv4 or IPv6 address family, and enters address family configuration mode.<br>- This example specifies the unicast IPv4 address family. |
| **Step 7** | **etr map-server** *IP_address* { **key** [ **clear** \| **encrypted**] **LINE** \| **proxy-reply** }<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-lisp-afi)#etr map-server 204.1.0.1 key encrypted lisp` | Specifies the options related to the etr map-server (MS) such as locator and authentication key. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **itr map-resolver** *map-resolver-address*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-lisp-afi)#itr map-resolver 204.1.0.1` | Configures an IPv4 or IPv6 locator address of the LISP Map-Resolver to be used by the ITR Map-Requests for IPv4 EID-to-RLOC mapping resolution. |
| **Step 9** | **map-cache** *destination-EID-prefix / prefix-length* { **action** { **drop** \| **map-request** \| **native-forward**} \| **locator** *locator-address* **priority** *priority_value* \| **weight** *weight_value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-lisp-afi)#map-cache 12.2.0.0/24 map-request`<br>`RP/0/RP0/CPU0:router(config-lisp-afi)#map-cache 102.2.0.0/24 map-request`<br>`RP/0/RP0/CPU0:router(config-lisp-afi)#map-cache 103.2.0.0/24 map-request` | Configures a static IPv4 EID-to-RLOC or static IPv6 EID-to-RLOC mapping relationship and its associated traffic policy, or statically configure the packet handling behavior associated with a destination IPv4 EID-prefix or a destination IPv6 EID-prefix. |
| **Step 10** | **database-mapping** *EID-prefix/prefixlength locator* **locator-set site** **priority** *priority* **weight** *weight*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-lisp-afi)#database-mapping 11.2.0.0/24 201.1.0.1`<br>`priority 1 weight 100` | Configures an EID-to-RLOC mapping relationship and its associated traffic policy for this LISP site.<br>**Note** Repeat this step until all EID-to-RLOC mappings within this eid-table vrf and instance ID for the LISP site are configured. |
| **Step 11** | **exit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-lisp-afi)#exit` | Returns the router to LISP configuration mode. |
| **Step 12** | **decapsulation filter rloc source** [ **locator-set** *locator_set_name* ][ **members** ]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-lisp)#decapsulation filter rloc`<br>`source member locator-set loc_sh1_vrf1` | Enables the source RLOC based decapsulation filtering feature.<br>• The **members** keyword enables the establishment of a reliable transport (TCP) session with configured Map-Server(s), and the download of the decapsulation filer list maintained by the Map-Server(s)and the download of the decapsulation filer list maintained by the Map-Server(s)<br>• The **locator-set** keyword is used, the prefixes named in the locator-set are used, if included alone, or added to the (downloaded) dynamic list when used in conjunction with the member keyword. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 13** | **locator-table** *name* [ **default** \| **vrf** *vrf_name*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-lisp)#locator-table vrf 1` | Associate a virtual routing and forwarding (VRF) table through which the routing locator address space is reachable to a router Locator ID Separation Protocol (LISP) instantiation. |
| **Step 14** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

### Example

In this example, an xTR is configured to establish a reliable transport session with the Map-Server at 204.1.0.1, download the decapsulation filter list (in this case for IID 1002), and source-check all LISP-encapsulated packets using this filter list prior to decapsulation.

```
router lisp
 address-family ipv4 unicast
 !
 locator-set loc_sh1_vrf1
  202.1.0.1
  203.1.0.1
 !
 eid-table vrf sh1_vrf2 instance-id 1002
  address-family ipv4 unicast
   etr map-server 204.1.0.1 key encrypted lisp
   etr
   itr map-resolver 204.1.0.1
   itr
   map-cache 12.2.0.0/24 map-request
   map-cache 102.2.0.0/24 map-request
   map-cache 103.2.0.0/24 map-request
   database-mapping 11.2.0.0/24 201.1.0.1 priority 1 weight 100
   database-mapping 101.2.0.0/24 201.1.0.1 priority 1 weight 100
  !
  decapsulation filter rloc source member locator-set
     loc_sh1_vrf1
 !
 locator-table default
```

# Create, Maintain and Distribute Decapsulation Filter Lists

A Map-Server can be configured to dynamically create, maintain, and distribute decapsulation filter lists, on a per instance-ID basis, to appropriate LISP devices using the map-server rloc members distribute command in site configuration mode. When configured:

- The Map-Server allows the establishment of TCP-based LISP reliable transport sessions with appropriate xTRs
- The Map-Server creates/maintains lists (per-IID) of LISP site RLOCs (per-IID) based on RLOC addresses of registered LISP sites
- The Map-Server pushes/updates filters lists over the reliable transport mechanism to established devices

**Note**

- Data plane security is enabled by the use of the “map-server roc members distribute” command. The optional command “map-server rloc members modified-discovered [add | override] is used to append to or override the dynamically maintained RLOC filter list.
- This feature is used in conjunction with the decapsulation filter rloc source command, configured on (P)xTR devices which are performing the decapsulation

This example shows how you can configure the Map-Server to create reliable transport sessions with specific LISP sites, to dynamically create, maintain, and distribute decapsulation filter lists.

```
router lisp
 locator-set PxTR_set
  2001:DB8:E:F::2
  exit
 !
 eid-table vrf 1001 instance-id 1001
  map-server rloc members modify-discovered add locator-set PxTR_set
  exit
 !
---<skip>---
 !
 map-server rloc members distribute
!
```

# Add or Override Decapsulation Filter List

When a Map-Server is configured to dynamically create, maintain, and distribute a decapsulation filter list, the decapsulation filter list can be added to or overridden by using the map-server rloc members modify-discovered command in EID-table configuration mode. Uses may include:

- When a PxTR is included in the architecture, the PITR LISP-encapsulates packets to an ETR – and the ETR must therefore include the PITR RLOC in its decapsulation filter list. Since PITRs do not register with Map-Servers, their RLOCs are not automatically included in the decapsulation filter list and must be added via configuration using this command.
- A PETR can also be configured to filter upon decapsulation, but again, because a PETR does not register with a Map-Server, it needs a way to obtain the decapsulation filter list. The **add** form of this command includes the mechanisms to establish the reliable transport session with the Map-Server for obtaining the decapsulation filter list on the PETR.

- For diagnostic/troubleshooting reasons, it may be useful to (temporarily) override the entire decapsulation filter list.

**Note** The add function must be included in order for a PETR to “fake registers” with the Map-Server to obtain the decapsulation filter list. The inclusion of the PETR RLOC in this command allows the PETR to establish the reliable transport session.

In this example, the Map-Server is configured to create reliable transport sessions with specific LISP sites, to dynamically create, maintain, and distribute decapsulation filter lists. It has also been configured to modify the dynamically created filter list (comprised of registered site RLOC addresses) with the statically configured PxTR IPv6 RLOC address of 2001:db8:e:f::2.

```
router lisp
 locator-set PxTR_set
  2001:DB8:E:F::2
  exit
 !
 eid-table vrf 1001 instance-id 1001
  map-server rloc members modify-discovered add locator-set PxTR_set
  ipv4 route-export site-registration
  exit
 !
---<skip>---
 !
 map-server rloc members distribute
!
```

# Reset LISP TCP Reliable Transport Session

To reset the LISP TCP reliable transport session between an xTR and an MS use the **clear lisp vrf** command in the EXEC mode.

### SUMMARY STEPS

1. **clear lisp vrf** *VRF_name* **session** {*peer_address* | *

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **clear lisp vrf** *VRF_name* **session** {*peer_address* \| * <br>**Example:**<br>`RP/0/0/CPU0:ios#clear lisp vrf test session *` | When a peer-address is specified the TCP connection to that peer will be cleared. When the “*” option is specified all LISP reliable transport sessions will be cleared. |

# Verify Data Plane Security Configurations

Perform this task to verify data plane security configurations:

### SUMMARY STEPS

1. **show lisp session**

2. **show lisp site** [ **instance-id** *EID instance-ID* ] **rloc members** [ **registrations** [ *rloc-addr*]]
3. **show lisp vrf** *vrf_name* **session** [*peer_address*]
4. **show lisp decapsulation filter**
5. **show cef vrf** [*locator-vrf*] *address_family* **lisp decapsulation** [ **instance-id** *EID-instance-ID* ] **detail location** *RLOC-facing LC*
6. **show controllers np struct LISP-INSTANCE-HASH detail all-entries** [ **all** | *np* ] **location** *RLOC-facing LC*

## DETAILED STEPS

<table>
<tr><th></th><th>Command or Action</th><th>Purpose</th></tr>
<tr><td><b>Step 1</b></td><td>

**show lisp session**

**Example:**

```
R11-MSMR#show lisp session

Sessions for VRF default, total: 8, established:
7
Peer                    State      Up/Down
In/Out    Users
2001:DB8:A:1::2         Up         00:04:13
 2/7      2
2001:DB8:A:2::2         Up         00:04:13
 2/7      2
2001:DB8:A:3::2         Up         00:03:53
 2/7      2
2001:DB8:B:1::2         Up         00:04:04
 2/6      2
2001:DB8:B:2::2         Init       never
 0/0      1
2001:DB8:C:1::2         Up         00:03:55
 2/6      2
2001:DB8:C:2::2         Up         00:03:54
 2/6      2
2001:DB8:E:F::2         Up         00:04:04
 6/19     4
R11-MSMR#
```

</td><td>On a LISP device, to display a current list of reliable transport (TCP) sessions, use the <b>show lisp session</b> command in the EXEC configuration mode. In this example, reliable transport LISP sessions are displayed on the Map-Server. In the output, seven sessions are established and one session is in the Init state (the decapsulation filter rloc source member command had not been applied at that site; the session was not established).</td></tr>
<tr><td><b>Step 2</b></td><td>

**show lisp site** [ **instance-id** *EID instance-ID* ] **rloc members** [ **registrations** [ *rloc-addr*]]

**Example:**

```
R114-MSMR#show lisp site rloc members
LISP RLOC membership for EID table default (IID
0), 5 entries

RLOC                   Origin                 Valid
1.2.3.4                 config                  Yes
10.0.1.2                registration            Yes
10.0.2.2                config & registration  Yes
13:12::1                config                  Yes
2001:DB8:2:3::2         registration            Yes
```

</td><td>To display the gleaned and configured EID instance membership use the <b>show lisp site</b> command in the EXEC configuration mode. The 'origin' column in the output shows whether the RLOC member has been manually configured, automatically gleaned from received registrations, or both. The 'valid' column shows whether the RLOC is a valid member that is distributed to (P)xTRs. A listed RLOC may not be valid if it is gleaned from registrations, but the 'override' option is used in the 'modify-discovered' configuration and the specified locator-set does not include the RLOC. When the optional 'registrations' keyword is specified the command displays the list of registrations contributing to a membership entry.</td></tr>
<tr><td><b>Step 3</b></td><td>

**show lisp vrf** *vrf_name* **session** [*peer_address*]

**Example:**

</td><td></td></tr>
</table>

**Command or Action** | **Purpose**

```
On xTR:
RP/0/RSP1/CPU0:VKG-1#sh lisp vrf default session

Sessions for VRF default, total: 1, established:
1
Peer                    State      Up/Down
 In/Out    Users
204.1.0.1               Up         06:49:05
  0/1      1
RP/0/RSP1/CPU0:VKG-1#

On MSMR:
sh lisp vrf default session

Sessions for VRF default, total: 2, established:
2
Peer                     State      Up/Down
  In/Out    Users
201.1.0.1                Up         06:48:49
   1/0      0
202.1.0.1                Up         06:48:36
   2/0      0
RP/0/RSP0/CPU0:VKG-4#
```

**Step 4**

**show lisp decapsulation filter**

**Example:**

```
RP/0/RSP0/CPU0:lisp9-a9k-1#show lisp eid-table se2
 decapsulation filter
LISP decapsulation filter for EID table vrf se2
(IID 16777212), 5 entries
Source RLOC                               Added by
22:22::10                                 MS
190::190
33:33::20                                 MS
190::190
88:88::30                                 MS
190::190
99:99::30                          MS 190::190
110:110::40                               MS
190::190
RP/0/RSP0/CPU0:lisp9-a9k-1#
```

On a LISP device, to display decapsulation filter-related data for the selected source RLOCs, use the **show lisp decapsulation filter** command in the EXEC configuration mode. In this example, decapsulation filter information is displayed on an (P)xTR for Instance-ID (IID 16777212). In this output, five source RLOC addresses are defined, and all members of this list were defined within the list provided by the reliable transport session with the Map-Server.

**Step 5**

**show cef vrf** [*locator-vrf*] *address_family* **lisp decapsulation** [ **instance-id** *EID-instance-ID* ] **detail location** *RLOC-facing LC*

**Example:**

```
RP/0/RSP0/CPU0:lisp9-a9k-1#show cef ipv6 lisp
decapsulation
instance-id 16777212 loc 0/0/cpu0

Number of EID tables handling LISP payload received
 in this table: 3

Transport LISP ipv6 packets received in VRF:
default, Instance ID: 16777212
        Payload IPv4 is         : decapsulated
        Payload IPv6 is         : decapsulated
        Payload switched in VRF : se2
```

In this example, decapsulation filter summary information is displayed on an (P)xTR.

**Command or Action** | **Purpose**

```
(0xe000001d/0xe080001d)
        H/W driver signalled    : active
        Binding in retry        : no

Source RLOC Prefix Filter       : enabled
        Lookup statistics (s/w) :
         Misses                 : 8
         Matches (historic)     : 0
        H/W driver signalled    : active
        Binding in retry        : no
        Platform space          : allocated

Len  Prefix           Action      Matches
Attributes

128  22:22::10        accept          0 h/w
[active, plt space]
128  33:33::20        accept          0 h/w
[active, plt space]
128  88:88::30        accept          0 h/w
[active, plt space]
128 99:99::30  accept          0 h/w [active, plt
 space]
128  110:110::40      accept          0 h/w
[active, plt space]
```

**Step 6**

**show controllers np struct LISP-INSTANCE-HASH detail all-entries** [ **all** | *np* ] **location** *RLOC-facing LC*

### Example:

```
RP/0/RSP0/CPU0:lisp9-a9k-1#show controllers np
struct LISP-INSTANCE-HASH
detail all-entries np0 location 0/0/cpu0

                Node: 0/0/CPU0:
-------------------------------------------------


NP: 0
Struct 114: LISP_INSTANCE_HASH_STR (maps to uCode
 Str=79)
Struct is a LOGICAL entity inside a shared PHYSICAL
 resource
Reserved Entries: Logical          0, Physical
       0
Used Entries:     Logical         79, Physical
      79
Max Entries:      Logical      86016, Physical
   86016
Entries Shown:    Logical         79

-------------------------------------------------

 Entry 1: >>
    Key: 4affffe 00000099 00990000 00000000
00000000 0030    Size: 22
   Mask: ffffffff ffffffff ffffffff ffffffff
ffffffff ffff    Size: 22
 Result: 51000000 1e000000    Size: 8
 Entry 2: >>
    Key: 4976adf1 1c005858 0e1e0000 00000000
```

| | Command or Action | Purpose |
|---|---|---|

```
00000000 0000    Size: 22
   Mask: ffffffff ffffffff ffffffff ffffffff
ffffffff ffff    Size: 22
 Result: 51000000 1c000000    Size: 8
 Entry 3: >>
    Key: 4976adf1 1c006e6e 0e280000 00000000
00000000 0000    Size: 22
   Mask: ffffffff ffffffff ffffffff ffffffff
ffffffff ffff    Size: 22
 Result: 51000000 1c000000    Size: 8
 Entry 4: >>
    Key: 41000000 20002121 0b140000 00000000
00000000 0000    Size: 22
   Mask: ffffffff ffffffff ffffffff ffffffff
ffffffff ffff    Size: 22
 Result: 51000000 1f000000    Size: 8
 Entry 5: >>
    Key: 4afffffc 00000099 00990000 00000000
00000000 0030    Size: 22
   Mask: ffffffff ffffffff ffffffff ffffffff
ffffffff ffff    Size: 22
 Result: 51000000 1d000000    Size: 8
 Entry 6: >>
    Key: 4a0003e8 19000033 00330003 00000000
00000000 0020    Size: 22
   Mask: ffffffff ffffffff ffffffff ffffffff
ffffffff ffff    Size: 22
 Result: 51000000 19000000    Size: 8
 Entry 7: >>
 <Snipped>
End NP Show Structure Display
```

# Additional References

The following sections provide references related to implementing LISP data plane security:

### Related Documents

| Related Topic | Document Title |
|---|---|
| LISP commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *Routing Command Reference for Cisco CRS Routers* |
| LISP Configuration Guide, Cisco IOS Release | IP Routing: LISP Configuration Guide, Cisco IOS Release 15M&T |

### Standards

| Standards | Title |
|---|---|
| draft-kouvelas-lisp-reliable-transport-00.txt | *LISP Reliable Transport* by C. Cassar, I. Kouvelas and D. Lewis |

| Standards | Title |
|---|---|
| RFC 6830 | *Locator/ID Separation Protocol (LISP)* |
| RFC 6832 | *Interworking between Locator/ID Separation Protocol (LISP) and Non-LISP Sites* |
| RFC 6833 | *Locator/ID Separation Protocol (LISP) Map-Server Interface* |

### Technical Assistance

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