Implementing URPF

This section describes the implementation of URPF.

Understanding uRPF

Table 1: Feature History Table

Feature Name	Release Information	
uRPF in Loose Mode	Release 7.3.15	When the source IP address of an incoming packet is not present in the Forwarding Information Base (FIB), the router considers it as an invalid packet and drops it. Use the **allow-default** keyword of **ipv4/ipv6 verify unicast source reachable-via** command and configure the default route for the interface so that the router does not drop a packet even when the source IP address is not present in the FIB. The command **ipv4/ipv6 verify unicast source reachable-via**is introduced.

It has become commonplace practice for hackers planning a Denial of Service (DoS) attack to use forged IP addresses (the practice is known as IP address spoofing). Hackers constantly change the source IP address to avoid detection by service providers. DoS uses more than one forged IP address from thousands of hosts that are infected with malware to flood a device. Therefore, it is complicated to identify and defeat the malware attack.

The uRPF is a mechanism for validating the source IP address of packets that are received on a router. A router that is configured with uRPF performs a reverse path lookup in the FIB table to validate the presence of the source IP address. If the FIB table lists the source IP address, then it indicates that the source is reachable and valid. If the FIB table does not list the source IP address, the router treats the packet as malicious and drops it.

The router supports uRPF in two modes:

- **uRPF in Loose Mode**: In uRPF loose mode, the router checks if it has a matching entry for the source IP address in the FIB and does not drop the legitimate regardless of interfaces the source address is learned on
- **uRPF in Strict Mode**: In uRPF strict mode, the router check if the interface receiving traffic packets is the same as the interface to reach the incoming packet's source address, the router considers such traffic packets legitimate and processes them. If not, the router drops it. The router supports uRPF in Strict Mode since IOS XR Release 7.9.1

Configuring uRPF in Loose Mode

When you configure uRPF in loose mode, the router checks if it has a matching entry for the source IP address in the FIB and does not drop the legitimate traffic that uses an alternate interface to reach the router. The uRPF in loose mode is useful in multihomed, service provider, edge networks.

Configuration

Use the following configuration to configure uRPF in loose mode on the router.

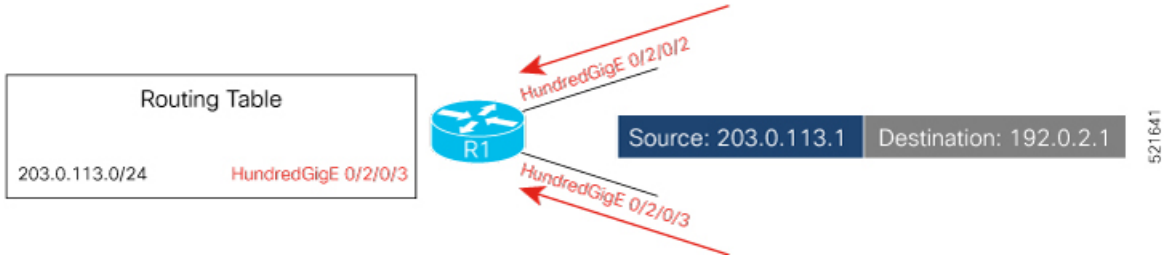
Note

- You can configure uRPF in loose mode on router interfaces, subinterfaces, bundle interfaces, and bundle subinterfaces
- Configure both IPv4 and IPv6 commands (as described in this section) for uRPF to work.

```
Router(config)# interface HundredGigE 0/2/0/2
Router(config-if)# ipv4 verify unicast source reachable-via any
Router(config-if)# ipv6 verify unicast source reachable-via any
Router(config-if)# commit
```

In the following figure, in router R1, the FIB table lists HundredGigE0/2/0/3 as the egress interface for the network 203.0.113.0/24. The ingress interface is HundredGigE 0/2/0/2. R1 receives packets with source IP address as 203.1.113.1 from both the interfaces, HundredGigE0/2/0/2 and HundredGigE0/2/0/3. When you configure uRPF in loose mode on the ingress interface, the router checks if the source address has a matching entry in the FIB table. The router does not drop the packet even if the ingress interface is not listed in the FIB tableas the outgoing interface for that prefix.

Figure 1: uRPF in Loose Mode



Running Configuration

To verify that the number of packets dropped due to uRPF configuration, you can use the **show cef drops**:

```
Router(config-if)# show cef drops
Node: 0/0/CPU0
  Unresolved drops     packets :              0
  Unsupported drops    packets :              0
  Null0 drops          packets :              0
  No route drops       packets :              2
  No Adjacency drops   packets :              0
  Checksum error drops packets :              0
  RPF drops            packets :              1911
  RPF suppressed drops packets :              0
  RP destined drops    packets :              0
  Discard drops        packets :              0
  GRE lookup drops     packets :              0
  GRE processing drops packets :              0
  LISP punt drops      packets :              0
  LISP encap err drops packets :              0
  LISP decap err drops packets :              0
Node: 0/RP0/CPU0
  Unresolved drops     packets :              0
  Unsupported drops    packets :              0
  Null0 drops          packets :              0
  No route drops       packets :              2
  No Adjacency drops   packets :              0
  Checksum error drops packets :              0
  RPF drops            packets :              1503
```

You have successfully configured uRPF in loose mode on the router.

Configuring Default Route for uRPF in Loose Mode

When you configure uRPF in loose mode, the source address of the packet must appear in the FIB for the verification process. However, you can use the **allow-default** option to use the default route in the source IP address verification process.

- When you do not configure the **allow-default** option, the router drops the packet that does not have its source address listed in the FIB table.
- When you configure the **allow-default** option, you must configure the default route for the interface. Otherwise, the router drops the packet.
- When you configure uRPF in loose mode with **allow-default** on any VRF interface, then it is applicable to all the interfaces in that VRF of the router.

Use the following configuration to configure uRPF in loose mode on the router along with the default address.

```
Router(config)# interface HundredGigE 0/2/0/2
Router(config-if)# ipv4 verify unicast source reachable-via any allow-default
Router(config-if)# ipv6 verify unicast source reachable-via any allow-default
Router(config-if)# commit
```

Configuring uRPF in Strict Mode

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
uRPF in Strict Mode	Release 7.9.1	You can protect the router against DoS attacks with spoofed source IP addresses by enabling the Strict mode in uRPF. When this feature is enabled, the router accepts the incoming packet only if the source IP address of the packet is present in its routing table and if the source IP address of the input packet is reachable via the interface on which the packet has been received. If not, the router drops the packet.In earlier releases IOS XR supports only loose mode uRPF. This feature introduces the **hw-module profile cef unipath-surpf** command. This feature modifies the **ipv4/ipv6 verify unicast source reachable-via** command.

When you enable uRPF in strict mode, the router checks for the source address of the packet in the Forwarding Information Base (FIB). If the router receives the incoming packet on the same interface that the router would use to forward the traffic to the source of the packet, the packet passes the check and is further processed; otherwise, it is dropped. uRPF in strict mode should only be applied where there's natural or configured symmetry. Because internal interfaces are likely to have routing asymmetry. That is, multiple routes to the source of a packet, uRPF in strict mode shouldn't be implemented on interfaces that are internal to the network.

Usage Guidelines

- You can configure uRPF in strict mode on router interfaces, subinterfaces, bundle interfaces, and bundle subinterfaces.
- The tunnel and BVI interfaces don't support uRPF strict mode.
- Configure both IPv4 and IPv6 traffic types for uRPF to work.
- uRPF Strict mode is disabled in the router, by default.
- The uRPF in strict mode supports the **allow default** option. When the **allow default** option is enabled with the uRPFin strict mode, the packet is processed further only if it arrived through the default routes.

Prerequisites

Configure both IPv4 and IPv6 traffic types for uRPF to work.

Configuration

Use the following configuration to configure uRPF in strict mode on the router:

```
Router(config)# hw-module profile cef unipath-surpf enable
Router(config)# interface HundredGigE 0/2/0/2
Router(config-if)# ipv4 address 10.0.0.1 255.255.255.0
Router(config-if)# ipv4 verify unicast source reachable-via rx
Router(config-if)# ipv6 address 2001::1/64
Router(config-if)# ipv6 verify unicast source reachable-via rx
```

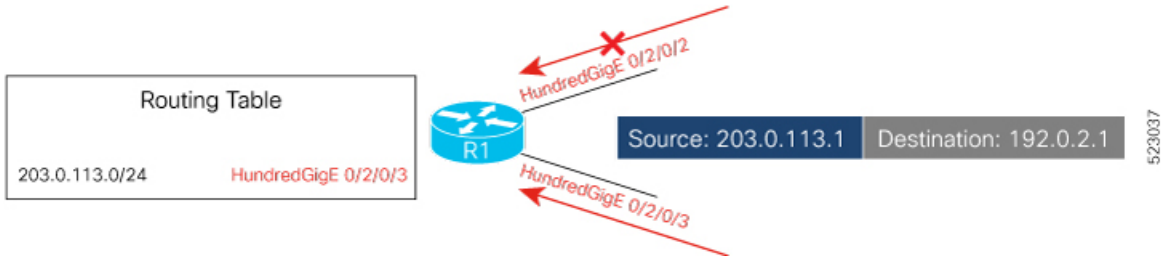
```
Router(config-if)# commit
Router(config-if)# exit
Router(config)# reload
```

Note You must reload the router after executing the **hw-module profile cef unipath-surpf** command.

In the following figure, in router R1, the FIB table lists HundredGigE0/2/0/3 as the egress interface for the network 203.0.113.0/24. R1 receives packets with source IP address as 203.1.113.1 from two different interfaces, HundredGigE0/2/0/2 and HundredGigE0/2/0/3. R1 accepts the packet coming from HundredGigE0/2/0/3 as the route to reach the source is 203.1.113.1 according to the FIB table. But the incoming packet via HundredGigE0/2/0/2 is dropped as the entries in the FIB table doesn't specifies HundredGigE0/2/0/2 as the interface to reach 203.1.113.1.

Note In the above example, the **hw-module profile cef unipath-surpf** configuration ensures the router R1 drops incoming packets via HundredGigE0/2/0/2, as according to the FIB table, the only interface to reach 203.0.113.0/24 is HundredGigE0/2/0/3. If there are multiple egress interfaces in router R1 for the 203.0.113.0/24 network, they will ensure to check all of these entries before dropping the packet.

Figure 2: uRPF in Strict Mode



Running Configuration

Confirm your configuration as shown:

```
Router(config-if)# show running-config
...
!
interface HundredGigE 0/2/0/2
 ipv4 address 10.0.0.1 255.255.255.0
 ipv4 verify unicast source reachable-via rx
ipv6 address 2001::1/64
 ipv6 verify unicast source reachable-via rx
!
```

Running Configuration

To verify that the number of packets dropped due to uRPF configuration, you can use the **show cef drops**:

```
Router(config-if)# show cef drops
Node: 0/0/CPU0
  Unresolved drops      packets :               0
  Unsupported drops     packets :               0
  Null0 drops           packets :               0
  No route drops        packets :               2
```

```
  No Adjacency drops   packets :              0
  Checksum error drops packets :              0
  RPF drops            packets :              1911
  RPF suppressed drops packets :              0
  RP destined drops    packets :              0
  Discard drops        packets :              0
  GRE lookup drops     packets :              0
  GRE processing drops packets :              0
  LISP punt drops      packets :              0
  LISP encap err drops packets :              0
  LISP decap err drops packets :              0
Node: 0/RP0/CPU0
  Unresolved drops     packets :              0
  Unsupported drops    packets :              0
  Null0 drops          packets :              0
  No route drops       packets :              2
  No Adjacency drops   packets :              0
  Checksum error drops packets :              0
  RPF drops            packets :              1503
```