Cisco Prime Access Registrar 9.2 Release Notes

Cisco Prime Access Registrar (Prime Access Registrar) is a high performance, carrier class, 3GPP-compliant, 64-bit RADIUS/Diameter solution that provides scalable, flexible, intelligent authentication, authorization, and accounting (AAA) services.

Prime Access Registrar comprises a RADIUS/Diameter server designed from the ground up for performance, scalability, and extensibility for deployment in complex service provider environments including integration with external data stores and systems. Session and resource management tools track user sessions and allocate dynamic resources to support new subscriber service introductions.

Note Prime Access Registrar can be used with Red Hat Enterprise Linux (RHEL) 7.x and 8.2 or CentOS 7.x operating system. Also, Prime Access Registrar is qualified with VMware ESXi 7.0 Update 1c.

Note Support for EAP-FAST has been deprecated from Prime Access Registrar release 9.2.

Contents

This release note contains the following sections:

System Requirements

This section describes the system requirements to install and use the Prime Access Registrar software.

Cisco Systems, Inc.
www.cisco.com

Table 1 lists the system requirements for Prime Access Registrar 9.2.

Table 1 Minimum Hardware and Software Requirements for Prime Access Registrar Server

OS Version	RHEL 7.x and 8.2 CentOS 7.x **Note** Prime Access Registrar supports OpenStack Stein and Victoria versions. You must have the 64-bit rpm files for the relevant RHEL versions while installing Prime Access Registrar. For the list of required rpms for the relevant OS versions, see Required 64-bit rpms for Relevant RHEL OS Versions, page 2.
Model	X86
CPU Type	Intel Xeon CPU 2.30 GHz
Processors	4
CPU Speed	2.30 GHz
Memory (RAM)	8 GB
Swap Space	10 GB
Disk Space	1*146 GB

Prime Access Registrar supports JDK versions 1.8.x and 11.x. Also, Prime Access Registrar is qualified with VMware ESXi 7.0 Update 1c.

Note These are the minimum system requirements to have Prime Access Registrar up and running. This may vary based on the deployments. Please contact your BU team to know the specific system requirements for your deployment.

Required 64-bit rpms for Relevant RHEL OS Versions

rpm	RHEL OS Version 7.x	RHEL OS Version 8.x
brotli	No	Yes
c-ares	Yes	Yes
cyrus-sasl-lib	Yes	Yes
gamin	Yes	Yes
glibc	Yes	Yes
gdome2	Yes	Yes
glib	Yes	Yes
glib2	Yes	Yes
json-c	Yes	Yes
keyutils-libs	Yes	Yes
krb5-libs	Yes	Yes

rpm	RHEL OS Version 7.x	RHEL OS Version 8.x
libbson	No	Yes
libcom_err	Yes	Yes
libcurl	Yes	Yes
libicu	Yes	Yes
libidn2	No	Yes
libgcc	Yes	Yes
libmongocrypt	No	Yes
libnghttp2	No	Yes
libnsl	No	Yes
libpsl	No	Yes
libselinux	Yes	Yes
libssh	No	Yes
libstdc++	Yes	Yes
libtool-ltdl	Yes	Yes
libunistring	No	Yes
libxcrypt	No	Yes
libxml2	Yes	Yes
libzstd	No	Yes
lksctp-tools	Yes	Yes
mongo-c-driver-libs	No	Yes
ncurses-libs	Yes	Yes
nss-softokn-freebl	Yes	Yes
nss-util	Yes	Yes
nspr	Yes	Yes
nss	Yes	Yes
openldap	No	Yes
openssl-libs	Yes	Yes
pcre	Yes	Yes
pcre2	No	Yes
pcre-cpp	No	Yes
snappy	No	Yes
sqlite-libs	No	Yes
tcl	No	Yes
unixODBC	No	Yes
xz-libs	No	Yes
zlib	Yes	Yes

Co-Existence With Other Network Management Applications

To achieve optimal performance, Prime Access Registrar should be the only application running on a given server. In certain cases, when you choose to run collaborative applications such as a SNMP agent, you must configure Prime Access Registrar to avoid UDP port conflicts. The most common conflicts occur when other applications also use ports 2785 and 2786. For more information on SNMP configuration, see the "Configuring SNMP" section in the "Configuring Cisco Prime Access Registrar" chapter of the *Cisco Prime Access Registrar 9.2 Administrator Guide*.

New and Enhanced Features in Cisco Prime Access Registrar 9.2

Cisco Prime Access Registrar 9.2 provides the following features:

Configuring LDAP Remote Server over SSL

Prime Access Registrar allows you to configure Lightweight Directory Access Protocol (LDAP) remote server over Secure Sockets Layer (SSL) protocol. For this, you must do the following under ***/Radius/RemoteServers/LDAP***:

- Set the **UseSSL** parameter to **TRUE.**
- Configure the port value accordingly. Default SSL port value is 636.

Till release 9.1.x, Prime Access Registrar supports LDAP configuration with Transport Layer Security (TLS) protocol up to TLSv1.0. From release 9.2 onwards, Prime Access Registrar supports TLSv1.1, TLSv1.2, and TLSv1.3 (Support for SSL3 and TLSv1.0 versions is deprecated).

The following parameters are added under ***/Radius/Advanced/*** to support this feature:

- CertificateDBPath—Path that contains valid client certificates for the LDAP server. Any changes to this parameter requires a restart of Prime Access Registrar.
- LDAPTLSVersion—Supported TLS versions for the LDAP server, which could be TLSv1.1, TLSv1.2, or TLSv1.3. Default is TLSv1.2. Any changes to this parameter requires at least a reload of Prime Access Registrar.

The following is a sample CLI of the LDAP remote server configuration over SSL:

```
[ /Radius/RemoteServers/ldap ]
    Name = ldap
    Description =
    Protocol = ldap
    Port = 636
    ReactivateTimerInterval = 5000
    Timeout = 15
```

```
HostName = 10.197.66.84
BindName = "cn=Directory Manager"
BindPassword = <encrypted>
UseSSL = TRUE
UseBinaryPasswordComparison = FALSE
EnableKeepAlive = FALSE
SearchPath~ = dc=example,dc=com
Filter~ = (uid=%s)
UserPasswordAttribute = employeeNumber
LimitOutstandingRequests = FALSE
MaxOutstandingRequests = 0
MaxReferrals = 0
ReferralAttribute =
ReferralFilter =
PasswordEncryptionStyle = Dynamic
EscapeSpecialCharInUserName = FALSE
DNSLookupAndLDAPRebindInterval =
DataSourceConnections = 1
SearchScope = SubTree
UseBindBasedAuthentication = FALSE
LDAPToRadiusMappings/
LDAPToEnvironmentMappings/
LDAPToCheckItemMappings/
```

Qualification with Latest Oracle Servers

Prime Access Registrar is qualified with the latest oracle servers 18c and 19c. This requires a compatible client to be installed or placed as per oracle recommendations. All oracle client library files must be placed under $ORACLE_HOME/lib.

Qualification with Latest MySQL Servers

Prime Access Registrar is qualified with the latest MySQL versions 8.0.19, 8.0.23, and 8.0.24. This requires compatible mysql-connector-odbc and mysql-community-client-plugins to be installed or placed as per MySQL recommendations. Supported mysql-connector-odbc versions are 8.0.19, 8.0.22, and 8.0.23.

Support for Secure ODBC Connection

Prime Access Registrar is enhanced to support the secure Open Database Connectivity (ODBC) connections with MySQL server using SSL. The following new attributes are added under ***/Radius/Advanced/ODBCDataSources*** in aregcmd to support this feature:

- SSLSecureTransport—Set this to TRUE to configure the MySQL server to connect over SSL.
- SSLCA—Path to a local file that contains a list of trusted Certificate Authorities.
- SSLCERT—Path to a local file that contains a list of trusted SSL client certificates.
- SSLKEY—Path to a local file that contains a list of trusted SSL client keys.

Following is a sample CLI configuration with the new parameters added for ODBC data sources:

```
[ //localhost/Radius/Advanced/ODBCDataSources/mysql ]
    Name = mysql
    Description =
```

```
    Type = myodbc
    Driver = /usr/lib64/libmyodbc8w.so
    UserID = root
    Password = <encrypted>
    DataBase = mysql
    Server = 10.197.95.182
    Port = 3306
    SSLSecureTransport = TRUE
    SSLCA = /etc/certs/ca.pem
    SSLCERT = /etc/certs/client-cert.pem
    SSLKEY = /etc/certs/client-key.pem
```

Following example shows a sample CLI configuration of the secure ODBC remote server:

```
[ /Radius/RemoteServers/mysql ]
    Name = mysql
    Description =
    Protocol = odbc
    ReactivateTimerInterval = 5000
    Timeout = 15
    DataSourceConnections = 8
    ODBCDataSource = mysql
    SNMPTrapIP =
    SNMPTrapPort = 1521
    KeepAliveTimerInterval = 0
    SQLDefinition/
        UserPasswordAttribute = password
        SQLStatements/
            Entries 1 to 1 from 1 total entries
            Current filter: <all>

            sql/
                Name = sql
                Description =
                Type = query
                SQL = "select password from artest where username = ?"
                ExecutionSequenceNumber = 1
                MarkerList = UserName/SQL_CHAR
    ODBCToRadiusMappings/
    ODBCToEnvironmentMappings/
    ODBCToCheckItemMappings/
```

Support for LDAP Multi-Value Attributes Mapping

LDAP attributes mapping support has been enhanced to accommodate multiple values to get mapped with the information fetched from LDAP. The LDAP query returns multi-value attributes in LDAP Authentication services. These will get mapped to corresponding attributes based on the LDAPToEnvironmentMappings. E.g. the parameter **Data** under LDAPToEnvironmentMappings is mapped to two values **NAS-Identifier** and **Reply-Message** using a supported delimiter configured using the **LDAPMultiValDelimiter** parameter under ***/Radius/Advanced***. The default delimiter is comma (,).

Following example shows a sample CLI of the LDAP configuration:

```
[ //localhost/Radius/RemoteServers/ldap ]
    Name = ldap
    Description =
    Protocol = ldap
    Port = 389
    ReactivateTimerInterval = 300000
    Timeout = 15
```

```
    HostName = 10.197.95.182
    BindName = "cn=Directory Manager"
    BindPassword = <encrypted>
    UseSSL = FALSE
    UseBinaryPasswordComparison = FALSE
    EnableKeepAlive = FALSE
    SearchPath~ = dc=examplear,dc=com
    Filter~ = (uid=%s)
    UserPasswordAttribute = employeeNumber
    LimitOutstandingRequests = FALSE
    MaxOutstandingRequests = 0
    MaxReferrals = 0
    ReferralAttribute =
    ReferralFilter =
    PasswordEncryptionStyle = Dynamic
    EscapeSpecialCharInUserName = FALSE
    DNSLookupAndLDAPRebindInterval =
    DataSourceConnections = 8
    SearchScope = SubTree
    UseBindBasedAuthentication = FALSE
    LDAPToRadiusMappings/
    LDAPToEnvironmentMappings/
        employeeNumber = eno
        Data = NAS-Identifier, Reply-Message
    LDAPToCheckItemMappings/
```

The LDAP data is mapped to environment dictionary variables and we need to put them in the response dictionary if they are required to be sent in the response packet.

A sample script is shown below:

```
proc Map {request response environ} {
             $response put "NAS-Identifier" [ $environ get "NAS-Identifier" ]
             $response put "Reply-Message" [ $environ get "Reply-Message" ]
}
```

Following is a sample CLI with the **LDAPMultiValDelimiter** parameter configuration:

```
[ //localhost/Radius/Advanced ]
    LogServerActivity = FALSE
    TLSv1Enabled = TRUE
    MaximumNumberOfRadiusPackets = 8192
    UDPPacketSize = 4096
    SocketWaitTime = 3
    NumberOfRemoteUDPServerSockets = 4
    NumberOfRadiusIdentifiersPerSocket = 256
    PerPacketHeapSize = 6500
    RequireNASsBehindProxyBeInClientList = FALSE
    AAAFileServiceSyncInterval = 75
    SessionBackingStoreSyncInterval = 100
    BackingStoreDiscThreshold = "5 Gigabyte"
    SessionBackingStorePruneInterval = "6 Hours"
    PacketBackingStorePruneInterval = "6 Hours"
    RemoteLDAPServerThreadTimerInterval = 10
    RemoteSigtranServerThreadTimerInterval = 10
    InitialBackgroundTimerSleepTime = 5
    MinimumSocketBufferSize = 65536
    CertificateDBPath = /opt/sslv3_certs
    LDAPTLSVersion = TLSv1.3
    LogFileSize = "1 Megabyte"
    LogFileCount = 2
    TraceFileSize = "1 Gigabyte"
    TraceFileCount = 2
    MemoryLimitForRadiusProcess = "3584 Megabyte"
```

```
UseAdvancedDuplicateDetection = FALSE
EnableDNAAA = FALSE
AdvancedDuplicateDetectionMemoryInterval = 10000
InitialSessionBufferSize = 0
DetectOutOfOrderAccountingPackets = FALSE
DefaultReturnedSubnetSizeIfNoMatch = BIGGER
ClasspathForJavaExtensions =
JavaVMOptions =
MaximumODBCResultSize = 256
ARIsCaseInsensitive = TRUE
RemoteRadiusServerInterface =
ODBCEnvironmentMultiValueDelimiter =
PacketBackingStoreSyncInterval = 75
ListenForDynamicAuthorizationRequests = FALSE
MaximumNumberOfXMLPackets = 1024
XMLUDPPacketSize = 4096
RollingEncryptionKeyChangePeriod = "1 week"
SessionPurgeInterval =
StaleOCSRemovalTimerForDOIC =
EapBadMessagePolicy = SilentDiscard
StaleSessionTimeout = "1 Hour"
MaximumOutstandingRequests = 0
MaximumIncomingRequestRate = 0
HideSharedSecretAndPrivateKeys = TRUE
DefaultRadiusSharedSecret =
ServerStatusSharedSecret = <encrypted>
EnableLocationCapability = FALSE
LogTPSActivity = TRUE
TPSLogFileCount = 15
TPSLogFilenamePrefix = tps
TPSSamplingPeriodInSecs = 15
LogSessionActivity = TRUE
EnableLengthFlag = FALSE
SessionLogFileCount = 15
SessionLogFilenamePrefix = sm
SessionSamplingPeriodInSecs = 30
LogIPActivity = FALSE
IPLogFileCount = 15
IPLogFilenamePrefix = ip
IPSamplingPeriodInSecs = 30
FlushDiskInBackground = FALSE
AdditionalNativeOracleErrors =
SendOpCodeInISDResponse = FALSE
EnableRoutingContextInM3UA = FALSE
EnableStickySessionCount = TRUE
ServerMonitorAltApproach = FALSE
EnableSIGTRANStackLogs = TRUE
SIGTRANStackLogFileSize = "100 Megabyte"
SIGTRANLogFileCount = 10
StickySessionCountInterval = 60000
StickySessionSyncInterval = 500
ReserveRADIUSPacketPool = 0
UserLogDelimiter = |
LDAPMultiValDelimiter = ,
DiameterStaleSessionPurgeTime = 00:00:00
DiameterStaleSessionPurgeFrequency =
UISessionTimeoutInMins = 0
DiameterStaleConnectionDeletionTimeOut = 300000
DiameterSessionRestorationPurgeTime = 02:00:00
IsMaster = FALSE
DisplayUserForFailedLogin = FALSE
EnableDuplicateSessionIdDetection = TRUE
ReservationFailed = FALSE
IPDataBackingStoreSyncInterval = 75
```

```
    IPDataBackingStorePruneInterval = "30 minutes"
    IPDataBackingStoreDiscThreshold = "1 Gigabyte"
    IPDataPurgeInterval = "30 Minutes"
    IPDocumentTimeOut = "2 Minutes"
    Ports/
    Interfaces/
    ReplyMessages/
    Attribute Dictionary/
    SNMP/
    ServerMonitor/
    RemoteSessionServer/
    HealthMonitor/
    RFCCompliance/
    DDNS/
    DOICPriorities/
    ODBCDataSources/
    AttributeGroups/
    KeyStores/
    Diameter/
    DiameterDictionary/

--> set LDAPMultiValDelimiter ,

Set LDAPMultiValDelimiter ,
```

Support for Session Query, POD, CoA over XML Interface

With this feature, the existing XML interface on UDP port (8080) is enhanced to handle session query, Change Of Authorization (CoA), and Packet of Disconnect (PoD) requests over REST/CLI.

A new XML tag attribute **Action** is introduced to handle the three types of requests. The Action tag value can be **Query**, **CoA**, or **PoD**. If a request does not contain the **Action** XML tag, Prime Access Registrar treats it as a session cache request and sends the response accordingly.

Based on the incoming request, Prime Access Registrar returns attributes which are configured in the session cache resource manager along with the attributes that are cached by default e.g. User-Name, Nas-Identifier, and so on.

If the Action XML tag in the request contains a value other than **Query**, **CoA**, or **PoD**, Prime Access Registrar drops the request.

Configuration Details

For XML Client, you need to configure the port in ***/Radius/Advanced/Ports/***. This is the port that the client uses to send the XML Packet.

Configure the default port as 1812 for RADIUS client.

```
--> cd advanced/ports/

[ //localhost/Radius/Advanced/Ports ]
Entries 1 to 2 from 2 total entries
Current filter: all

1812/
8080/

--> ls -R

[ //localhost/Radius/Advanced/Ports ]
Entries 1 to 2 from 2 total entries
```

```
Current filter: all

1812/
Port = 1812
Description =
Type = Radius
8080/
Port = 8080
Description =
Type = XML
```

Configure the xmlclient in ***/localhost/Radius/Client***.

```
[ //localhost/Radius/Clients ]
Entries 1 to 2 from 2 total entries
Current filter: all

localhost/
xml-client/

--> cd xml-client/

[ //localhost/Radius/Clients/xml-client ]
Name = xml-client
Description =
Protocol = radius
IPAddress = 10.126.246.117
SharedSecret =
Type = NAS
Vendor =
IncomingScript~ =
OutgoingScript~ =
3GPP-Teardown-Indicator = 0
EnableDynamicAuthorization = FALSE
NetMask =
EnableNotifications = FALSE
EnforceTrafficThrottling = TRUE

--> cd ..

[ //localhost/Radius/Clients ]
Entries 1 to 2 from 2 total entries
Current filter: all

localhost/
xml-client/

--> cd localhost/

[ //localhost/Radius/Clients/localhost ]
Name = localhost
Description =
Protocol = radius
IPAddress = 127.0.0.1
SharedSecret =
Type = NAS+Proxy
Vendor =
IncomingScript~ = ParseServiceHints
OutgoingScript~ =
3GPP-Teardown-Indicator = 0
EnableDynamicAuthorization = true
NetMask =
```

```
EnableNotifications = FALSE
EnforceTrafficThrottling = TRUE
DynamicAuthorizationServer/

--> cd dynamicAuthorizationServer/

[ //localhost/Radius/Clients/localhost/DynamicAuthorizationServer ]
Port = 3799
DynamicAuthSharedSecret =
InitialTimeout = 5000
MaxTries = 3
PODAttributeGroup =
COAAttributeGroup =
```

You need to configure the attributes which Prime Access Registrar needs to cache and return in response for each of the query requests as shown below:

```
//localhost/Radius/SessionManagers/sm1 ]
    Name = sm1
    Description =
    Type = local
    EnableDiameter = trUE
    IncomingScript =
    OutgoingScript =
    AllowAccountingStartToCreateSession = FALSE
    SessionTimeOut = 24H
    PhantomSessionTimeOut =
    SessionKey = User-Name:Session-Id
    SessionCreationCmdList = 268||305
    SessionDeletionCmdList = 275
    SessionRestorationTimeOut = 24h
    ResourceManagers/

--> cd resourceManagers/

[ //localhost/Radius/SessionManagers/sm1/ResourceManagers ]
    1. 3gpp
    2. sessioncache
    3. per-user

//localhost/Radius/ResourceManagers/sessioncache]
Name = sessioncache
       Description =
Type = session-cache
       OverwriteAttributes = FALSE
       EnableNon3GPPUserDataCaching = TRUE
       QueryKey =
       PendingRemovalDelay = 10
       AttributesToBeCached/
       QueryMappings/

       --> cd attributesToBeCached/

      [ //localhost/Radius/ResourceManagers/SessionCache/AttributesToBeCached ]
      1. Acct-Session-Id
      2. Framed-IP-Address
      3. Calling-Station-Id
      4. Cisco-SSG-Service-Info
      5. Cisco-SSG-Account-Info
```

Following examples show the request and response samples for Query, CoA, and PoD:

Query Request and Response

1. If Input Attribute is **User-Name**:

Request

```
<?xml version="1.0"?>
<Request>
    <Action>query</Action>
    <UserIdRequest>
        <UserId id_type="subscriber_id">bob</UserId>
    </UserIdRequest>
</Request>
```

Success Response

```
<?xml version="1.0"?>
<Response>
    <Association>
        <UserId id_type="subscriber_id">bob</UserId>
        <Action>query</Action>
        <status>Completed</status>
        <User-Name>bob</User-Name>
        <NAS-IP-Address>192.168.0.0</NAS-IP-Address>
        <NAS-Port>1</NAS-Port>
        <Framed-IP-Address>192.168.0.0</Framed-IP-Address>
        <NAS-Identifier>localhost</NAS-Identifier>
        <Acct-Session-Id>123</Acct-Session-Id>
        <Cisco-SSG-Service-Info>wifi</Cisco-SSG-Service-Info>
        <Cisco-SSG-Account-Info>credit</Cisco-SSG-Account-Info>
    </Association>
</Response>
```

Failure Response

```
<?xml version="1.0"?>
<Response>
  <Association>
    <UserId id_type="subscriber_id">bob</UserId>
     <Action>query</Action>
     <status>Failed</status>
  </Association>
</Response>
```

2. If Input Attribute is **Framed-IP-Address**:

Request

```
<?xml version="1.0"?>
<Request>
    <Action>query</Action>
    <UserIdRequest>
        <Address format="IPv4">192.168.0.4</Address>
    </UserIdRequest>
</Request>
```

Success Response

```
<?xml version="1.0"?>
<Response>
  <Association>
    <Address format="IPv4">192.168.0.4</Address>
     <Action>query</Action>
     <status>Completed</status>
     <User-Name>bob</User-Name>
     <NAS-IP-Address>192.168.0.0</NAS-IP-Address>
```

```
      <NAS-Port>1</NAS-Port>
      <Framed-IP-Address>192.168.0.0</Framed-IP-Address>
      <NAS-Identifier>localhost</NAS-Identifier>
      <Acct-Session-Id>123</Acct-Session-Id>
      <Cisco-SSG-Service-Info>wifi</Cisco-SSG-Service-Info>
      <Cisco-SSG-Account-Info>credit</Cisco-SSG-Account-Info>
  </Association>
</Response>
```

Failure Response

```
<?xml version="1.0"?>
<Response>
  <Association>
    <Address format="IPv4">192.168.0.4</Address>
     <Action>query</Action>
     <status>Failed</status>
  </Association>
</Response>
```

3. If Input Attribute is **Framed-IPv6-Address**:

Request

```
<?xml version="1.0"?>
<Request>
    <Action>query</Action>
    <UserIdRequest>
        <Address format="IPv4">2001:420:c0e0:1008::7e2</Address>
    </UserIdRequest>
</Request>
```

Success Response

```
<?xml version="1.0"?>
<Response>
    <Association>
        <Address format="IPv6">2001:420:c0e0:1008::7e2</Address>
        <Action>query</Action>
        <status>Completed</status>
        <User-Name>bob</User-Name>
        <NAS-IP-Address>192.168.0.0</NAS-IP-Address>
        <NAS-Port>1</NAS-Port>
        <Framed-IPv6-Address>2001:420:c0e0:1008::7e2</Framed-IPv6-Address>
        <NAS-Identifier>localhost</NAS-Identifier>
        <Acct-Session-Id>123</Acct-Session-Id>
        <Cisco-SSG-Service-Info>wifi</Cisco-SSG-Service-Info>
        <Cisco-SSG-Account-Info>credit</Cisco-SSG-Account-Info>
    </Association>
</Response>
```

Failure Response

```
<?xml version="1.0"?>
<Response>
    <Association>
        <Address format="IPv6">2001:420:c0e0:1008::7e2</Address>
        <Action>query</Action>
        <status>Failed</status>
    </Association>
</Response>
```

PoD Request and Response

1. If Input Attribute is **User-Name**:

Request

```
<?xml version="1.0"?>
    <Request>
        <Action>pod</Action>
        <UserIdRequest>
            <UserId id_type="subscriber_id">bob</UserId>
        </UserIdRequest>
</Request>
```

Success Response

```
<?xml version="1.0"?>
<Response>
    <Association>
        <UserId id_type="subscriber_id">bob</UserId>
        <Action>pod</Action>
        <status>Completed</status>
    </Association>
</Response>
```

Failure Response

```
<?xml version="1.0"?>
<Response>
    <Association>
        <UserId id_type="subscriber_id">bob</UserId>
        <Action>pod</Action>
        <status>Failed</status>
    </Association>
</Response>
```

2. If Input Attribute is **Framed-IP-Address**:

Request

```
<?xml version="1.0"?>
<Request>
    <Action>pod</Action>
    <UserIdRequest>
        <Address format="IPv4">192.168.0.4</Address>
    </UserIdRequest>
</Request>
```

Success Response

```
<?xml version="1.0"?>
<Response>
  <Association>
    <Address format="IPv4">192.168.0.4</Address>
     <Action>pod</Action>
     <status>Completed</status>
  </Association>
</Response>
```

Failure Response

```
<?xml version="1.0"?>
<Response>
  <Association>
    <Address format="IPv4">192.168.0.4</Address>
     <Action>pod</Action>
     <status>Failed</status>
```

```
  </Association>
</Response>
```

3. If Input Attribute is **Framed-IPv6-Address**:

Request

```
<?xml version="1.0"?>
<Request>
    <Action>pod</Action>
    <UserIdRequest>
    <Address format="IPv6">2001:420:c0e0:1008::7e2</Address>
    </UserIdRequest>
</Request>
```

Success Response

```
<?xml version="1.0"?>
<Response>
    <Association>
     <Address format="IPv6">2001:420:c0e0:1008::7e2</Address>
     <Action>pod</Action>
     <status>Completed</status>
    </Association>
</Response>
```

Failure Response

```
<?xml version="1.0"?>
<Response>
    <Association>
        <Address format="IPv6">2001:420:c0e0:1008::7e2</Address>
        <Action>pod</Action>
        <status>Failed</status>
    </Association>
</Response>
```

CoA Request and Response

1. If Input Attribute is **User-Name**:

Request

```
<?xml version="1.0"?>
<Request>
    <Action>coa</Action>
    <UserIdRequest>
        <UserId id_type="subscriber_id">bob</UserId>
    </UserIdRequest>
</Request>
```

Success Response

```
<?xml version="1.0"?>
<Response>
    <Association>
        <UserId id_type="subscriber_id">bob</UserId>
        <Action>coa</Action>
        <status>Completed</status>
    </Association>
</Response>
```

Failure Response

```
<?xml version="1.0"?>
```

```
<Response>
    <Association>
        <UserId id_type="subscriber_id">bob</UserId>
        <Action>coa</Action>
        <status>Failed</status>
    </Association>
</Response>
```

2. If Input Attribute is **Framed-IP-Address**:

Request

```
<?xml version="1.0"?>
<Request>
    <Action>coa</Action>
    <UserIdRequest>
        <Address format="IPv4">192.168.0.4</Address>
    </UserIdRequest>
</Request>
```

Success Response

```
<?xml version="1.0"?>
<Response>
  <Association>
    <Address format="IPv4">192.168.0.4</Address>
     <Action>coa</Action>
     <status>Completed</status>
  </Association>
</Response>
```

Failure Response

```
<?xml version="1.0"?>
<Response>
  <Association>
    <Address format="IPv4">192.168.0.4</Address>
     <Action>coa</Action>
     <status>Failed</status>
  </Association>
</Response>
```

3. If Input Attribute is **Framed-IPv6-Address**:

Request

```
<?xml version="1.0"?>
<Request>
    <Action>coa</Action>
    <UserIdRequest>
        <Address format="IPv6">2001:420:c0e0:1008::7e2</Address>
    </UserIdRequest>
</Request>
```

Success Response

```
<?xml version="1.0"?>
<Response>
    <Association>
        <Address format="IPv6">2001:420:c0e0:1008::7e2</Address>
        <Action>coa</Action>
        <status>Completed</status>
    </Association>
</Response>
```

Failure Response

```
<?xml version="1.0"?>
<Response>
    <Association>
        <Address format="IPv6">2001:420:c0e0:1008::7e2</Address>
        <Action>coa</Action>
        <status>Failed</status>
    </Association>
</Response>
```

Cisco Prime Access Registrar 9.2 Bugs

This section contains the following information:

- Using the Bug Search Tool, page 17

Using the Bug Search Tool

Use the Bug Search tool (BST) to get the latest information about Cisco Prime Access Registrar bugs. BST allows partners and customers to search for software bugs based on product, release, and keyword, and it aggregates key data such as bug details, product, and version.

BST allows you to:

- Quickly scan bug content
- Configure e-mail notifications for updates on selected bugs
- Start or join community discussions about bugs
- Save your search criteria so you can use it later

When you open the Bug Search page, check the interactive tour to familiarize yourself with these and other Bug Search features.

Step 1 Log into the Bug Search Tool.

a. Go to https://tools.cisco.com/bugsearch.

b. At the Log In screen, enter your registered Cisco.com username and password; then, click **Log In**. The Bug Search page opens.

Note If you do not have a Cisco.com username and password, you can register for them at http://tools.cisco.com/RPF/register/register.do.

Step 2 To search for a specific bug, enter the bug ID in the Search For field and press **Return**.

Step 3 To search for bugs in a particular release:

a. In the Search For field, enter the product name and the release version, e.g. Cisco Prime Access Registrar 9.2, and press **Return**. (Leave the other fields empty.)

b. When the search results are displayed, use the filter and sort tools to find the types of bugs you are looking for. You can search for bugs by severity, by status, how recently they were modified, according to the number of support cases associated with them, and so forth.

Related Documentation

For a complete list of Cisco Prime Access Registrar documentation, see the *Cisco Prime Access Registrar 9.2 Documentation Overview*.

Note We sometimes update the documentation after original publication. Therefore, you should also review the documentation on Cisco.com for any updates.