CHAPTER 10

IPsec Provisioning

This feature is not supported in this release.

Cisco IP Solution Center (ISC) supports IPsec provisioning for site-to-site VPNs, remote access VPNs, and network-based VPNs. To provision IPsec services, you create IPsec service definitions (policies) to use in IPsec service requests. The security policy defines the parameters for the VPN traffic and the VPN tunnel. The service request bundles together the service policy and the list of CPE devices for IPsec provisioning. The service request deployment creates device-level configurations (called configlets) and downloads these configlets to all security devices in your network.

This chapter describes IPsec service concepts and the process for provisioning IPsec services using the ISC API. The provisioning process example includes the steps from creating the inventory to auditing the service deployment.

For information on IPsec provisioning using the ISC GUI, refer to the *Cisco IP Solution Center Integrated VPN Management Suite Security User Guide, 4.0*.

This chapter contains the following sections:

IPsec Service Definitions

IPsec service definitions (policies) specify the parameters for the VPN traffic and the VPN tunnel. IPsec provisioning generally requires an encryption policy and an IPsec policy. However, the service definition types you need to create depend on the type of IPsec provisioning your network needs. IPsec policies can be for pure IPsec, IPsec + GRE, or for easy VPN services. IPsec Easy VPN services require a remote access policy, and remote access policies require an encryption policy.

Table 10-1 describes the service policies supported by ISC for IPsec provisioning.

Table 10-1 IPsec Service Policy Types

Service Policy Type	Definition	Requirements
Encryption	Defines the security parameters for the VPN traffic and consists IPsec proposals, IKE proposals, and global security parameters.	Required for all site-to-site and network based VPN policies (except Easy VPN), and for remote access VPN policies.
Site-to-Site IPsec	Defines pure IPsec tunnels (with no routing) between CPEs at customer sites. Provides data encryption at the packet level.	Requires an encryption policy.
Site-to-Site IPsec + GRE	Configures IPsec with Generic Routing Encapsulation (GRE) to allow multi-protocol routing, dynamic routing updates, and multicast access.	Requires an encryption policy.
Site-to-Site DMVPN	Dynamic Multipoint VPN (DMVPN) combines GRE tunnels, IPsec encryption, and the Next Hop Resolution Protocol (NHRP) into crypto profiles. Uses dynamically assigned IP addresses to allow dynamic tunnel creation between devices.	Requires an encryption policy.
Site-to-Site Easy VPN	Allows most VPN tunnel parameters to be defined on an Easy VPN server and pushed to Easy VPN Client devices.	Requires a remote access policy.
Network-Based IPsec	Network-based IPsec policies provide secure access through public networks to MPLS VPNs. The provisioning process using the API is the same as for site-to-site policies, except for the **NetworkBasedSolution** parameter value.	Requires an encryption policy.
Network-Based IPsec + GRE		Requires an encryption policy.
Network-Based IPsec Easy VPN		Requires a remote access policy.
Remote Access VPN	A VPN client on a user's workstation initiates the VPN tunnel with a VPN gateway (typically a CPE device at the edge of the corporate site).	Requires an encryption policy.

The following sections describe the IPsec policy types.

Encryption Policies

An encryption service definition (policy) defines the security parameters for protecting data traveling through the VPN tunnel. It specifies the IKE and IPsec proposal parameters for the IPsec VPN and through the global attributes defines the level of encryption used in the IPsec VPN tunnels.

- IKE proposals—Provides authentication between IPsec peers and negotiates IPsec keys and security associations (SAs). IKE proposal parameters include encryption algorithm, hash algorithm, authentication method, Diffie-Hellman group settings, and SA lifetime.
- IPsec proposals—Defines the network layer encryption and authentication control. IPsec parameters include the encryption, authentication, and compression algorithms.
- Global attributes—Perfect Forward Secrecy (PFS) settings, IKE and IPsec lifetime and size parameters, and IKE keepalive intervals.

Site-to-Site VPN Policies

An IPsec site-to-site VPN service definition (policy) defines the parameters for site-to-site VPN tunnels, the IPsec parameters used to create the VPN tunnel, the network topology (either full mesh or hub-and-spoke), and the routing protocol to use in the VPN tunnel.

The ISC API supports the following site-to-site VPN policy types:

- **SitetoSite_IPSec**—Pure IPsec uses no routing and provides network data encryption at the IP packet level.
- **SitetoSite_IPSec_GRE**—GRE is used in conjunction with IPsec so that routing updates can take place within the tunnel.
- **SitetoSite_DMVPN**— Combines GRE tunnels, IPsec encryption, and NHRP into crypto profiles, to use in place of static crypto maps.
- **SitetoSite_EasyVPN**—VPN parameters are defined on an Easy VPN server and downloaded to an Easy VPN Client device.

Network-Based VPN Policies

Network-based service definitions (policies) are used with MPLS VPN service requests to connect an existing MPLS VPN network with sites separated from the MPLS VPN by a public network such as the Internet. The IPsec-to-MPLS mapping in the network-based service policy enables these sites to connect to the MPLS core and join an existing MPLS VPN.

The process for creating network-based service definitions (policies) and the parameters to set is similar to the process for site-to-site policies. For network-based service definitions you must specify a network-based **SubType** parameters in the **ServiceDefinitionDetails**. The API supports these subtypes:

- **NetworkBased_IPSec**
- **NetworkBased_IPSec_GRE**
- **NetworkBased_EasyVPN**

Note For information on provisioning network-based VPN policies using the ISC GUI, see the *Cisco IP Solution Center Integrated VPN Management Suite Network-Based IPsec VPN User Guide, 4.0*.

Remote Access VPN Policies

A remote access service definition (policy) defines the IPsec parameters for the VPN client and VPN gateway to use to create the VPN tunnel. The initial tunnel negotiation consists of device authentication through Internet Key Exchange (IKE), followed by user authentication using IKE extended authentication (XAuth). The group profile is pushed to the VPN Client using mode configuration, and an IPsec security association (SA) is created to complete the VPN connection.

The remote access VPN policy defines:

- Encryption policy
- IKE XAuth parameters for user authentication
- Mode configuration parameters for policy push and features such as dynamically assigned client IP addresses
- Remote access parameters
- User group parameters

Note The group policy information is stored locally in the VPN device configuration. When the user or group information is stored on AAA servers, you must also configure access to the AAA server.

IPsec Service Requests

When you create an IPsec service request, you define the traffic to send through the VPN tunnel. All devices in the same service request must use the same encryption policy and VPN definition.

The ISC API supports two types of IPsec service requests:

- **IPSecSitetoSite**—For site-to-site and network-based IPsec.
- **IPSecRA**—For remote access service requests

IPsec Site-to-Site Service Requests

An IPsec site-to-site service request consists of the VPN, the IPsec site-to-site or network-based IPsec service policy, the topology of the network (hub-and-spoke or full mesh), and the **IPsecLink**. The IPsecLink object defines the CPE to receive the service request configuration, the CPE role, backup and failover devices, and template information. An IPsec service request can also include any attribute marked as editable in the site-to-site or network-based VPN policy.

The provisioning process for network-based policies is the same as the process for site-to-site policies, except for the **NetworkBasedSolution** parameter value.

- For service requests used to provision site-to-site VPN policies, set **NetworkBasedSolution=None**.
- For service requests used to provision network-based VPN policies, set **NetworkBasedSolution=IPSEC_TO_MPLS_MAPPING**.

A site-to-site service request example is described in the "Creating the IPsec Service Request" section on page 10-19.

IPsec Remote Access Service Requests

To provision remote access service policies, create an IPsec remote access (**IPsecRA**) service request.

An IPsec remote access service request consists of the VPN, customer, the remote access service policy, the AAA servers, and the **IPsecLink**. The IPsecLink object defines the CPE to receive the service request configuration, the AAA device interface, and template information.

Table 10-2 describes the object definitions (classNames) and parameters for creating a remote access service request.

Table 10-2 Creating Remote Access IPsec Service Request

Operation	className	Required Parameters
createInstance	ServiceRequest	• RequestName • Organization • Type=IPSecRA • ServiceRequestDetails
	ServiceRequestDetails	• VPN • ServiceDefinition – ServiceDefinitionType=IPSecRA • AAAServer • IPSecLink
	IPSecLink	• Cpe • AAAServerInterface **Note** Use **IPSecRATemplate** to add templates to this service request. See the "Link Template" section on page 4-9 for more information.

Tip Because each remote access policy defines a user group, you can use multiple remote access policies in the same service request. This allows you to configure multiple user groups on the same CPE device.

XML Examples:

- CreateIPSecRAServiceOrder.xml

See the following example:

```
<ns1:performBatchOperation>
    <actions xsi:type="ns1:CIMActionList"
             soapenc:arrayType="ns1:CIMAction[]">
     <action>
      <actionName xsi:type="xsd:string">createInstance</actionName>
      <objectPath xsi:type="ns1:CIMObjectPath">
       <className xsi:type="xsd:string">ServiceOrder</className>
       <properties xsi:type="ns1:CIMPropertyList"
                   soapenc:arrayType="ns1:CIMProperty[]">
        <item xsi:type="ns1:CIMProperty">
         <name xsi:type="xsd:string">ServiceName</name>
```

```
    <value xsi:type="xsd:string">ServiceOrderforIPSecRA</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">CarrierId</name>
    <value xsi:type="xsd:string">322</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">Organization</name>
    <value xsi:type="xsd:string">NbiCustomer</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">DesiredDueDate</name>
    <value xsi:type="xsd:dateTime">2002-12-13T14:55:38.885Z</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">Remarks</name>
    <value xsi:type="xsd:string">This is the remark</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">NumberOfRequests</name>
    <value xsi:type="xsd:string">1</value>
   </item>
  </properties>
 </objectPath>
</action>
<action>
 <actionName xsi:type="xsd:string">createInstance</actionName>
 <objectPath xsi:type="ns1:CIMObjectPath">
 <className xsi:type="xsd:string">ServiceRequest</className>
 <properties xsi:type="ns1:CIMPropertyList"
             soapenc:arrayType="ns1:CIMProperty[]">
  <item xsi:type="ns1:CIMProperty">
   <name xsi:type="xsd:string">Organization</name>
   <value xsi:type="xsd:string">NbiCustomer</value>
  </item>
  <item xsi:type="ns1:CIMProperty">
   <name xsi:type="xsd:string">RequestName</name>
   <value xsi:type="xsd:string">MYSR-2</value>
  </item>
  <item xsi:type="ns1:CIMProperty">
   <name xsi:type="xsd:string">Type</name>
   <value xsi:type="xsd:string">IPSecRA</value>
  </item>
   </properties>
  <objectPath xsi:type="ns1:CIMObjectPath">
  <className xsi:type="xsd:string">ServiceRequestDetails</className>
  <properties xsi:type="ns1:CIMPropertyList"
              soapenc:arrayType="ns1:CIMProperty[]">
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">Description</name>
    <value xsi:type="xsd:string">This is an IPSec Remote Access SR</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">VPN</name>
    <value xsi:type="xsd:string">vpnX</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">ServiceDefinition</name>
    <value xsi:type="xsd:string">RAPolicy0709_2</value>
    <qualifier xsi:type="xsd:string">
      <name xsi:type="xsd:string">ServiceDefinitionType</name>
      <value xsi:type="xsd:string">IPSecRA</value>
    </qualifier>
   </item>
```

```
 <item xsi:type="ns1:CIMProperty">
  <name xsi:type="xsd:string">ServiceDefinition</name>
  <value xsi:type="xsd:string">70</value>
  <qualifier xsi:type="xsd:string">
    <name xsi:type="xsd:string">ServiceDefinitionType</name>
    <value xsi:type="xsd:string">IPSecRA</value>
  </qualifier>
 </item>
 <item xsi:type="ns1:CIMProperty">
  <name xsi:type="xsd:string">NetworkBasedSolution</name>
<value xsi:type="xsd:string">None</value>
</item>
 <item xsi:type="ns1:CIMProperty">
  <name xsi:type="xsd:string">AAAServer</name>
  <value xsi:type="xsd:string">2</value>
 </item>
</properties>
<objectPath xsi:type="ns1:CIMObjectPath">
<className xsi:type="xsd:string">IPSecLink</className>
  <properties xsi:type="ns1:CIMPropertyList"
              soapenc:arrayType="ns1:CIMProperty[]">
   <item xsi:type="ns1:CIMProperty">
   <name xsi:type="xsd:string">Cpe</name>
    <value xsi:type="xsd:string">ensw2950-1</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
   <name xsi:type="xsd:string">AAAServerInterface</name>
   <value xsi:type="xsd:string">1</value>
   </item>
  </properties>
<objectPath xsi:type="ns1:CIMObjectPath">
 <className xsi:type="xsd:string">IPSecRATemplate</className>
  <properties xsi:type="ns1:CIMPropertyList"
              soapenc:arrayType="ns1:CIMProperty[]">
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">DeviceId</name>
    <value xsi:type="xsd:string">1</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">DatafilePath</name>
    <value xsi:type="xsd:string">/nbi/AccessList</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">DatafileName</name>
    <value xsi:type="xsd:string">MyTemplate1</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">TemplateActive</name>
    <value xsi:type="xsd:string">true</value>
   </item>
   <item xsi:type="ns1:CIMProperty">
    <name xsi:type="xsd:string">TemplateAction</name>
    <value xsi:type="xsd:string">APPEND</value>
   </item>
  </properties>
</objectPath>
```

Provisioning Example

This provisioning example describes the process for creating an IPsec site-to-site service policy for Easy VPN and an IPsec site-to-site service request to provision the site-to-site policy. IPsec Easy VPN services require a remote access policy, and remote access policies require an encryption policy.

Prerequisites

To provision IPsec services with ISC, you must have:

- IPv4 connectivity among devices in your network, and between ISC and the devices in your network.
- Any unmanaged devices you use in the VPN must have an IPsec configuration that matches the managed peer CPE device IPsec configuration, and a matching crypto map between peer devices.

Process Summary

This IPsec provisioning example includes the following operations:

- Preparing Inventory
  - Create devices
  - Create customers
  - Create customer sites
  - Declare CPE devices and mark security interfaces for CPE devices
  - Create a VPN to use in the site-to-site and remote access service policies
  - Create a AAA server
- Creating the IPsec service definition
  - Encryption policy (to use in the site-to-site policy)
  - Remote Access policy (to use in the site-to-site service request)
  - IPsec site-to-site service policy for Easy VPN
- Creating the site-to-site IPsec service request

Provisioning Process

This section describes the process for provisioning IPsec using XML examples.

The complete list of XML examples for IPsec are located at:
http://www.cisco.com/univercd/cc/td/doc/product/rtrmgmt/isc/4_0/api/apiref/examples/index.htm

Note For clarity, this provisioning process shows each step as a separate XML request. Many of these steps can be combined using **performBatchOperations**.

Preparing Inventory

Use these steps to prepare the inventory in the ISC repository for IPsec provisioning.

Step 1 Create devices.

Every network element that ISC manages must be defined as a device in the system. An element is any device from which ISC can collect configuration information. ISC supports these devices for security provisioning:

- Cisco IOS devices (**CiscoRouter** or **CatIOS**) for pure IPsec, GRE + IPsec, DMVPN, Easy VPN, and remote access.
- PIX security appliances (**PIX**) for pure IPsec, Easy VPN, and remote access.
- VPN 3000 concentrator (**VPN3000**) for pure IPsec and remote access.
- VPN service modules (**VPNSM**) for catalyst switches running Cisco IOS (**CatIOS**)

When you add devices to your network, you must designate device interface roles and assign an IP address to each interface (except for **VPNSM**s). ISC uses the interface roles to configure IPsec services.

Table 10-3 Create Devices

Operation	className	Required Parameters
createInstance	• CiscoRouter • CatIOS **Note** You must use a **CatIOS** device (not *CatOS*) for IPsec provisioning. • PIX • VPN3000	One or more of the following: • ManagementIPAddress • HostName • DomainName • Interface
	Interface	• Name • IPAddress • InterfaceEncapType (for **CiscoRouter**, **CatIOS**, and **VPN3000**) • PortType (for **CatIOS**)= – ACCESS – TRUNK • VlanId (for **CatIOS**) • InterfaceType (for **PIX**)

XML Examples:

- CreateCiscoRouter.xml
- CreateCatIOS.xml
- CreatePIX.xml
- CreateVPN3000.xml

A VPN service module (VPMSM) is a blade on a catalyst switch device running the Cisco IOS operating system. You must add the parent catalyst switch device to the repository before you can create the VPNSM to use as a CPE device in IPsec provisioning.

Table 10-4 Create VPN Service Module

Operation	className	Required Parameters
createInstance	CreateVPNSM	• ModuleName • ModuleNumber (slot number) • Device=CatIOS **Note** The catalyst switch running Cisco IOS (**CatIOS**) must already exist in the repository.

XML Examples:

- CreateVPNSM.xml

Step 2 Create customers (organizations).

A customer is a requestor of VPN services. Each customer can contain multiple customer sites. Each site belongs to only one customer and can contain multiple CPEs.

Table 10-5 Create Organization

Operation	className	Required Parameters
createInstance	Organization	• Name

XML Examples:

- CreateOrganization.xml

Step 3 Create sites and assign organizations to them.

Table 10-6 Create Sites

Operation	className	Required Parameters
createInstance	Site	• Name • Organization

XML Examples:

- CreateSite.xml

Step 4 Declare devices as CPEs. Each device in your network to receive IPsec provisioning must be designated as a CPE device.

Table 10-7 Create CPE Devices

Operation	className	Required Parameters
createInstance	Cpe	• Site • ManagementType • Device • PresharedKeys • IPSecHA (for IPsec High Availability options) • UserDefinedPublicIpAddress

XML Examples:

- CreateCpe.xml

Note If you are using preshared keys for authentication in your Encryption policy, specify the appropriate type of preshared key (wildcard, host, or group) when you create the CPE device.

See the following example:

```
<ns1:createInstance>
    <objectPath xsi:type="ns1:CIMObjectPath">
     <className xsi:type="xsd:string">Cpe</className>
     <properties xsi:type="ns1:CIMPropertyList"
                  soapenc:arrayType="ns1:CIMProperty[]">
      <item xsi:type="ns1:CIMProperty">
       <name xsi:type="xsd:string">Site</name>
       <value xsi:type="xsd:string">Site1</value>
      </item>
      <item xsi:type="ns1:CIMProperty">
       <name xsi:type="xsd:string">Device</name>
       <value xsi:type="xsd:string">Device1</value>
      </item>
      <item xsi:type="ns1:CIMProperty">
       <name xsi:type="xsd:string">ManagementType</name>
       <value xsi:type="xsd:string">MULTI_VRF</value>
      </item>
     </properties>
     <!-- This is an example of a Host Preshared Key: -->
     <!--objectPath xsi:type="ns1:CIMObjectPath">
      <className xsi:type="xsd:string">PresharedKey</className>
      <properties xsi:type="ns1:CIMPropertyList"
                  soapenc:arrayType="ns1:CIMProperty[]">
       <item xsi:type="ns1:CIMProperty">
        <name xsi:type="xsd:string">Device</name>
        <value xsi:type="xsd:string">Device1</value>
       </item>
       <item xsi:type="ns1:CIMProperty">
        <name xsi:type="xsd:string">RemoteDevice</name>
        <value xsi:type="xsd:string">Device2</value>
       </item>
       <item xsi:type="ns1:CIMProperty">
        <name xsi:type="xsd:string">PresharedKey</name>
        <value xsi:type="xsd:string">12345678</value>
       </item>
      </properties>
```

```
</objectPath-->
<!-- This is an example of a Group Preshared Key: -->
<!--objectPath xsi:type="ns1:CIMObjectPath">
 <className xsi:type="xsd:string">PresharedKey</className>
 <properties xsi:type="ns1:CIMPropertyList"
             soapenc:arrayType="ns1:CIMProperty[]">
  <item xsi:type="ns1:CIMProperty">
   <name xsi:type="xsd:string">Device</name>
   <value xsi:type="xsd:string">Device1</value>
  </item>
  <item xsi:type="ns1:CIMProperty">
   <name xsi:type="xsd:string">SubnetAddress</name>
   <value xsi:type="xsd:string">10.0.0.0/24</value>
  </item>
  <item xsi:type="ns1:CIMProperty">
   <name xsi:type="xsd:string">PresharedKey</name>
   <value xsi:type="xsd:string">87654321</value>
  </item>
 </properties>
</objectPath-->
<!-- This is an example of a Wildcard Preshared Key. You can have
     only one wildcard preshared key: -->
<!--objectPath xsi:type="ns1:CIMObjectPath">
 <className xsi:type="xsd:string">PresharedKey</className>
 <properties xsi:type="ns1:CIMPropertyList"
             soapenc:arrayType="ns1:CIMProperty[]">
  <item xsi:type="ns1:CIMProperty">
   <name xsi:type="xsd:string">Device</name>
   <value xsi:type="xsd:string">Device1</value>
  </item>
  <item xsi:type="ns1:CIMProperty">
   <name xsi:type="xsd:string">PresharedKey</name>
   <value xsi:type="xsd:string">13577531</value>
  </item>
 </properties>
</objectPath-->
```

Step 5 Mark CPE device interfaces. For IPsec, you must define at least one public and one private interface on each device.

- Public—Interfaces on which VPN tunnels terminate
- Private—Interfaces behind which the customer subnets reside

You can define the public or private interfaces when you create the CPE device (see Step 4), or if the CPE device is already in the repository, use a **modifyInstance** to mark the device interfaces.

Table 10-8 Mark CPE Device Interfaces using a modifyInstance

Operation	className	Required Parameters
modifyInstance	Cpe	• Site • Device • Interface
	Interface	• Name • PublicInterface (**true** or **false**) • PrivateInterface (**true** or **false**) **Note** Each device must have one public and one private interface defined.

Note When you create a CPE device from a VPNSM, the port VLAN interface for devices must be marked **PublicInterface=true**.

XML Examples:

- ModifyCpe.xml

See the following example to modify a CPE device:

```
<ns1:modifyInstance>
      <objectPath subAction="modifyInstance" xsi:type="ns1:CIMObjectPath">
        <className xsi:type="xsd:string">Cpe</className>
        <keyProperties xsi:type="ns1:CIMKeyPropertyList"
            soapenc:arrayType="ns1:CIMKeyProperty[]">
          <item xsi:type="ns1:CIMKeyProperty">
            <name xsi:type="xsd:string">Device</name>
            <value xsi:type="xsd:string">ipsecdevice2</value>
          </item>
        </keyProperties>
        <objectPath subAction="createInstance" xsi:type="ns1:CIMObjectPath">
          <className xsi:type="xsd:string">Interface</className>
          <properties xsi:type="ns1:CIMPropertyList"
              soapenc:arrayType="ns1:CIMProperty[]">
            <item xsi:type="ns1:CIMProperty">
              <name xsi:type="xsd:string">Name</name>
              <value xsi:type="xsd:string">device2intf</value>
            </item>
              <item xsi:type="ns1:CIMProperty">
              <name xsi:type="xsd:string">PublicInterface</name>
              <value xsi:type="xsd:string">true</value>
             </item>
            </item>
              <item xsi:type="ns1:CIMProperty">
              <name xsi:type="xsd:string">PrivateInterface</name>
              <value xsi:type="xsd:string">false</value>
             </item>
          </properties>
        </objectPath>
      </objectPath>
    </ns1:modifyInstance>
```

Step 6 Create a VPN to group customer site devices for site-to-site and remote access service policies.

Table 10-9 Create a VPN

Operation	className	Required Parameters
createInstance	VPN	• Name • Organization

XML Example:

- CreateVPN.xml

Step 7 Create the AAA server (optional).

The AAA server is required if the user authentication method is external for the IPsec group (**TYPE=EXTERNAL** in the remote access service policy).

Table 10-10 Create the AAA Server

Operation	className	Required Parameters
createInstance	AAAServer	• Name • Organization • Number of Retries • Timeout • Address • AuthServerType= – RADIUS – NTDOMAIN – SDI – TACACS+ • Role= – AUTHENTICATION – ACCOUNTING – BOTH

XML Examples:

- CreateAAServer.xml
- CreateAAServerNTDOMAIN.xml
- CreateAAServerRADIUS.xml
- CreateAAServerSDI.xml
- CreateAAServerTACACS.xml

Creating the Encryption Policy

An encryption policy is required to create a remote access service policy. It is also required to create site-to-site and network-based service policies for pure IPsec, IPsec+GRE, and DMVPN.

The encryption policy specifies the IKE and IPsec proposal parameters for the IPsec VPN and the global attributes which defines the level of encryption used in the IPsec VPN tunnels.

Table 10-11 Create an Encryption Policy

Operation	className	Required Parameters
createInstance	ServiceDefinition	• Name • Type=IPSec • Global or Organization **Note** If you do not specify an **Organization**, the service policy is global. • ServiceDefinitionDetails
	ServiceDefinitionDetails	• Global attributes: – GlobalIPSecLifetime – GlobalIPSecLifesize – GlobalIKELifetime – PFS – IKEKeepaliveInterval – IKEKeepaliveRetryInterval **Tip** To disable IKE keepalives, set *both* IKE keepalive values to zero. • IKEProposal attributes: – AuthenticationMethod – EncryptionAlgorithm – HashAlgorithm – DHGroup – SALifetime • IPSecProposal attributes: – AHAuthenticationAlgorithm – ESPAuthenticationAlgorithm – ESPEncryptionAlgorithm – CompressionAlgorithm **Note** You can have multiple **IKEProposals** and **IPSecProposals**.

XML Examples:

- CreateIPSecEncryptionServiceDef.xml
- CreateIPSecEncryptionServiceDef2.xml

Creating the Remote Access VPN Policy

The remote access service definition (policy) defines the characteristics of an IPsec tunnel between a customer site and a remote user. It consists of an encryption policy, VPN group attributes, IP address pools, split tunneling subnets, and device group parameters.

Table 10-12 Create a Remote Access IPsec Policy

Operation	className	Required Parameters
createInstance	ServiceDefinition	• Name • Type=IPSecRA • Global or Organization **Note** If you do not specify an **Organization**, the service policy is global. • ServiceDefinitionDetails
	ServiceDefinitionDetails	• SubType=RemoteAccess • Policy=<choose an encryption policy> General parameters: • TYPE= – INTERNAL – EXTERNAL • IKE_PASSWORD • TUNNELING_PROTOCOL= – IPSEC – L2TPOVERIPSEC (for **VPN3000** only) • AUTH_METHOD= – NONE – RADIUS – INTERNAL – NTDOMAIN – SDI – TACACS+

Table 10-12 Create a Remote Access IPsec Policy (continued)

Operation	className	Required Parameters
		• AddressPool • SPLIT_TUNNELING_POLICY= – everything – in_list – not_in_list • User • SplitTunnel (required if **SPLIT_TUNNELING_POLICY= in_list** or **not_in_list**)
		Device group parameters: • GroupIOS • GroupPIX • Group3K • AccessHours (for **VPN3000** only)

XML Examples:

- CreateRemoteAccessServiceDefn.xml

Creating the Site-to-Site VPN Policy

This site-to-site VPN policy for Easy VPN consists of an encryption policy, the network topology, the mode, and extended authorization (XAuth) parameters.

Note For each service definition parameter (except in remote access policies), you can set an additional attribute, **editable=true,** to allow the network operator to override these attributes when creating the service request. If an attribute is set to **editable=false**, these attributes cannot be changed in the service request.

Table 10-13 Create IPsec Site-to-Site Policy for Easy VPN

Operation	className	Required Parameters
createInstance	ServiceDefinition	• Name • Type=IPSecSitetoSite • Global or Organization **Note** If you do not specify an **Organization**, the service policy is global. • ServiceDefinitionDetails
	ServiceDefinitionDetails	• SubType= – SitetoSite_EasyVPN • Policy=<choose an encryption policy> • Topology=Hub_and_Spoke • Mode= – CLIENT – NETWORK-EXTENSION • XAuthUserName • XAuthPassword **Note** XAuth is for PIX devices only. • Enable_DHCP_Server_(Remote) (for **CiscoIOS** only)

Note In IPsec service definitions, the className **Policy** is used to specify both remote access and encryption policies. Be sure to use the correct policy name for your policy type. Site-to-site and network based policies for IPsec, IPsec+GRE, and DMVPN, and remote access policies all require an encryption policy. Easy VPN policies require a remote access policy.

XML Examples:

- CreateSitetoSiteEasyVPNServiceDefn.xml

Creating the IPsec Service Request

Use a standard IPsec service request to provision an Easy VPN service policy. The service request defines what traffic should be sent through the VPN tunnel and what traffic should not. You can provision multiple CPE devices using the same policy in one or more service requests.

Note To provision a remote access service policy see "IPsec Remote Access Service Requests" section on page 10-5

An IPsec site-to-site service request for Easy VPN consists of the VPN, the remote access VPN policy, the network topology, and the **IPsecLink**. The IPsecLink object defines the CPE to receive the service request configuration, the CPE role, backup and failover devices, and template information. The service request can also include any attribute marked as editable in the service policy.

Note All devices in the same service request must use the same encryption policy and VPN definition.

Table 10-14 Create a IPsec Site-to-Site Service Request

Operation	className	Required Parameters
performBatchOperations		
createInstance	ServiceOrder	• ServiceName • Organization • NumberOfRequests • ServiceRequest
createInstance	ServiceRequest	• RequestName • Organization • Type=IPSecSitetoSite • ServiceRequestDetails
	ServiceRequestDetails	• VPN • ServiceDefinition – ServiceDefinitionType=IPSecSitetoSite • SRTopology= – HUB_N_SPOKE – FULL_MESH • IPSecLink
	IPSecLink	• Cpe • Role= – SPOKE – HUB • BackupCpe (for failover devices) **Note** Use **IPSecTemplate** to add templates to this service request. See the "Templates in a Service Request" section on page 4-9.

XML Examples:

- CreateIPSecServiceOrder.xml

Auditing Service Requests

A configuration audit occurs automatically each time you deploy a service request. During this configuration audit, ISC verifies that all Cisco IOS commands are present and that they have the correct syntax. An audit also verifies that there were no errors during deployment by examining the commands configured by the service request on the target devices. If the device configuration does not match what is defined in the service request, the audit flags a warning and sets the service request to a *Failed Audit* or *Lost* state.

If you do not want the configuration audit to occur, change the value for the **Audit** parameter. The **Audit** parameter supports these values:

- **Audit**—This is the default. A successfully deployed service request is automatically audited unless this flag is changed.
- **NoAudit**—Do not perform a configuration audit when the service request is deployed.
- **ForceAudit**—Perform a configuration audit even if the service request deployment is not successful.

You can use the Audit parameter with a **Create**, **Modify**, or **Decommission** service request or a **Deployment** task. See the "Service Decommission" section on page 3-10 for more information.

To perform a configuration audit as a separate task, an IPSec functional audit, or a certificate enrollment audit, see the "Tasks" section on page 3-8.