Creating and Modifying EAP-FAST Profiles for Distribution to Users

This chapter explains how configure EAP-FAST module profiles both by using a Group Policy Object editor and by modifying the EAP-FAST XML schema.

The following topics are covered in this chapter:

- Overview of Group Policy Objects, page 4-2
- Adding a Group Policy Object Editor, page 4-2
- Creating a Group Policy Object in Windows Vista, page 4-3
- The EAP-FAST XML Schema, page 4-4
- Configuring Machine Authentication, page 4-15
- Configuring Single Sign-On, page 4-15
Overview of Group Policy Objects

Group Policy is an infrastructure that allows you to specify managed configurations for users and computers in an Active Directory directory service environment. Group Policy settings are contained in Group Policy objects (GPOs). GPOs exist in a domain and can be linked to the following Active Directory containers: sites, domains, or organizational units (OUs).

Microsoft provides a program snap-in that allows you to use the Group Policy Object editor in the Microsoft Management Console (MMC).

For more information about the MMC, refer to the Microsoft Management Console Help at this URL: http://www.microsoft.com/technet/WindowsVista/library/ops/06e1cb7b-19c9-4c49-9db8-a941f6f593c3.mspx

Adding a Group Policy Object Editor

Before you configure a Group Policy Object, you must add a Group Policy Object Editor snap-in. To add the snap-in, perform the following steps:

Step 1 Open the MMC:
   a. Click the Start button on the lower-left corner of the desktop.
   b. Enter mmc in the Search box and press Enter.

   To open an existing or saved MMC console, browse to the snap-in console or a shortcut to the snap-in console in Windows Explorer, and then double-click it.

   You can also open an existing MMC console from another console in which you are working. To do this, click the File menu, and then click Open.

Step 2 Add the Group Policy Object Editor snap-in:
   a. Go to File > Add/Remove Snap-in...
      The Add or Remove Snap-ins dialog box is displayed.
   b. From the Add or Remove Snap-ins dialog box, highlight Group Policy Object Editor in the Available snap-ins list, and click the Add button.
      The Select Group Policy Object dialog box is displayed.
   c. From the Select Group Policy Object dialog box, click Browse.
      The Browse for a Group Policy Object dialog box is displayed.
   d. From the Browse for a Group Policy Object dialog box, select the Domains/OUs tab.
   e. Select your domain controller from the Look in drop down list.
Chapter 4  Creating and Modifying EAP-FAST Profiles for Distribution to Users

Creating a Group Policy Object in Windows Vista

To create a new EAP group policy object, perform the following steps:

**Step 1**  In the **Default Domain Policy** pane, select **Windows Settings > Security Settings > Wireless Network Policies**.

**Step 2**  Right-click **Wireless Network Policies** and select **Create a New Policy**.

**Step 3**  Set your wireless network properties, such as SSID, encryption, and authentication method.

**Step 4**  Select the EAP method.

**Step 5**  Open the EAP-FAST properties and configure the EAP-FAST settings.

*Note*  In the **Advanced Security** screen, you can configure supplicant settings such as machine authentication and SSO. For more information about machine authentication, see the “Configuring Machine Authentication” section on page 4-15. For more information about SSO see the “Configuring Single Sign-On” section on page 4-15.

*Note*  You can configure settings for a wired network by selecting the **Wired Network Policy** object.

**Step 6**  After you are done, save the GPO. You can refresh the Vista client by running "gpupdate /force" to force update of the GPO. You should see the new profile being added to Vista machine.

After you create a GPO network profile, it cannot be changed by the user on the Vista machine.

On the General tab of a wireless network policy, you can configure a name and description for the policy, specify whether the WLAN AutoConfig service is enabled, and configure a list of wireless network policies and their settings in a preferred order. You can also export profiles as XML files and import XML files as wireless profiles.

For detailed information about configuring policies, exporting profiles, and importing profiles, see the following documentation:

- *Windows Vista Wireless Networking Evaluation Guide*  
  http://technet2.microsoft.com/WindowsVista/en/library/f0b0d1fd-6dff-46a2-8e6a-bdd152d2337f1033.mspx?mfr=true

- *Wireless Group Policy Settings for Windows Vista*  
The EAP-FAST XML Schema

The EAP-FAST module stores all settings in the Native EAP method section of the network profile as XML by using the following schema:

```xml
<?xml version="1.0"?>
<!--
*******************************************************************************
Cisco EAP-FAST Schema             (1.0.40)
Copyright 2006-2007, Cisco Systems, Inc.                   All rights reserved.
*******************************************************************************
-->

<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns="http://www.cisco.com/CCX"
 targetNamespace="http://www.cisco.com/CCX"
 elementFormDefault="qualified"
 attributeFormDefault="unqualified">

<xs:element name="eapFast" type="EapFast"/>

<xs:complexType name="EapFast">
<xs:complexContent>
<xs:extension base="TunnelMethods">
<xs:sequence>
<xs:choice>
<xs:element name="usePac">
<xs:complexType>
<xs:sequence>
<xs:element name="allowUnauthPacProvisioning" type="xs:boolean" default="true">
<xs:annotation>
<xs:documentation>Will accept a PAC from an unauthenticated server.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="autoGrouping" type="xs:boolean" default="true">
<xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:choice>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:schema>
```
An aid-group is a set of A-IDs that are all trusted equally. Any A-ID in the group can be utilized. Auto-grouping means that when an untrusted A-ID is accepted by the end-user then that A-ID is grouped with the A-ID(s) that were already trusted for that profile, hence automatically creating and growing an A-ID group based on user actions. The advantage of an A-ID group is that if a profile initially starts with the same trusted A-ID(1) and then at some point the end-user authorizes the use of a new A-ID(2) when using this profile it will accept A-ID(2) without bothering the end-user a second time.

If true, then when the client is about to do unauthenticated provisioning, the user will be prompted to allow or disallow the unauthenticated provisioning.
The EAP-FAST XML Schema

</xs:element>
</xs:choice>

</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
<xs:element name="authenticateWithToken">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="protectedIdentityPattern" type="IdentityPattern" minOccurs="0">
        <xs:annotation>
          <xs:documentation>Format rules same as for unprotectedIdentityPattern. Typical pattern: [username]@[domain] </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="tokenSource" type="TokenSource"/>
      <xs:element name="methods">
        <xs:complexType>
          <xs:all>
            <xs:element name="eapGtc" type="Empty"/>
          </xs:all>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="authenticateWithCertificate">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="protectedIdentityPattern" type="IdentityPattern" minOccurs="0">
        <xs:annotation>
          <xs:documentation>Format rules same as for unprotectedIdentityPattern. Typical pattern: [username]@[domain] </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="certificateSource" type="CertificateSource"/>
      <xs:choice>
        <xs:element name="doNotUseInnerMethod"/>
      </xs:choice>
    </xs:sequence>
  </xs:complexType>
</xs:element>
The EAP-FAST XML Schema

</xs:element>
<xs:element name="sendViaInnerMethod">
<xs:complexType>
<xs:all>
<xs:element name="eapTls" type="Empty"/>
</xs:all>
</xs:complexType>
</xs:element>
</xs:choice>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:choice>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:extension>
</xs:complexContent>
</xs:complexType>

<xs:complexType name="IdentityPattern">
<xs:simpleContent>
<xs:extension base="NonEmptyString">
<xs:attribute name="encryptContent" type="xs:boolean" use="optional" default="true">
<xs:annotation>
<xs:documentation>this is defaulted to 'true' as an indication to the post-process tool that it should encrypt this element.</xs:documentation>
</xs:annotation>
</xs:attribute>
<xs:attribute>
<xs:extension>
<xs:complexType>
<xs:complexType name="PasswordFromProfile">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="encryptContent" type="xs:boolean" use="optional" default="true">
        <xs:annotation>
          <xs:documentation>this is defaulted to 'true' as an indication to the post-process tool that it should encrypt this element.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

<xs:complexType name="PasswordSource">
  <xs:choice>
    <xs:element name="passwordFromLogon" type="Empty"/>
    <xs:element name="passwordFromUser" type="Empty"/>  
    <xs:element name="passwordFromProfile" type="PasswordFromProfile"/>
  </xs:choice>
</xs:complexType>

<xs:complexType name="TokenSource">
  <xs:choice>
    <xs:element name="passwordFromOtherToken" type="Empty">
      <xs:annotation>
        <xs:documentation>this will result in a prompt to user to obtain identity and otp from token</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:choice>
</xs:complexType>

<xs:complexType name="CertificateSource">
  <xs:choice>
    <xs:element name="certificateFromUser" type="Empty">
      <xs:annotation>
        <xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:choice>
</xs:complexType>
The client certificate to use during authentication is the one that the end-user selects from a list presented to them.

```xml
<xs:element name="certificateFromLogon" type="Empty"/>
<xs:documentation>The client certificate to use during authentication is the one the end-user used in order to logon to windows.</xs:documentation>
</xs:element>
<xs:element name="certificateFromProfile" type="ClientCertificate"/>
<xs:documentation>The client user certificate to use during authentication is indicated here.</xs:documentation>
</xs:element>
<xs:choice>
</xs:complexType>

<xs:complexType name="ExtendedInnerEapMethod">
<xs:sequence>
<xs:element name="methodName" type="xs:string"/>
<xs:element name="methodEapId" type="xs:unsignedInt"/>
<xs:element name="vendorId" type="xs:integer" default="0"/>
<xs:element name="AuthorName" type="xs:string"/>
<xs:element name="AuthorId" type="xs:unsignedInt"/>
<xs:any namespace="##any" processContents="lax" minOccurs="0"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="TunnelMethods">
<xs:sequence>
<xs:choice>
<xs:element name="validateServerCertificate" type="serverCertificateValidationParameters"/>
<xs:element name="doNotValidateServerCertificate" type="Empty"/>
</xs:choice>
<xs:element name="unprotectedIdentityPattern" type="IdentityPattern" minOccurs="0"/>
<xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
```
<xs:documentation>If the [username] and/or [domain] placeholders are used in the pattern then: if a client certificate is used for authentication then placeholder's values shall be obtained from the CN field of the client certificate. if the credentials are obtained from the end-user then these shall be obtained from the information the user enters. if the credentials are obtained from the operating system then these shall be obtained from the information the logon provides. Typical pattern: anonymous@[domain] for tunneled methods or [username]@[domain] for non-tunneled methods. If the credential source is this profile then the pattern would be the actual string to send as the username (no placeholders).</xs:documentation>
The EAP-FAST XML Schema

<x:annotation>
  <x:documentation>This is a reference to an OS pre-stored certificate.</x:documentation>
</x:annotation>
</xs:element>
</xs:choice>
</xs:complexType>

<x:complexType name="CertificateIdentifier">
  <xs:simpleContent>
    <xs:annotation>
      <xs:documentation>SHA 1 hash over the whole binary certificate in X509 format that uniquely identifies a certificate in the global list of trusted CAs for the machine (OS managed store in windows).</xs:documentation>
    </xs:annotation>
    <xs:extension base="NonEmptyString">
      <xs:attribute name="reference" type="xs:boolean">
        <xs:annotation>
          <xs:documentation>true means the element value is a file reference to a certificate in PEM format, the post-process tool will retrieve the certificate file, convert to a hash, populate the certificateId element, and set the reference to false to indicate this is the SHA1 hash over that certificate.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

<x:complexType name="Empty"/>

<x:simpleType name="NonEmptyString">
  <xs:restriction base="xs:string">
    <xs:minLength value="1"/>
  </xs:restriction>
</xs:simpleType>

<x:complexType name="ServerRuleFormat">
  <xs:simpleContent>
    <xs:extension base="NonEmptyString">
      <xs:attribute name="match" use="required">
        <xs:documentation></xs:documentation>
      </xs:attribute>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:simpleType>
  <xs:restriction base="xs:string">
    <xs:enumeration value="exactly"/>
    <xs:enumeration value="endsWith"/>
  </xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:extension>
</xs:simpleContent>
</xs:complexType>

<xs:complexType name="ServerValidationRules">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:annotation>
      <xs:documentation>
        Optional only when product allows user to trust server. In which case it allows a profile that has no server validations rules to start with and when a user validates an untrusted server the validation process still validates the server name.
      </xs:documentation>
    </xs:annotation>
    <xs:element name="matchSubject AlternativeName" type="ServerRuleFormat"/>
    <xs:annotation>
      <xs:documentation>DNSName: typically takes the form of a Fully Qualified Domain Name (FQDN)</xs:documentation>
    </xs:annotation>
    <xs:element name="matchSubject" type="ServerRuleFormat"/>
    <xs:annotation>
      <xs:documentation>Either Subject: CN (Common Name) - typically a simple ASCII string. Or Subject: DN (Domain Name) - a composite of a set of DC (Domain Component) attributes</xs:documentation>
    </xs:annotation>
  </xs:choice>
</xs:complexType>

<xs:complexType name="serverCertificateValidationParameters">
  <xs:sequence>
    <xs:choice>
      <xs:element name="serverNameValidationRules" type="ServerValidationRules"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
<xs:element name="anyServerName" type="Empty">
  <xs:annotation>
    <xs:documentation>the server name within the certificate will not be tested.</xs:documentation>
  </xs:annotation>
</xs:element>
</xs:choice>
</xs:element>
</xs:choice>
<xs:element name="validateChainWithSpecificCa">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="CertificateContainer"/>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:element name="validateChainWithAnyCaFromOs" type="Empty">
  <xs:annotation>
    <xs:documentation>the certificate chain will be trusted if it ends in a CA cert from the global CA cert store.</xs:documentation>
  </xs:annotation>
</xs:element>
</xs:choice>
<xs:element name="userValidatesUntrustedServerCertificate" type="xs:boolean">
  <xs:annotation>
    <xs:documentation>if the server certificate fails to validate then if this is true the end-user will be asked to validate the server. If they do so then appropriate trustedCaCerts will be remembered as well as the server name fields so it will be automatically trusted in the future.</xs:documentation>
  </xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
<xs:complexType name="ValidateWithSpecificPacs">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:annotation>
      <xs:documentation>This is optional because it allows the profile to indicate that we want the engine to validate the server PACs but that the PACs will be dynamically added by the end-user actions or via unauthenticated provisioning rather than being statically defined here in the profile.</xs:documentation>
    </xs:annotation>
  </xs:choice>
</xs:complexType>
Configuring Machine Authentication

You can enable machine authentication from the Advanced Security screen when you create a Group Policy Object.

The EAPHost notifies the EAP-FAST module that the current authentication is a machine authentication. Machine authentication is achieved by using one of the following:

- a machine PAC
- a machine certificate
- a machine password

The EAP-FAST module attempts to fetch the machine PAC first. If a machine PAC is unavailable, the EAP-FAST module attempts to fetch a machine certificate. If a machine certificate is unavailable, the EAP-FAST module attempts to fetch the machine password for the machine account in the Active Directory.

When the machine is authenticated with either a machine certificate or a machine password, the EAP-FAST module then requests the provisioning of a machine PAC for subsequent use. If neither a machine certificate nor a machine password is available, the EAP-FAST module requests a machine PAC during the next successful user authentication after a user has logged on. If an existing machine PAC is invalid or expired, the EAP-FAST module relies on this process to request a new machine PAC.

Because machine authentication is integrated with and supported by the Windows 802.1X supplicant, the EAP-FAST module is only responsible for authentication to gain network access. Additional network operations to support machine authentication, such as DHCP, machine-level GPO, and other related network services, are the responsibility of the operating system and the 802.1X supplicant.

Configuring Single Sign-On

SSO is supported by Microsoft Windows Vista in the following ways:

- Windows user credentials are passed to the EAP-FAST module through the EAPHost interface. The system does not prompt the user to provide additional credentials if the EAP-FAST module is configured to use Windows user credentials for network authentication and if the network profile is configured for single sign-on.
• Non-Windows network credentials are collected during the Microsoft Windows Vista logon process. The EAP-FAST module requests the logon module to prompt the user for these network credentials.

• If necessary, the EAP-FAST module is able to prompt the user for additional network credentials before the user logs in to Microsoft Windows Vista.

• If network credentials are stored in the configuration, the EAP-FAST module has access to these credentials before the user logs in to Microsoft Windows Vista.