Attribute-Based Access Control

Attributes are customized network objects for use in your configuration. You can define and use them in ASA configurations to filter traffic associated with one or more virtual machines in an VMware ESXi environment managed by VMware vCenter. Attributes allow you to define access control lists (ACLs) to assign policies to traffic from groups of virtual machines sharing one or more attributes. You assign attributes to virtual machines within the ESXi environment and configure an attribute agent, which connects to vCenter or a single ESXi host using HTTPS. The agent then requests and retrieves one or more bindings which correlate specific attributes to the primary IP address of a virtual machine.

Attribute-based access control is supported on all hardware platforms, and on all ASA Virtual platforms running on ESXi, KVM, or HyperV hypervisors. Attributes can only be retrieved from virtual machines running on an ESXi hypervisor.

Guidelines for Attribute-Based Network Objects

IPv6 Guidelines

- IPv6 addresses not supported by vCenter for host credentials.
- IPv6 is supported for virtual machine bindings where the primary IP address of the virtual machine is an IPv6 address.

Additional Guidelines and Limitations

- Multi-context mode is not supported. Attribute-based network objects are supported for single-mode context only.
- Attribute-based network objects support binding to the virtual machine's primary address only. Binding to multiple vNICs on a single virtual machine is not supported.
- Attribute-based network objects may only be configured for objects used for access groups. Network objects for other features (NAT, etc.) are not supported.

- Virtual machines must be running VMware Tools in order to report primary IP addresses to vCenter. The ASA is not notified of attribute changes unless vCenter knows the IP address of the virtual machine. This is a vCenter restriction.
- Attribute-based network objects are not supported in the Amazon Web Services (AWS) or Microsoft Azure public cloud environments.

Configure Attribute-Based Access Control

The following procedure provides a general sequence for implementing attribute-based access control on managed virtual machines in a VMware ESXi environment.

Procedure

Step 1 Assign custom attribute types and values to your managed virtual machines. See Configure Attributes for vCenter Virtual Machines, on page 2.

Step 2 Configure an attribute agent to connect to your vCenter Server or ESXi host. See Configure a VM Attribute Agent, on page 4.

Step 3 Configure attribute-based network objects needed for your deployment scheme. See Configure Attribute-Based Network Objects, on page 6.

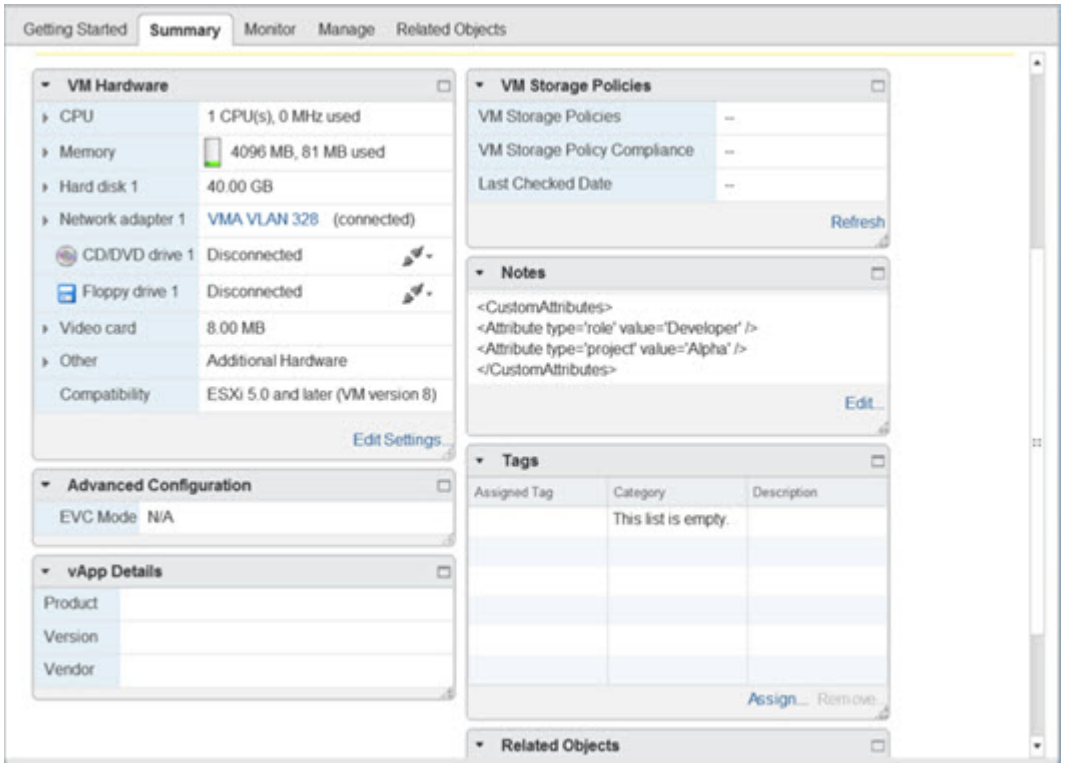
Step 4 Configure the access control lists and rules. See Configure Access Control Using Attribute-Based Network Objects, on page 7 .

Configure Attributes for vCenter Virtual Machines

You assign custom attribute types and values to virtual machines, and associate these attributes to network objects. You can then use these attribute-based network objects to apply ACLs to a set of virtual machines with common user-defined characteristics. For example, you could isolate developer build machines from test machines, or group virtual machines by project and/or location. For the ASA to monitor virtual machines using attributes, you need to make the attributes available to vCenter from the managed virtual machines. You do this by inserting a formatted text file into the Notes field, which is found on the Summary page of virtual machines in vCenter.

You can see the Notes field in the following figure.

Figure 1: Summary Tab of a Virtual Machine in vCenter



To specify custom attributes, you copy a properly formatted XML file into the Notes field for the virtual machine. The format of the file is:

```
<CustomAttributes>
<Attribute type='attribute-type' value='attribute-value'/>
...
</CustomAttributes>
```

A single virtual machine may have multiple attributes defined by repeating the second line above. Note that each line must identify a unique attribute type. If the same attribute type is defined with multiple attribute values, each binding update for that attribute type will overwrite the previous one.

For string attribute values, the value associated with the object definition must be an exact match to the value reported to vCenter by the virtual machine. For example, an attribute value *Build Machine* does not match the annotation value *build machine* on the virtual machine. A binding would not be added to the host-map for this attribute.

You can define multiple unique attribute types in a single file.

Procedure

Step 1 Select the virtual machine from your vCenter inventory.

Step 2 Click the **Summary** tab for the virtual machine.

Step 3 In the **Notes** field, click the **Edit** link.

Step 4 Paste the custom attributes text file into the **Edit Notes** box. The text file should follow the XML template format:

Example:

```
<CustomAttributes>
<Attribute type='attribute-type' value='attribute-value'/>
...
</CustomAttributes>
```

Step 5 Click **OK**.

Example

The following example shows a properly formatted XML text file that defines custom attributes for 'role' and 'project' that you can apply to virtual machines:

```
<CustomAttributes>
<Attribute type='role' value='Developer'/>
<Attribute type='project' value='Alpha'/>
</CustomAttributes>
```

Configure a VM Attribute Agent

You configure a VM attribute agent to communicate with vCenter or a single ESXi host. When you assign attributes to virtual machines within the VMware environment, the attribute agent sends a message to vCenter indicating which attributes have been configured, and vCenter responds with a binding update for every virtual machine where a matching attribute type is configured.

The VM attribute agent and vCenter exchange binding updates as follows:

- If the agent issues a request containing a new attribute type, vCenter responds with a binding update for every virtual machine where the attribute type is configured. After that point, vCenter only issues a new binding when an attribute value is added or changed.
- If a monitored attribute changes for one or more virtual machines, a binding update message is received. Each binding message is identified by the IP address of the virtual machine reporting the attribute value.
- If multiple attributes are being monitored by a single agent, a single binding update contains the current value of all monitored attributes for each virtual machine.
- If a specific attribute being monitored by the agent is not configured on a virtual machine, the binding will contain an empty attribute value for that virtual machine.
- If a virtual machine has not been configured with any monitored attributes, vCenter does not send a binding update.

Each attribute agent communicates with exactly one vCenter or ESXi host. A single ASA may have multiple attribute agents defined, each communicating with a different vCenter, or one or more communicating with the same vCenter.

Procedure

Step 1 Create the VM attribute agent to communicates with vCenter: **attribute source-group** *agent-name* **type** *agent-type*

Example:

```
hostname(config)# attribute source-group VMAgent type esxi
```

The *agent-name* argument specifies the VM attribute agent name. The *type* argument is the type of attribute agent.

Note
Currently ESXi is the only supported agent type.

Step 2 Configure your vCenter host credentials: **host** *ip-address* **username** *ESXi-username* **password** *ESXi-password*

Example:

```
hostname(config-attr)# host 10.122.202.217 user admin password Cisco123
```

Step 3 Configure keepalive settings for vCenter communication: **keepalive retry-interval** *interval* **retry-count** *count*

Example:

```
hostname(config-attr)# keepalive retry-timer 10 retry-count 3
```

The default keepalive timer values are 3 retries at 30-second intervals.

Step 4 Examine the VM attribute agent configuration: **show attribute source-group** *agent-name*

Example:

```
hostname(config-attr)# sh attribute source-group VMAgent

   Attribute agent VMAgent
   Agent type: ESXi
   Agent state: Inactive
   Connection state: Connected
   Host Address: 10.122.202.217
   Retry interval: 30 seconds
   Retry count: 3
```

The *Agent State* remains inactive until you configure a network object and specify attributes to associate with the object.

Step 5 Exit from the attribute configuration mode: **exit**

Example:

```
hostname(config-attr)# exit
```

Configure Attribute-Based Network Objects

Attribute-based network objects filter traffic according to attributes associated with one or more virtual machines in a VMware ESXi environment. You can define access control lists (ACLs) to assign policies to traffic from groups of virtual machines sharing one or more attributes.

For example, you can configure access rules that permit machines with an *engineering* attribute to access machines with a *eng_lab* attribute. A network admininstrator can add or remove engineering machines and lab servers while the security policy managed by the security administrator continues to work automatically without manual updates to the access rules.

Procedure

Step 1 Enable object group search: **object-group-search access-control**

Example:

```
hostname(config)# object-group-search access-control
```

You must enable object-group-search to configure attribute-based network objects.

Step 2 Create or edit an attribute-based network object using the object name: **object network** *object-id*

Example:

```
hostname(config)# object network dev
```

Step 3 Specify an agent, attribute type, and attribute value to associate with the object: **attribute** *agent-name attribute-type attribute-value*

Example:

```
hostname(config-network-object)# attribute VMAgent custom.role Developer
```

The *agent-name* specifies the VM attribute agent; see <XREF>. If you configure an attribute-based network object to use an attribute agent which has not been configured, a placeholder agent is automatically created with no credentials and default keepalive values. This agent remains in the "No credentials available" state until host credentials are supplied using the **host** subcommand

Together, the *attribute-type* and *attribute-value* pair define a unique attribute. The *attribute-type* is an arbitrary string and must include the **custom.** prefix. If you define the same attribute type more than once with multiple attribute values, the last value defined overwrites the previous one.

Examples

The following example creates the attribute-based network object *dev* for a development group, with a role of 'Developer'. The VM attribute agent communicates with vCenter and returns all of the virtual machine bindings that match the attribute *custom.role*:

```
hostname(config)# object network dev
hostname(config-network-object)# attribute VMAgent custom.role Developer
```

The following example creates the attribute-based network object *test* for a test group, with a role of 'Automation'. The VM attribute agent communicates with vCenter and returns all of the virtual machine bindings that match the attribute *custom.role*. Note that this is the same list of virtual machines as the previous example:

```
hostname(config)# object network test
hostname(config-network-object)# attribute VMAgent custom.role Automation
```

The following example creates the attribute-based network object *project* for a project group, with a role of 'Alpha'. The VM attribute agent communicates with vCenter and returns all of the virtual machine bindings that match the attribute *custom.project*. Note that some machines overlap more than one attribute:

```
hostname(config)# object network project
hostname(config-network-object)# attribute VMAgent custom.project Alpha
```

The following example shows a VM attribute agent in active status with pending attribute requests:

```
hostname(config-attr)# show attribute source-group VMAgent

   Attribute agent VMAgent
   Agent type: ESXi
   Agent state: Active
   Connection state: Connected
   Host Address: 10.122.202.217
   Retry interval: 30 seconds
   Retry count: 3
   Attribute requests pending:
       'custom.project'
       'custom.role'
```

Configure Access Control Using Attribute-Based Network Objects

You can use attribute-based network objects when you define access control lists (ACLs) to traffic from groups of virtual machines sharing one or more attributes. Access lists are made up of one or more access control entries (ACEs). An ACE is a single entry in an access list that specifies a permit or deny rule (to forward or drop the packet). Typically a permit or deny rule is applied to a protocol, to a source and destination IP address or network, and, optionally, to the source and destination ports.

When you use attribute-based network objects, you can replace source and/or destination IP addresses with these objects. As virtual machines are deployed, moved, or retired, attributes can be updated on the virtual machines while the assigned access control policies can remain in effect without configuration changes.

For complete information on all of the available options for ACLs, see Configure ACLs.

Procedure

Step 1 Create and configure an extended ACL entry (ACE) using attribute-based network objects: **access-list** *access_list_name* **extended** {**deny | permit**} *protocol_argument* **object** *source_object_name* **object** *dest_object_name*

Example:

```
hostname(config)# access-list lab-access extended permit ip object dev object test
```

Note
Repeat as needed for your policies.

The options are:

- *access_list_name*—The name of the new or existing ACL.
- Permit or Deny—The **deny** keyword denies or exempts a packet if the conditions are matched. The **permit** keyword permits or includes a packet if the conditions are matched.
- Protocol—The *protocol_argument* specifies the IP protocol:
  - *name* or *number*—Specifies the protocol name or number. Specify **ip** to apply to all protocols.
  - **object-group** *protocol_grp_id*—Specifies a protocol object group created using the **object-group protocol** command.
- Source Object—**object** specifies an attribute-based network object created using the **object network** command. The *source_object_name* specifies the object from which the packet is being sent.
- Destination Object—**object** specifies an attribute-based network object created using the **object network** command. The *dest_object_name* specifies the object to which the packet is being sent.

Step 2 Bind the ACL to an interface or apply it globally: **access-group** *access_list_name* {**in interface** *interface_name* | **global**}

Example:

```
hostname(config)# access-group lab-access in interface inside
```

For an interface-specific access group:

- Specify the extended ACL name. You can configure one **access-group** command per ACL type per interface.
- The **in** keyword applies the ACL to inbound traffic.
- Specify the **interface** name.

For a global access group, specify the **global** keyword to apply the extended ACL to the inbound direction of all interfaces.

Example

The following example shows how to apply an attribute-based extended ACL globally:

```
hostname(config)# access-list lab-access extended permit ip object dev object test
hostname(config)# access-group lab-access global
hostname(config)# show access-list
access-list cached ACL log flows: total 0, denied 0 (deny-flow-max 4096)
            alert-interval 300
access-list lab-access; 1 elements; name hash: 0x62b4790b
access-list lab-access line 1 extended permit ip object dev object test (hitcnt=0) 0x64a1be76

  access-list lab-access line 1 extended permit ip object dev(2) object test(3) (hitcnt=0)
 0x64a1be76
```

Monitoring Attribute-Based Network Objects

To monitor attribute-based network objects, enter the following commands:

- **show attribute host-map**

  Displays attribute bindings for a given attribute's agent, type, and value.

- **show attribute object-map**

  Displays the object-to-attribute bindings.

- **show attribute source-group**

  Displays the configured VM attribute agents.

Examples

The following example shows a map of the host-to-attribute bindings:

```
hostname# show attribute host-map /all
IP Address-Attribute Bindings Information

         Source/Attribute                          Value
=========================================================================
VMAgent.custom.project                         'Alpha'
     10.15.28.34
     10.15.28.32
     10.15.28.31
     10.15.28.33
VMAgent.custom.role                            'Automation'
     10.15.27.133
     10.15.27.135
     10.15.27.134
VMAgent.custom.role                            'Developer'
     10.15.28.34
     10.15.28.12
     10.15.28.31
```

```
     10.15.28.13
```

The following example shows the object-to-attribute bindings:

```
hostname# show attribute object-map /all
Network Object-Attribute Bindings Information

Object
         Source/Attribute                          Value
=========================================================================
dev
     VMAgent.custom.role                           'Developer'
test
     VMAgent.custom.role                           'Automation'
project
     VMAgent.custom.project                        'Alpha'
```

The following example shows the attribute agent configuration:

```
hostname# show attribute source-group
   Attribute agent VMAgent
   Agent type: ESXi
   Agent state: Active
   Connection state: Connected
   Host Address: 10.122.202.217
   Retry interval: 30 seconds
   Retry count: 3
   Attributes being monitored:
      'custom.role' (2)
```

History for Attribute-Based Access Control

Feature Name	Platform Releases	Description
Support for Attribute-Based Network Objects	9.7.(1)	You can now control network access using virtual machine attributes in addition to traditional network charateristics such as IP addresses, protocols, and ports. The virtual machines must be in a VMware ESXi environment. We introduced the following commands: **object network** *attribute* **attribute** *agent-name attribute-type attribute-value* **attribute source-group** *agent-name* **type** *agent-type* **host** *ip-address* **username** *ESXi-username* **password** *ESXi-password* **keepalive retry-interval** *interval* **retry-count** *count*
Remove support for VM attribute-based network objects from ASA 5506-X (all models), 5508-X, 5512-X, 5516-X.	9.10(1)	You can no longer use VM-attribute based network objects on the following platforms: ASA 5506-X (all models), 5508-X, 5512-X, 5516-X.