# IOS Context-Based Access Control (CBAC)

##### By [stretch](http://packetlife.net/users/stretch/) | Tuesday, March 10, 2009 at 12:00 p.m. UTC

Cisco's Context-Based Access Control (CBAC) is a component of the IOS firewall feature set. Similar to [reflexive ACLs](http://packetlife.net/blog/2008/nov/25/reflexive-access-lists/), CBAC enables dynamic modification of access lists to allow certain incoming flows by first inspecting and recording flows initiated from the protected internal network. However, whereas reflexive ACLs act solely on L2-L4 protocol attributes, CBAC is able to inspect all the way to the application layer, taking into consideration characteristics of a flow on a per-protocol basis (or context).

Rather than over-analyzing CBAC operation in this article, I offer a simple scenario: a router with exactly two interfaces (one internal and one external) placed between two networks, one trusted (internal) and one untrusted (external). Our goal is to configure the router to protect the trusted network (typically a LAN or enterprise network) from the untrusted network (in our example, the Internet).

From the conceptual illustration, we see that there are four logical points (marked in blue) at which the router can inspect traffic:

1. Inbound on the internal interface
2. Outbound on the external interface
3. Inbound on the external interface
4. Outbound on the internal interface

While we can deploy independent, static ACLs at one, some, or all of these points simultaneously, CBAC is configured and operates per interface, dynamically modifying ACL entries facing one direction based on the traffic it sees flowing in the opposite direction.

For example, let's assume we first want to allow by default all traffic traversing the router from the internal LAN (192.168.0.0/24) toward the Internet (for our purposes, 10.0.0.0/8). This is already the case, as the router will of course forward all routable traffic when no access restrictions have been applied. We also want to deny by default all traffic flowing from the Internet toward the LAN; to accomplish this we can use a simple ACL to deny all IP traffic. We'll apply it to the external interface in the inbound direction (versus the internal interface outbound) so that the router itself is protected from untrusted traffic as well.

R1(config)# ip access-list extended DENY\_ALL

R1(config-ext-nacl)# deny ip any any

R1(config-ext-nacl)# exit

R1(config)# interface f0/0

R1(config-if)# ip access-group DENY\_ALL in

Note that an extended ACL is used here; more on that in a bit.

At this point, traffic can flow uninhibited from our trusted network to the untrusted network, but is completely blocked in the opposite direction. Not great if you favor bidirectional communication.

For instance, assume we now want to allow web access initiated from the internal network to return. One option would be to configure a more granular ACL in place of DENY\_ALL, but that would not provide stateful tracking and may open a security hole. Another option would be to implement a reflexive ACL, but that would provide only limited state tracking. A more powerful solution is CBAC.

CBAC allows us to define an inspection rule for each protocol we want to monitor. In this case, we want to track HTTP and HTTPS sessions, so we create the following rule named "Web":

R1(config)# ip inspect name Web http

R1(config)# ip inspect name Web https

There are additional options per protocol, but for now we'll accept their defaults. Next we need to apply our inspection rule to an interface and in a particular direction. We apply the rule outbound on the external interface because:

1. We want to inspect traffic originating from the trusted network, and
2. We want to dynamically adjust the ACL restricting traffic inbound on the external interface.

This is done with the ip inspect command at interface configuration:

R1(config)# interface f0/0

R1(config-if)# ip inspect Web out

The CBAC configuration can be verified with show ip inspect all:

R1# show ip inspect all

Session audit trail is enabled

Session alert is enabled

one-minute (sampling period) thresholds are [400:500] connections

max-incomplete sessions thresholds are [400:500]

max-incomplete tcp connections per host is 50. Block-time 0 minute.

tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec

tcp idle-time is 3600 sec -- udp idle-time is 30 sec

dns-timeout is 5 sec

Inspection Rule Configuration

Inspection name WEB

http alert is on audit-trail is on timeout 3600

https alert is on audit-trail is on timeout 3600

Inspection name Web

http alert is on audit-trail is on timeout 3600

https alert is on audit-trail is on timeout 3600

Interface Configuration

Interface FastEthernet0/0

Inbound inspection rule is not set

Outgoing inspection rule is Web

http alert is on audit-trail is on timeout 3600

https alert is on audit-trail is on timeout 3600

Inbound access list is DENY\_ALL

Outgoing access list is not set

We've now enabled CBAC in one direction for HTTP and HTTPS traffic on this interface. Outbound traffic (going toward the Internet) is tracked as CBAC sessions, and the DENY\_ALL extended ACL restricting inbound traffic is automatically modified to accommodate legitimate return traffic. We can see CBAC in action when a host and the trusted network initiates an HTTP connection to a web server on the Internet:

R1# show ip inspect sessions

Established Sessions

Session 66E4E40C (192.168.0.2:12078)=>(10.0.0.2:80) http SIS\_OPEN

From the output of show ip inspect sessions we can see that the trusted host 192.168.0.2 has opened an HTTP connection to 10.0.0.2. CBAC will continue to track this session until it is closed by either end, or the configured idle timer is reached.

When the session was initiated, CBAC automatically created an entry at the top of the DENY\_ALL ACL in preparation for the legitimate return traffic.

We can enable audit trails to generate syslog messages for each CBAC session creation and deletion:

R1(config)# ip inspect audit-trail

R1(config)#

%FW-6-SESS\_AUDIT\_TRAIL\_START: Start http session: initiator (192.168.0.2:12078) --

responder (10.0.0.2:80)

Or, for even more verbosity, we can debug CBAC object creation:

R1# debug ip inspect object-creation

INSPECT Object Creations debugging is on

R1#

CBAC\* OBJ\_CREATE: Pak 6621F7A0 sis 66E4E154 initiator\_addr (192.168.0.2:12078) responder\_addr

(10.0.0.2:80) initiator\_alt\_addr (192.168.0.2:12078) responder\_alt\_addr (10.0.0.2:80)

CBAC OBJ-CREATE: sid 66E684B0 acl DENY\_ALL Prot: tcp

Src 10.0.0.2 Port [80:80]

Dst 192.168.0.2 Port [12078:12078]

CBAC OBJ\_CREATE: create host entry 66E568DC addr 10.0.0.2 bucket 8 (vrf 0:0) insp\_cb 0x66B61C0C

Of course there's far more to CBAC than we've covered here, but hopefully this example provides a decent illustration of the concept.

# Cisco Identity Services Engine – Part 1 – Overview

[Security](http://thenetworksurgeon.com/category/security/)[BYOD](http://thenetworksurgeon.com/tag/byod/), [Cisco](http://thenetworksurgeon.com/tag/cisco/), [Identity Services Engine](http://thenetworksurgeon.com/tag/identity-services-engine/), [ISE](http://thenetworksurgeon.com/tag/ise/), [ISE Licensing](http://thenetworksurgeon.com/tag/ise-licensing/), [Networking](http://thenetworksurgeon.com/tag/networking/), [Overview](http://thenetworksurgeon.com/tag/overview/), [Security](http://thenetworksurgeon.com/tag/security/)

## What is the Cisco Identity Services Engine?

Today’s enterprise network is rapidly changing, especially when it comes to employee mobility. Employees are no longer tethered to desktop workstations, but instead access enterprise resources via a variety of devices: tablets, smartphones, and personal laptops, just to name a few. Being able to access resources from anywhere greatly increases productivity, but it also increases the probability of data breaches and security threats because you may not control the security posture of devices accessing the network. Keeping track of all devices accessing the network is a huge task in itself, and as the need for more access arises, the more unsustainable it becomes to manage.

The Cisco Identity Services Engine (ISE) is an identity-based network access control and policy enforcement system. ISE allows a network administrator to centrally control access policies for wired and wireless endpoints based on information gathered via RADIUS messages passed between the device and the ISE node, also known as profiling. The profiling database is updated on a regular basis to keep up with the latest and greatest devices so there are no gaps in device visibility.

Essentially, ISE attaches an identity to a device based on user, function, or other attributes to provide policy enforcement and security compliance before the device is authorized to access the network. Based on the results from a variety of variables, an endpoint can be allowed onto the network with a specific set of access rules applied to the interface it is connected to, else it can be completely denied or given guest access based on your specific company guidelines.

Let’s analogize LOTR-style for clarification: ISE is Gandalf, and the end-user device is the pursuing Balrog. I think you know where this is going.

YOU SHALL NOT PASS!

ISE an automated policy enforcement engine that takes care of the mundane day-to-day tasks like BYOD device onboarding, guest onboarding, switchport VLAN changes for end-users, access list management, and many others, so a network administrator can focus on other important tasks (and cool projects!).

## ISE Basics

The ISE platform is typically a distributed deployment of nodes made up of three different personas: Policy Administration Node (PAN), Monitoring and Troubleshooting Node (MnT), and Policy Services Node (PSN). All three roles are required for ISE to function.

### Policy Administration Node (PAN)

The PAN persona is the interface an administrator logs into in order to configure policies. It is the control center of the deployment. This node allows an administrator to make changes to the entire ISE topology, and those changes are pushed out from the admin node to the Policy Services Nodes (PSN).

### Policy Services Node (PSN)

The PSN persona is where policy decisions are made. These are the nodes where network enforcement devices send all network messaging to; RADIUS messaging is an example of what is sent to the PSNs. The messages are processed and the PSN gives the go/no-go for access to the network.

### Monitoring and Troubleshooting Node (MnT)

The MnT persona is where logging occurs and reports are generated. All logs are sent to this node and it sorts through them so it can assemble them in a legible format. It is also used to generate various reports so you can make management happy with pretty pictures and numbers (\*wink wink\*) as well as notify you of any alarms for ISE.

### How ISE Works

Now that you know what each persona does, let’s take a look at how everything fits together as a complete system. The diagram shows a logical representation of ISE, because the personas may be distributed across many different appliances. Familiarize yourself with the figure below, and I will explain what each piece is doing:

ISE Communication Model

The figure above is from the [Cisco Trustsec How-To Guide: ISE Deployment Types and Guidelines](http://www.cisco.com/en/US/solutions/collateral/ns340/ns414/ns742/ns744/docs/howto_50_ise_deployment_tg.pdf). If you’re considering deploying ISE, I really recommend reading all [ISE Design Guides](http://www.cisco.com/en/US/products/ps11640/products_implementation_design_guides_list.html) before you plan your implementation.

1. Communication starts with the endpoint. This could be a laptop, smartphone, tablet, security camera, videoconferencing system — anything that requires network access.
2. The client must connect through a network access device — a switch, a wireless LAN controller, or a VPN concentrator — in order to gain access to the network. This is where enforcement of all policies takes place.
3. The endpoint is asked for authentication via an 802.1X request, and that request is sent to the Policy Services Node.
4. At this point, the PSN has already been given a specific configuration from the Admin node. The PSN will process the credentials (it may need to query an external database for this; LDAP or Active Directory, for example), and based on the configuration set the PSN will make an authorization decision.
5. The PSN sends the decision back to the network access device so it can enforce the decision. The network access device is sent specific actions to take for the session. Many actions can be taken at this point depending on the policy, but a few common features are dynamic access lists, change of authorization (to switch VLANs, for example), and Security Group Tags (part of the Cisco TrustSec solution).
6. Now the client can access specific resources based on what the PSN has sent back as a rule set. Alternatively, the client can be redirected to the guest login page or completely denied access to the network.
7. All of these messages passed back and forth are all logged to the monitoring node, where they can be viewed from the admin node in an organized format.

This is definitely a complex beast with a lot of moving parts, but as long as you keep the fundamentals in mind and break it down into different parts, it’s not too tough to implement and troubleshoot. The most time-consuming part of a deployment is figuring out your policies for authorization. Once you have standard policies across the board, enforcing those policies is a breeze with ISE. I will go into policies in my next post, but let’s move onto the last topic for the overview: Deployment Topologies and Licensing.

## Physical Deployment Examples

What I’m about to say here is probably the most important part of a deployment: DO NOT TRY TO SAVE MONEY BY GIVING UP HIGH AVAILABILITY! These nodes control access to your entire network. If these nodes go down, you might as well have a total network failure because nobody will get authenticated or authorized. Design ISE with as much high availability as you can afford. The only time a standalone deployment is acceptable is if you are doing a very small proof-of-concept which does not effect production end-users.

The other important part of a deployment is the hardware chosen to implement ISE on. Now, Cisco does offer an ESX/ESXi option, however, I don’t recommend that for a few reasons. First and foremost, the appliance option is tested and rated to scale to a certain number of endpoints. If you use the ESX/ESXi option, you are losing a little bit of that predictability. I said it before and I will say it again, these nodes control access to your entire network, so if you have unpredictable performance, then you will have unpredictable issues. The other thing I don’t like about an ESX/ESXi option is troubleshooting. If you do have an issue with ISE, you really want it resolved quickly. If you’re using the VM deployment and something goes wrong, you have to open tickets with Cisco, your server manufacturer, VMWare, and anything else that may be tied to your deployment. That’s not incredibly efficient, and you’re likely to run into a lot of vendor finger-pointing before you finally get the issue resolved. If you go with the Cisco appliance route, you open a ticket with Cisco, and that’s it! Smartnet covers the software and the hardware, so it makes the issue resolution process much simpler. I will say it one last time: These nodes control access to your entire network!

With that said, let’s get into the actual deployment options:

* Standalone
  + One node running all three personas (PAN, MnT, PSN).
  + No redundancy.
  + Limited to a maximum of 2000 endpoints, regardless of hardware type.
* Two-Node Deployment
  + Two nodes running all three personas (PAN, MnT, PSN).
  + Simple redundancy – one node is assigned the primary admin role and secondary monitoring role, and the other node is assigned the secondary admin role and primary monitoring role. They both run the policy services node, and network access devices are configured to use both PSNs.
  + Still limited to a maximum of 2000 endpoints.
* Mid-sized Deployment – Separate Policy Services Nodes
  + Two nodes taking on the admin and monitoring persona, but no policy service persona.
  + Policy service nodes are standalone boxes. You can have up to 5 PSNs in a deployment where the PAN and MnT are collocated.
  + Limited to a maximum of 10000 endpoints regardless of hardware capacity.
* Large Deployment
  + Every persona has its own dedicated node. You have a primary admin node, a secondary admin node, a primary monitoring node, a secondary monitoring node, and up to 40 policy services nodes. This deployment is so large that you will likely need load balancers to serve virtual IPs with a group of PSNs sitting behind it.
  + The endpoint capacity is based on the hardware of nodes. Prior to ISE version 1.2, the maximum capacity was 100000 endpoints. In version 1.2, that limitation was increased to 250000 endpoints.

If you’re looking for the performance metrics for each appliance, take a look at the design guides I linked to above. There are plenty of charts and other goodies to explain everything.

Got it? Good! Onto licensing this bad boy!

## Licensing

ISE is offered in a few different flavors of licensing: functionality-based or deployment-based.

Functionality-based is the “full-steam ahead” type of licensing where all network access devices are supported and feature-sets are licensed. You can choose from the Base License or Base + Advanced License. The Base License is for deployments that only need to authenticate and authorize users and devices, provision guest users, access reporting features, and monitor and troubleshoot access to the network. The base license is perpetual (it has no term subscription limit). The Advanced License expands on the base license and enables organizations to make more advanced decisions (it has all of the cool features that you really want in a deployment). The features include device onboarding and provisioning, device profiling and feed service, posture services, mobile device management integration, and security group access capabilities. This license is term-based with a choice of 3- or 5-year term subscriptions. The base license is a prerequisite for the advanced license.

Deployment-based licensing is the slow, phased approach to deploying ISE. This type of licensing allows you to start with wireless endpoints only and expand to wired and VPN later when your organization is ready. Due to the complexity of ISE, I recommend using the phased approach and really getting to know the product before rolling it out to the entirety of the network. The **Wireless License**includes everything that the base + advanced license does, but it only applies to wireless network access devices. The wireless license is term-based with a choice of 3- or 5-year term subscriptions. This license typically satisfies most BYOD (Bring Your Own Device) policies management may be asking for. Once ISE has been proven effective on the wireless front, it’s typically pretty easy to justify rolling it out to wired and VPN devices as well using the Wireless Upgrade License. The wireless upgrade license is the same as wireless, but it expands the functionality of ISE to wired and VPN network access devices. It is also licensed on 3- or 5-year terms.

Along with the term length, each license has an endpoint limit (100, 250, 500, 1000, 1500, and so on). Keep in mind, this is not total endpoints, but simultaneously authorized endpoints. If an endpoint isn’t authorized, it doesn’t increment the license count. If an endpoint is authorized and then leaves the network, the license count decrements because it is de-authorized.

SO! Hopefully this information was helpful. It’s a lot to take in, and there are many nooks and crannies to navigate in order to have a successful ISE deployment. Keep an eye out for Part 2: Wireless ISE Deployment, where I’ll get into the technical details of the deployment type most organizations decide to start with.

As always, leave a comment below or send me an email at [garret@thenetworksurgeon.com](mailto:garret@thenetworksurgeon.com) with any questions!