CREATING AND APPLYING IP ACLS ON A DELL™ POWERCONNECT™ 62xx SERIES

THIS APPLICATION NOTE RELATES TO THE DELL POWERCONNECT 62XX SERIES

CONTENTS

ABSTRACT

Layer 3 switches are an increasingly essential part of medium to large networks. The purpose of a layer 3 switch is to segment traffic between multiple networks but still provide routing between them. At the same time, they still pass high speed Ethernet traffic. There are many instances when an administrator would need to restrict access between networks. This application note will explain how to create and apply access control lists (ACLs) to restrict the flow of traffic between VLANs and network segments.

Note: The information in this document is written with firmware version 2.0.0.12. Subsequent releases may have altered the functionality of IP based ACLs. Be sure to refer to the release notes for the firmware you are currently using.

HOW ACLS WORK

ACLs are used to provide a series of checks on each packet received. The ACL, once saved, is applied to an interface on the switch. ACLs applied to an interface on a PowerConnect 62xx will only affect inbound traffic–that is traffic entering the interface. Traffic leaving the interface is not affected by an applied ACL.

An ACL consists of at least one Access Control Entry (ACE) which is a line that specifies a logical operation on how to handle traffic based on specific criteria. When traffic with an ACL applied to it, the traffic is compared against each ACE in order.

ACEs have two basic functions: permit and deny. When a packet is compared against an ACE, and it matches the criteria, a permit statement will allow the traffic to pass. Matched criteria on a deny statement will drop the packet. Something that does not match the criteria of an ACE is tested against the next ACE in the sequence. If the packet does not match the criteria of any ACE in the list, it will be dropped.

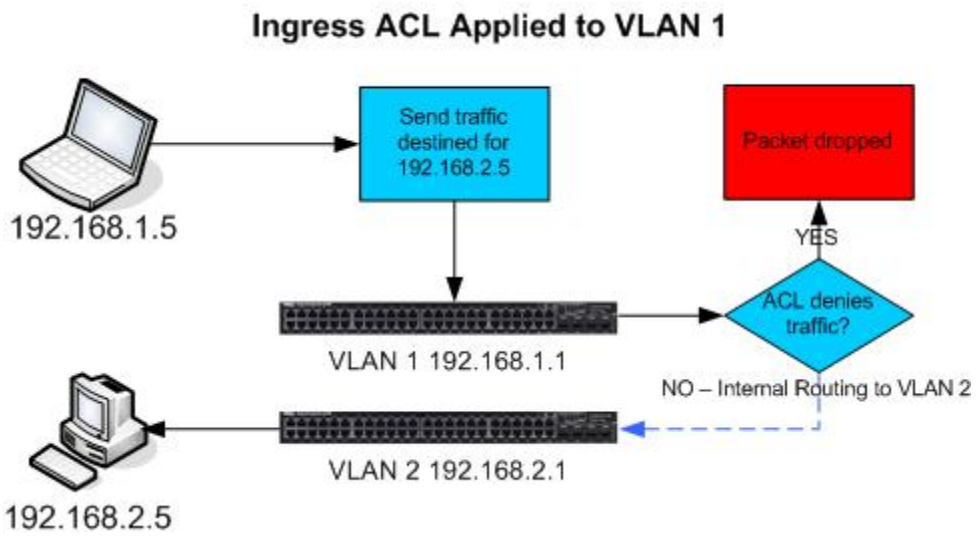
Note: ACLs in firmware revision 1.0.0.27 block ICMP(ping) traffic even if IP traffic is permitted. To allow ICMP, a specific rule has to be applied with ICMP as the traffic type. This is important to keep in mind when troubleshooting an ACL. In subsequent releases of the firmware, ICMP traffic is treated as a subset of IP traffic.

On a PowerConnect 62xx series switch, ACLs can only be applied to a port, a LAG group, a VLAN interface or as a global command. For the purposes of this document, we will refer strictly to ACLs applied to VLAN interfaces, unless otherwise noted.

APPLICATION AND BEST PRACTICES

Designing an effective security scheme with ACLs can be difficult, so there are some important things to consider before creating and deploying your ACL.

First, it should be understood that the ACL will only apply to traffic received from a port or exiting a port. On a PowerConnect 62xx switch, traffic that enters a VLAN interface via an internal source will be overlooked by an ingress ACL. This includes traffic originating from the switch interface and traffic that has been routed from another VLAN. Similarly, traffic exiting one VLAN interface only to enter another interface within the switch is also overlooked, as only traffic exiting the switch has ACLs applied to it.



In the above diagram, the ingress ACL applied on VLAN 1 only affects traffic received from hosts connected to VLAN 1. If traffic is flowing from VLAN 2 to VLAN 1, the ACL applied to VLAN 1 will have no effect, as it is not applicable to traffic that is entering the interface from an internal source (such as another VLAN). Traffic does not have to be routed for the ACL to take effect. Traffic from VLAN 1 that is destined for something on the same subnet is still subjected to the ACL.

In order to minimize unnecessary traffic across a network, it is best to apply an inbound ACL as close to the source IP as possible and an outbound ACL closest to the destination IP. One should consider this carefully, as it is generally not recommended to carry packets across a network only to have them dropped, when they could have been dropped on the first hop.

Another consideration is that each ACL on a PowerConnect 62xx switch supports a maximum of 10 ACEs. If more than 10 rules are needed for a particular interface, another ACL has to be created. Multiple ACLs can be applied to an interface, and they will be processed in the sequence they are issued. The implicit deny will not take place until all applicable ACLs on the interface have been run.

Note: Firmware revision 1.0.0.27 does not support multiple ACLs on a single interface, and as such the maximum amount of ACEs allowed is 10. This has been updated to 13 ACEs in subsequent releases. While multiple ACLs can be applied to an interface, the maximum amount of ACEs on an interface cannot exceed 13.

One final consideration should be in the ordering of ACEs. It makes sense that you would want to put the most general statements, such as a blanket permit statement, towards the end of your ACL. You should also consider that each rule is processed in order for every packet. With this in mind, it is recommended that the first rules should apply to the most common traffic in order to minimize switch the amount of switch resources used.

UNDERSTANDING INGRESS VS. EGRESS

When an ACL is applied to an interface, it can be followed with the option of "in" or "out" denoting which direction it applies to traffic. By default, all applied ACLs will default to the ingress setting.

An ingress ACL applies to all traffic that enters the interface. Ingress ACLs can be applied to an ethernet port, a port-channel, a VLAN or to the entire switch. Ingress ACLs will only take effect when traffic enters the interface from an external source. It will not address traffic that originated in the switch, or has arrived from another interface (for example, being routed between VLANs).

An egress ACL can only be applied to a port-channel or to a single ethernet port. Any traffic that leaves the switch through this interface is checked against the egress ACL. All ACEs in an ACL used for egress traffic must only have arguments for the destination IP and port numbers. The source IPs and ports cannot be defined for egress ACLs. It should also be understood that an egress ACL cannot match all the traffic types that an ingress ACL can.

ACCEPTABLE ACE MATCH TYPES; INGRESS VS. EGRESS

RULE TYPE	EGRESS	INGRESS
Destination IP	X	X
IP Protocol	X	X
L4 Source port	X	X
L4 Destination port	X	X
Source IP		X
IP Precedence		X
IP DSCP		X

ACL limitations are different for an egress ACL than they are for an ingress. The ACL itself will not be set specifically for an outbound or inbound traffic-that is only determined when it is applied. However, you should make sure your ACL is of the proper criteria before making it outbound. Below are the limitations for IP based ACLs.

ACL LIMITATIONS

	GLOBAL	VLAN	ETHERNET
Total ACLs	100	12	12
Total ACEs	1000	13	13
Outbound ACEs	N/A	0	3

COMMAND LINE SYNTAX

This section is for an overview of the common command line functions. For a detailed look at all of the commands available as well as creation and management via the web interface, please consult the Users Guide, located at http://support.dell.com/support/edocs/network/PC62xx/en/index.htm

The actual construction of an ACL from the command line is done by entering each ACE one at a time. The order they are entered will be the way the switch processes each rule. It should be noted that there is no way to edit or remove an ACE once it is entered without removing the entire access list. Careful consideration will need to be used before creating an ACL.

An ACE can be broken down into 4 basic parts: action, traffic type, source and destination. Consider this entry:

```
console(config)# access-list example deny tcp 192.168.1.0 0.0.0.255
eq 23 192.168.2.0 0.0.0.255
```

In this example, after the list is named (example), there is the action (deny) and then the traffic type (tcp), followed by the source (192.168.1.x port 23) and finally the destination (192.168.2.x). Every IP-based ACE will follow this basic structure.

NOTE: When defining a network or host, the wildcard mask should be used in place of a subnet mask. The following table compares class C subnet masks to wildcard masks. To extend to a Class B or Class A subnet, use the same table but for the appropriate octet(s).

SUBNET MASK	CIDR ANNOTATION	WILDCARD MASK
255.255.255.0	/24	0.0.0.255
255.255.255.128	/25	0.0.0.127
255.255.255.192	/26	0.0.0.63
255.255.255.224	/27	0.0.0.31
255.255.255.240	/28	0.0.0.15
255.255.255.248	/29	0.0.0.7
255.255.255.252	/30	0.0.0.3
255.255.255.254	/31	—
255.255.255.255	/32	0.0.0.0

To add another line to an ACL, just enter the command with the same name and with a new ACE defined. There is no way in the command line to remove a single ACE from an access-list. The entire ACL needs to be removed and re-created. If a single line needs to be removed or adjusted without removing the entire ACL, this can only be accomplished by using the web-based interface.

```
console(config)# access-list example permit ip any any
```

You'll see that this still has the four basic components described above, but in the most basic form. (action) permit (traffic type) ip (source) any (destination) any.

After creating a VLAN, verify that it has the rules needed to apply. This can be done with the following command:

```
console# show ip access-lists example

IP ACL Name: example

Rule Number: 1
Action..................................... deny
Match All.................................. FALSE
Protocol................................... 6(tcp)
Source IP Address.......................... 192.168.1.0
Source IP Mask............................. 0.0.0.255
Source L4 Port Keyword..................... 23(telnet)
Destination IP Address..................... 192.168.2.0
Destination IP Mask........................ 0.0.0.255

Rule Number: 2
Action..................................... permit
Match All.................................. TRUE
```

Note that a "deny ip any any" statement is applied after ACLs have been processed, but it is never displayed. If the ACL looks correct, it can be applied to the desired interface with the following commands (note that the argument "in" is a default and does not need to be added to specify the rule for ingress traffic):

```
console#config
console(config)# interface vlan 1
console(config-if-vlan1)# ip access-group example in
```

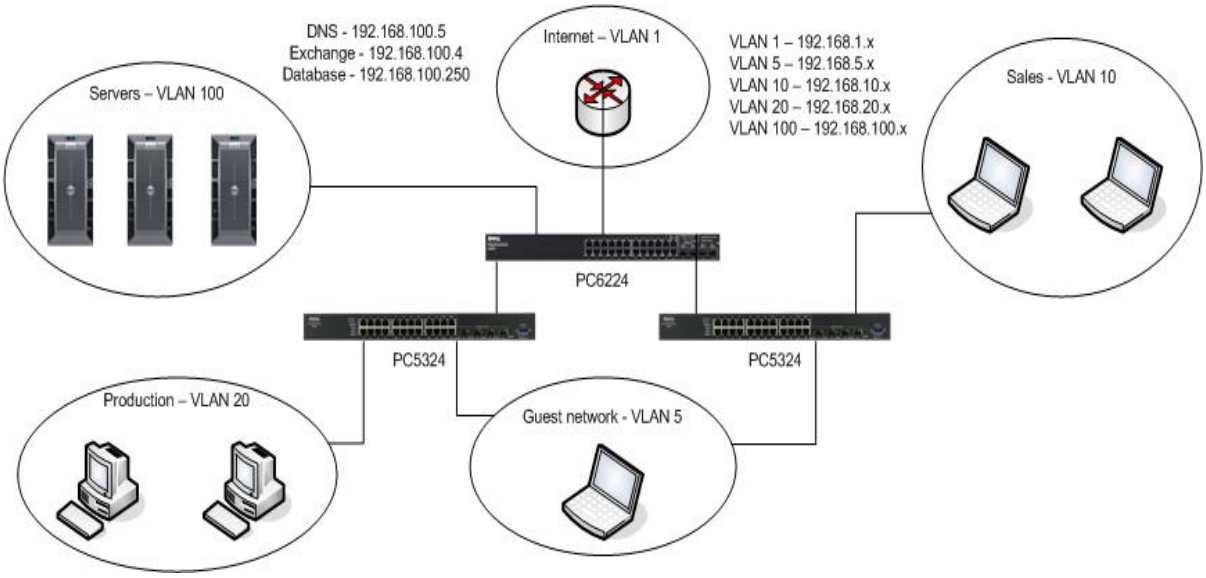
Finally, verify that the access list is properly applied with the following command:

```
console# show ip access-lists

Current number of ACLs: 1 Maximum number of ACLs: 100

       ACL Name        Rules   Interface(s)   Vlan(s)
       --------        ------  ------------   -------
       Example         2       vlan           1
```

APPLICABLE NETWORK SCENARIOS



In the diagram depicted above, each portion of the network is segmented into separate categories. As different VLANs need to communicate, routing would be enabled on the PowerConnect 6224. However, when routing is enabled, all segments have access to every other segment. In order to restrict access to only what is necessary, we need to apply ACLs at key points.

Using the example above, we will make a couple simple rules to restrict the flow of traffic.

EXAMPLE 1 – PRODUCTION VLAN ACCESS

In this example, we will state that the production team needs full access to all of the servers on VLAN 100, but it should not have any access to the guest or sales VLANs or to the internet.

To achieve the desired security, we will need to create an ACL on the 6224 and apply it to the VLAN 20 interface. All traffic from the production VLAN will arrive at the 6224 from this interface and will be processed by this ACL.

The step by step process is as follows:

1. DEFINE THE PERMITTED TRAFFIC.

In this case, all that needs to be allowed is from the 192.168.20.x subnet and the only access it needs is to the 192.168.100.x network.

```
console> enable
console# configure
console(config)# access-list Production permit ip 192.168.20.0
0.0.0.255 192.168.100.0 0.0.0.255
```

If the 192.168.20.x network were accessed from any other switch beyond the PowerConnect 6224 it would also be necessary to permit access to the 192.168.20.0 network. As traffic destined to VLAN 20 from other hosts on VLAN 20 will stay entirely on the PowerConnect 5324 without crossing the PC6224, a rule is not required for this.

With this in mind, there are no other ACEs that need to be created for this to work. All other traffic from VLAN 20 that will reach the 6224 is going to go directly to the server VLAN. The single ACE is sufficient to grant this access. As no other traffic needs to be explicitly allowed, the implicit deny will resolve any traffic headed to another destination.

2. APPLY THE ACL TO VLAN 20

```
console(config)# interface vlan 20
console(config-if-vlan20)# ip access-group Production
console(config-if-vlan20)# exit
console(config)# exit
console# copy running-config startup-config
```

After this is applied, production systems will now have access to the servers, but will not be able to reach any other network. It should be noted that by applying this ACL on VLAN 20, access to VLAN 20 from other networks is almost completely removed as well. It's possible that traffic will be able to enter VLAN 20, but return traffic to anywhere except VLAN 100 will not be permitted.

EXAMPLE 2 – DEFINING ACCESS FOR THE SALES VLAN

The sales team needs access to the Exchange, DNS and Database servers and to the internet. Every other internal resource should be forbidden from VLAN 10. It does not need access to the entire Server VLAN.

1. DEFINE PERMITTED TRAFFIC

```
console(config)# access-list Sales permit ip 192.168.10.0 0.0.0.255
192.168.100.5 0.0.0.0
console(config)# access-list Sales permit ip 192.168.10.0 0.0.0.255
192.168.100.4 0.0.0.0
console(config)# access-list Sales permit ip 192.168.10.0 0.0.0.255
192.168.100.250 0.0.0.0
```

With these commands, we have defined the Sales ACL and given permission to the specific servers. Users on the Sales VLAN still need to have access to the internet, but it would be inefficient and unwise to make an ACE for every external subnet. This would be better served by adding a "permit ip any any" line. However, before doing this we have to specify what VLANs are not allowed.

2. DEFINE BLOCKED TRAFFIC

```
console(config)# access-list Sales deny ip 192.168.10.0 0.0.0.255
192.168.5.0 0.0.0.255
console(config)# access-list Sales deny ip 192.168.10.0 0.0.0.255
192.168.20.0 0.0.0.255
console(config)# access-list Sales deny ip 192.168.10.0 0.0.0.255
192.168.100.0 0.0.0.255
```

With these statements we have blocked access to the guest VLAN, production VLAN and server VLAN. Because we already permitted access to specific hosts on VLAN 100 with an earlier ACE, the traffic will be forwarded before the deny ACEs can be applied. With this in mind, we need to allow access to the internet.

3. PERMIT ACCESS TO UNDEFINED HOSTS

```
console(config)# access-list Sales permit ip any any
```

The “permit ip any any” ACE allows all traffic to pass except for those that have already been denied by a previous ACE.

Note: In all of the ACEs entered up to this point, we did not address VLAN 1 as either explicitly permitted or denied. The reason for this is that IP traffic entering the switch will usually not be destined for VLAN 1. It will be sent to the local VLAN interface(the default gateway) and then routed internally to VLAN 1. It will still be scanned by the ACL for it’s final destination–but normal circumstances will not point it to VLAN 1.

Access to VLAN 1 can be specifically denied for security reasons, but as there should only be two hosts on this network(the WAN router and the PC6224), access to those resources are usually controlled in another manner. Letting traffic pass to this VLAN allows for easier network troubleshooting. If the administrator so wishes to restrict VLAN 1, a deny statement can be included before the permit ip any any statement. This ACE will only restrict traffic that has an end destination in VLAN 1.

4. APPLY ACL TO INTERFACE

```
console(config)# interface vlan 10
console(config-if-vlan10)# ip access-group Sales
console(config-if-vlan10)# exit
console(config)# exit
console# copy running-config startup-config
```

EXAMPLE 3 – RESTRICTING GUEST ACCESS

The guest network in this example should have no access to any other VLAN, except for the internet and a limited access to a couple of servers on VLAN 100.

For the purposes of getting online, guest users will need access to the DNS and DHCP servers, but they will only be granted access to those specific ports-not to the rest of the server. In this example, the DHCP server is on the same IP as the Exchange server.

Guests should not be granted full access to the Exchange server–they should just receive access to DHCP. DHCP communicates on TCP port 67. To do this, we’ll need to grant access for this port, but deny access to the rest of the host.

1. DEFINE PERMITTED TRAFFIC

```
console(config)# access-list Guest permit ip 192.168.5.0 0.0.0.255
192.168.100.5 0.0.0.0
console(config)# access-list Guest permit tcp 192.168.5.0 0.0.0.255 eq
67 192.168.100.4 0.0.0.0
```

In this example, we need to define allowed traffic that is specific to a tcp port. Note that the traffic-type has changed from ip to tcp. When the ACE is addressing specific layer 4 traffic, that is defined on either the source or destination side with the “eq #” option.

2. DEFINE BLOCKED TRAFFIC

```
console(config)# access-list Guest deny ip 192.168.5.0 0.0.0.255
192.168.1.0 0.0.0.255
console(config)# access-list Guest deny ip 192.168.5.0 0.0.0.255
192.168.20.0 0.0.0.255
console(config)# access-list Guest deny ip 192.168.5.0 0.0.0.255
192.168.100.0 0.0.0.255
```

The guest VLAN is not going to need access to any other VLAN. As with the example in the Sales ACL.

3. PERMIT ACCESS TO UNDEFINED HOSTS

```
console(config)# access-list Guest permit ip any any
```

4. APPLY ACL TO INTERFACE

```
console(config)# interface vlan 5
console(config-if-vlan5)# ip access-group Guest
console(config-if-vlan5)# exit
console(config)# exit
console# copy running-config startup-config
```

Upon applying this, the switch will now restrict traffic between VLANs, but still allow the previously defined exceptions to have access to the resources they need. All internal client systems are getting the resources they need and prevented from going where they shouldn't.

EXAMPLE 4 – SECURING THE SERVER VLAN

While all traffic originating from VLANs 5, 10 and 20 display restrictions to the server VLAN, there is nothing preventing internet traffic from reaching the servers. For this example, we will state that all servers on this VLAN are for internal use only and should not pass outbound traffic.

This can be managed from one direction, but to ensure security, it is recommended to apply an ACL to both inbound and outbound traffic. We also have to be careful not to interfere with rules that are already in place. While it would be easy to deny all traffic from the server VLAN, that would block return traffic allowed in the ACLs made to other VLANs.

1. BLOCK INBOUND INTERNET TRAFFIC

```
console(config)# access-list External deny any 192.168.100.0 0.0.0.255
console(config)# access-list External deny any 192.168.20.0 0.0.0.255
```

Now that the inbound traffic is blocked, we can make sure that it gets back to the users who actually need it. Since there have been no other restrictions on outbound internet traffic, we won't need to make any further rules for inbound traffic.

2. ALLOW REMAINING INBOUND TRAFFIC AND APPLY RULE

```
console(config)# access-list External permit ip any any
console(config)# interface vlan 1
console(config-if-vlan1)# ip access-group External
```

With this rule in place, no traffic from the internet will reach the server or Production VLANs, but it's still possible that one way traffic from the servers will reach the internet. All internal traffic heading towards the servers has been accounted for, but now the server VLAN needs to be secured.

3. CREATE RULES TO ALLOW REQUIRED INTERNAL TRAFFIC AND APPLY RULE

```
console(config)# access-list Server permit tcp 192.168.100.4 0.0.0.0
eq 67 192.168.5.0 0.0.0.255
console(config)# access-list Server permit ip 192.168.100.5 0.0.0.0
192.168.5.0 0.0.0.255
console(config)# access-list Server permit ip 192.168.100.4 0.0.0.0
192.168.10.0 0.0.0.255
console(config)# access-list Server permit ip 192.168.100.5 0.0.0.0
192.168.10.0 0.0.0.255
console(config)# access-list Server permit ip 192.168.100.250 0.0.0.0
192.168.10.0 0.0.0.255
console(config)# access-list Server permit ip 192.168.100.0 0.0.0.255
192.168.20.0 0.0.0.255
```

The above ACEs are the inverse of all permitted internal traffic. The sources match the destinations defined above. In every other respect, the server VLAN can be blocked. Everything else will be implicity denied, so no more rules need to be added. The only thing that needs be done is to apply this to the interface and save the configuration with the following commands:

```
console(config)# interface vlan 100
console(config-if-vlan100)# ip access-group Server
console(config-if-vlan100)# exit
console(config)# exit
```

SUMMARY

With the applied rules in place, the network is now secured from unauthorized traffic crossing VLANs. When troubleshooting a user who does not have access to the required network, it should be first verified what IP they are connecting from and what network segment they communicate with. Additional rules may need to be applied if new resources need to be accessed by different segments.

CONCLUSION

Restricting IP traffic is an important part of managing network performance and adding a layer of security. A network administrator should understand that there is no such thing as an absolutely secure network. Access Control Lists are a component of a network security solution–not the entire solution.

As with any security related concern, every change should be considered carefully. Granting or removing access to one resource may create another unforeseen result. It is recommended that for any configuration change, a controlled test environment should be used to simulate effects on a production network.