

ACLs & Views

Elements in an address match list

- Individual IP addresses
- Addresses/netmask pairs
- Names of other ACLs
- In some contexts, key names.

Purposes in BIND

- Restricting queries & zone xfer
- Authorizing dynamic updates
- Selecting interfaces to listen on
- Sorting responses

*Address match lists are always enclosed in curly braces.

Notes on Address Match list

- Elements must be separated by “ ; ”
- The list must be terminated with a “ ; ”
- Elements of the address match list are checked sequentially.
- To negate elements of the address match list prepend them with “!”
- Use *acl* statement to name an address match list.
- *acl* must be defined before it can be used elsewhere.

Example: Address match lists

- For network 192.168.0.0 255.255.255.0
{ 192.168.0.0/24; }
- For network plus loopback
{ 192.168.0.0/24; 127.0.0.1; }
- Addresses plus key name
{ 192.168.0.0/24; 127.0.0.1; tequila.apnic.net; }

The *acl* Statement

- Syntax:
acl <acl name> { address match list};
- Example:
acl internal { 127.0.0.1; 192.168.0/24; };
acl dynamic-update { key
dhcp.apnic.net; };

Notes on the *acl* Statement

- The *acl* name need not be quoted.
- There are four predefined ACLs:
 - any* (Any IP address)
 - none* (No IP address)
 - localhost* (loopback, 127.0.0.1)
 - localnets* (all networks the name server is directly connected to)

Blackhole

```
options {
blackhole { ACL-name or
itemized list; };
};
```

Allow-transfer

```
zone "myzone.example." {
type master;
file "myzone.example.";
allow-transfer { ACL-name
or
itemized list; };
};
```

Allow-Query

```
zone "myzone.example." {  
    type master;  
    file "myzone.example."  
    allow-query { ACL-name or  
    itemized list; };  
};
```

Listen-on

```
options {  
    listen-on port #  
        { ACL-name or  
        itemized list;};  
};
```

Summary

- ACLs and Configuration options can be used to create simple split DNS
- It is cumbersome and difficult to maintain.
- Good operational practice suggests that ACLs and configuration options be reviewed regularly to ensure that they accurately reflect desired behaviour.

Views

The view statement is a powerful new feature of BIND 9 that lets a name server answer a DNS query differently depending on who is asking. It is particularly useful for implementing split DNS setups without having to run multiple servers.

Syntax

- ```
view view_name [class] {
 match-clients {address_match_list } ;
 match-destinations {address_match_list } ;
 match-recursive-only yes_or_no ;
 [view_option; ...]
 [zone_statement; ...]
};
```

---

---

---

---

---

---

---

---

## Example Config

- ```
view "internal" {
    // This should match our internal networks.
    match-clients { 10.0.0.0/8; };

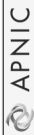
    // Provide recursive service to internal clients only.
    recursion yes;

    // Provide a complete view of the example.com zone
    // including addresses of internal hosts.
    zone "example.com" {
        type master;
        file "example-internal.db";
    };
};
```



Continued

```
view "external" {  
    // Match all clients not matched by the previous view.  
    match-clients { any; };  
  
    // Refuse recursive service to external clients.  
    recursion no;  
  
    // Provide a restricted view of the example.com zone  
    // containing only publicly accessible hosts.  
    zone "example.com" {  
        type master;  
        file "example-external.db";  
    };  
};
```



Questions?
