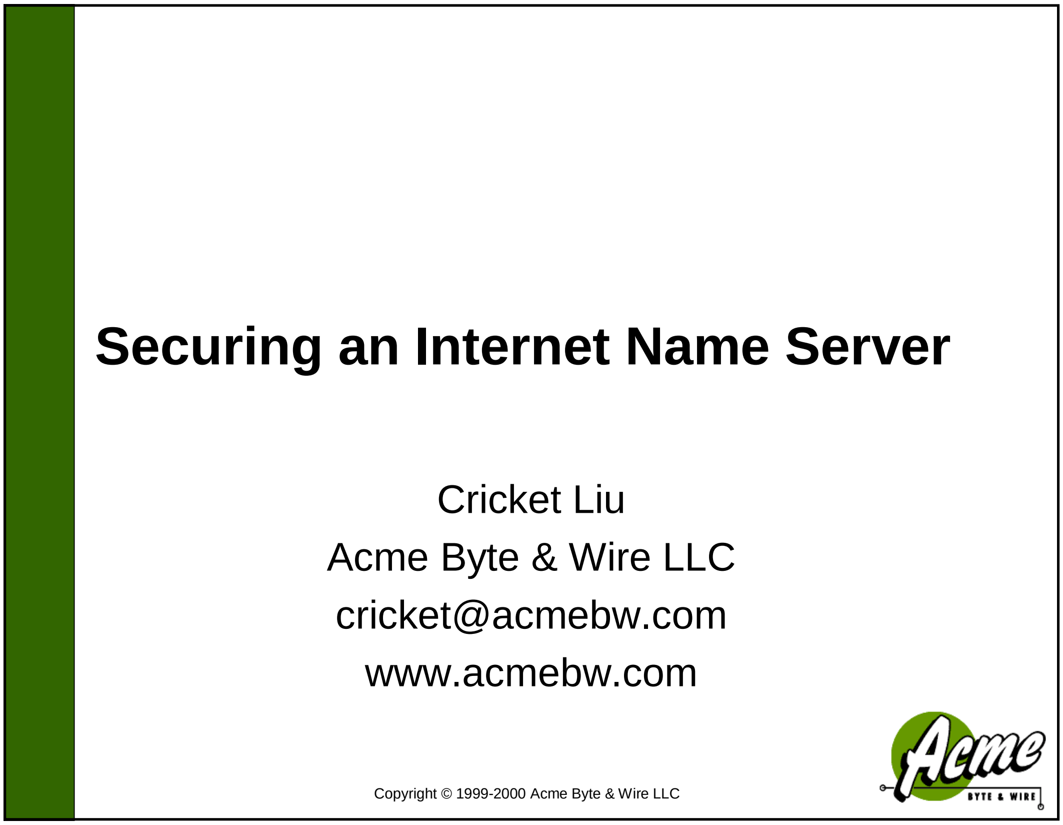
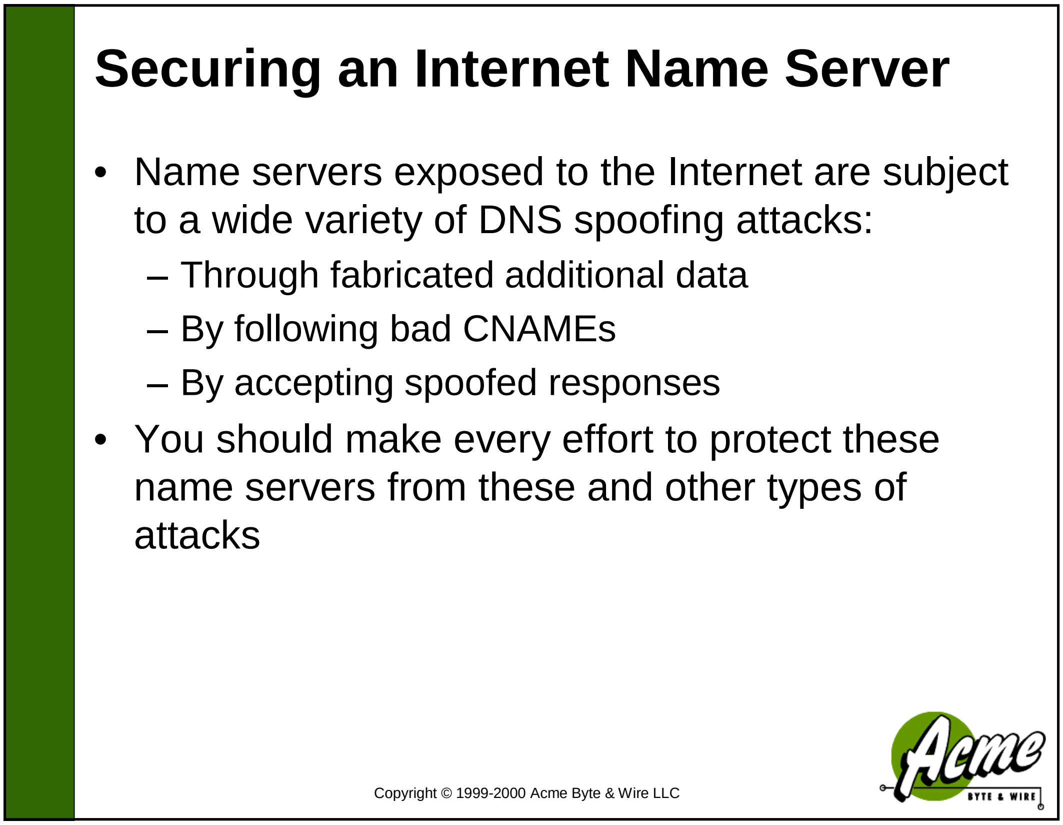


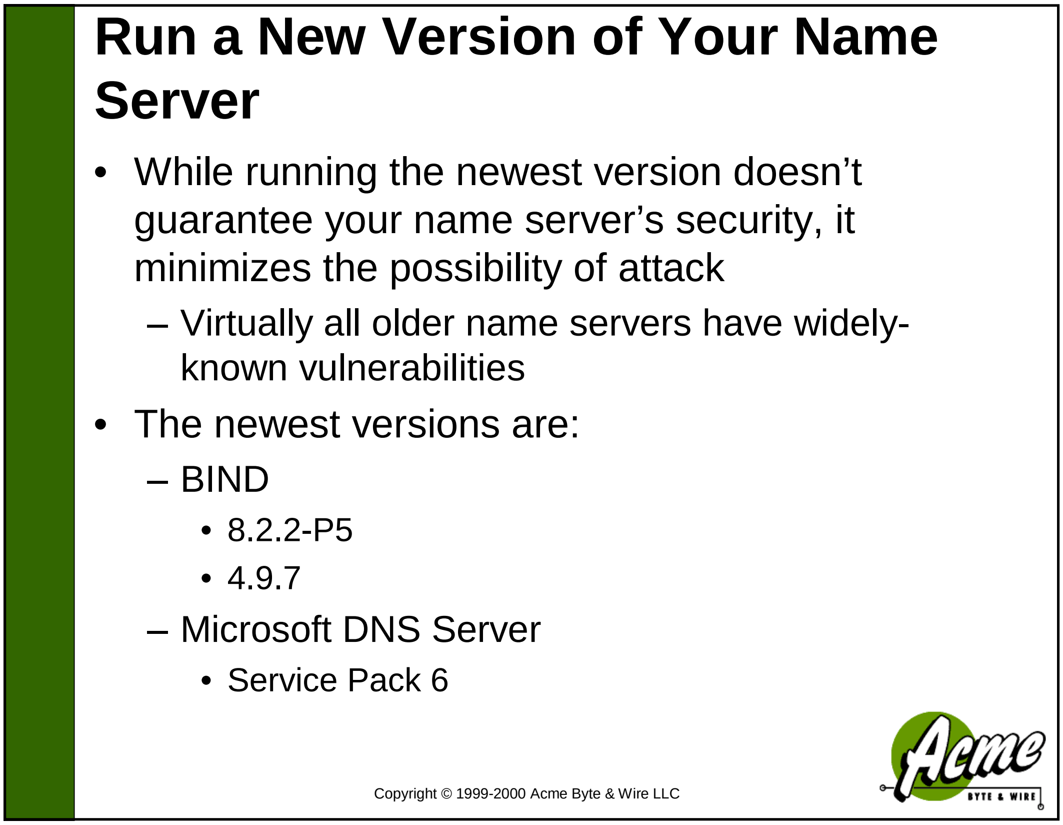
Securing an Internet Name Server

Cricket Liu
Acme Byte & Wire LLC
cricket@acmebw.com
www.acmebw.com



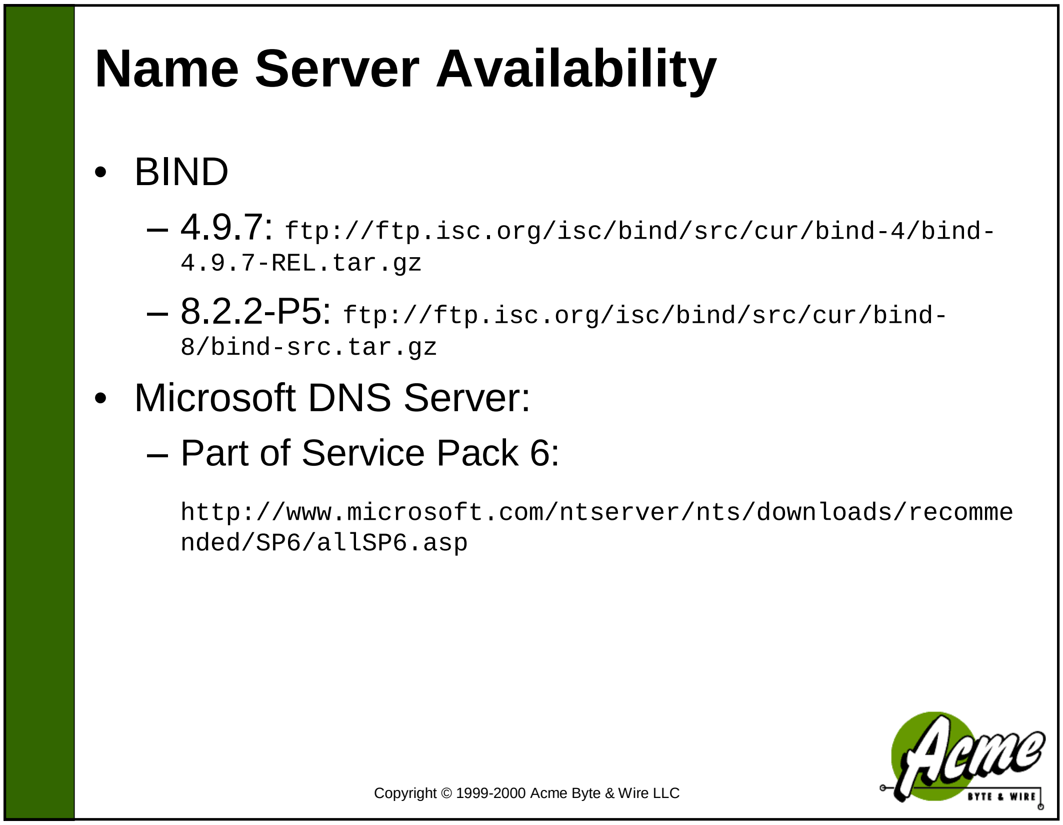
Securing an Internet Name Server

- Name servers exposed to the Internet are subject to a wide variety of DNS spoofing attacks:
  - Through fabricated additional data
  - By following bad CNAMEs
  - By accepting spoofed responses
- You should make every effort to protect these name servers from these and other types of attacks



Run a New Version of Your Name Server

- While running the newest version doesn’t guarantee your name server’s security, it minimizes the possibility of attack
  - Virtually all older name servers have widely-known vulnerabilities
- The newest versions are:
  - BIND
    - 8.2.2-P5
    - 4.9.7
  - Microsoft DNS Server
    - Service Pack 6



Name Server Availability

- BIND
  - 4.9.7: `ftp://ftp.isc.org/isc/bind/src/cur/bind-4/bind-4.9.7-REL.tar.gz`
  - 8.2.2-P5: `ftp://ftp.isc.org/isc/bind/src/cur/bind-8/bind-src.tar.gz`
- Microsoft DNS Server:
  - Part of Service Pack 6:

    `http://www.microsoft.com/ntserver/nts/downloads/recommended/SP6/allSP6.asp`

Restrict Zone Transfers

- Restricting zone transfers prevents
  - Others from taxing your name server
  - Hackers from listing the contents of your zones
    - To identify targets
      - Mail servers
      - Name servers
    - To gain “host demographic” information
      - How many hosts you have
      - What makes and models you have
      - What their names are (valuable if you name them after people or projects)

Restricting Zone Transfers with BIND 4

- With BIND 4.9, use the *xfrnets* directive:

```
xfrnets 206.168.119.178&255.255.255.255
```

- This controls which name servers can transfer *any* zones from this name server

Restricting Zone Transfers with BIND 8

- With BIND 8, use the *allow-transfer* substatement:

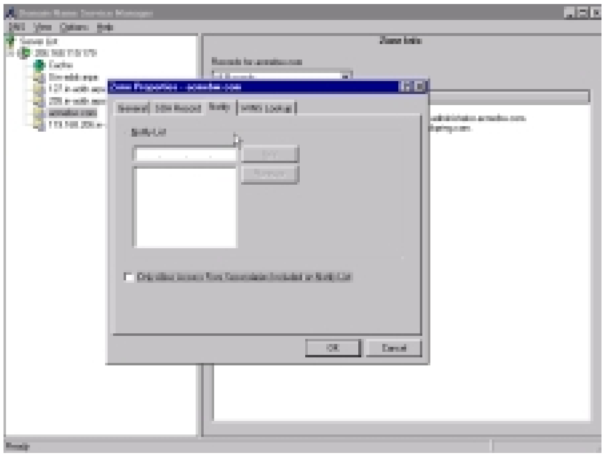
```
options {
     allow-transfer { 206.168.119.178; };
};
```

or, specific to a zone:

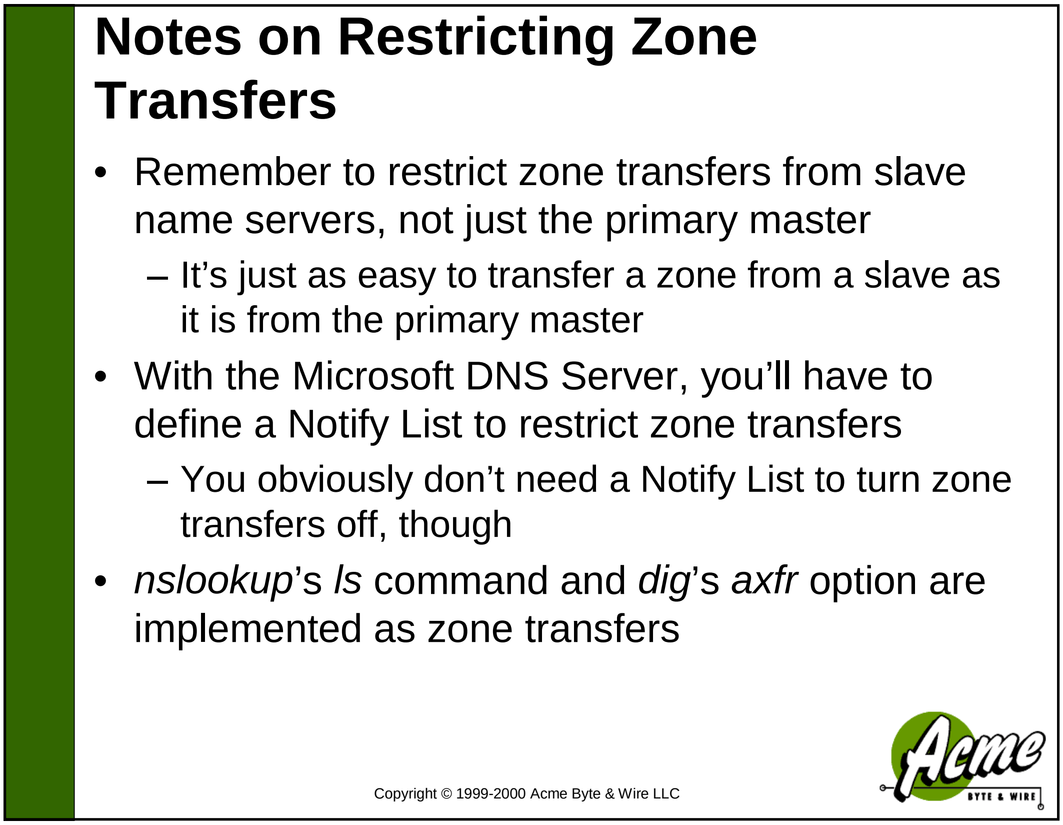
```
zone “acmebw.com” {
     type master;
     file “db.acmebw.com”;
     allow-transfer { 206.168.119.178; };
};
```

Restricting Zone Transfers with the Microsoft DNS Server

- Use the *Notify* tab of the *Zone Properties* window:

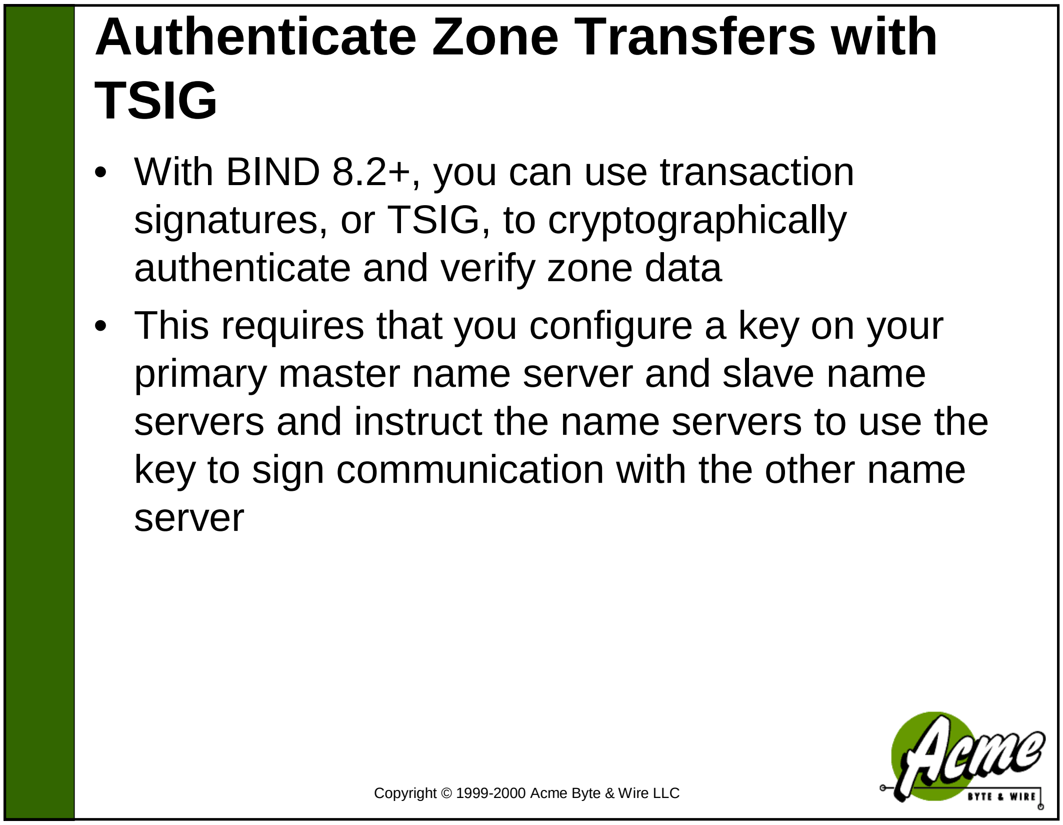

Domain Name Service Manager
DNS View Options Help
Server List
Cache
Zone Info
Zone Properties - acmebw.com
General
SOA Record
Notify
WINS Lookup
Notify List
Add
Remove
□ Only Allow Access From Secondaries Included on Notify List
OK
Cancel
Ready

- Check *Only Allow Access From Secondaries Included on Notify List*



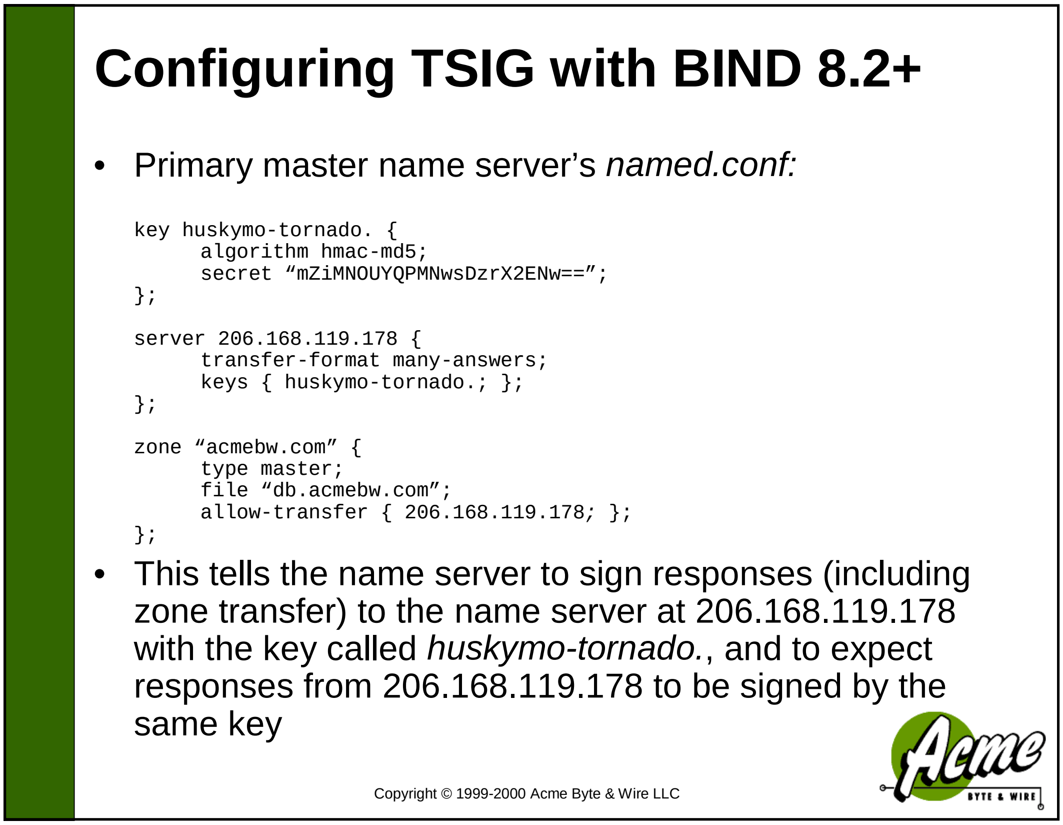
Notes on Restricting Zone Transfers

- Remember to restrict zone transfers from slave name servers, not just the primary master
  - It’s just as easy to transfer a zone from a slave as it is from the primary master
- With the Microsoft DNS Server, you’ll have to define a Notify List to restrict zone transfers
  - You obviously don’t need a Notify List to turn zone transfers off, though
- *nslookup*’s *ls* command and *dig*’s *axfr* option are implemented as zone transfers



Authenticate Zone Transfers with TSIG

- With BIND 8.2+, you can use transaction signatures, or TSIG, to cryptographically authenticate and verify zone data
- This requires that you configure a key on your primary master name server and slave name servers and instruct the name servers to use the key to sign communication with the other name server

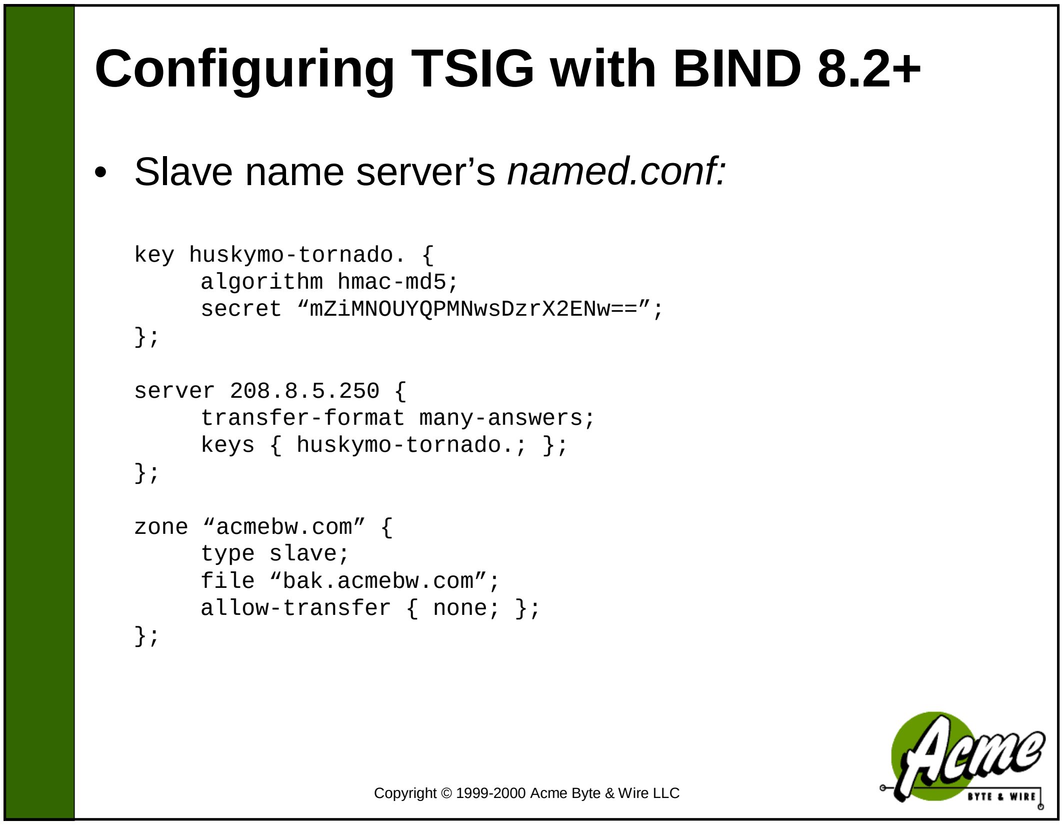


Configuring TSIG with BIND 8.2+

- Primary master name server’s *named.conf:*

```
key huskymo-tornado. {
     algorithm hmac-md5;
     secret “mZiMNOUYQPMNwsDzrX2ENw==”;
};

server 206.168.119.178 {
     transfer-format many-answers;
     keys { huskymo-tornado.; };
};

zone “acmebw.com” {
     type master;
     file “db.acmebw.com”;
     allow-transfer { 206.168.119.178; };
};
```

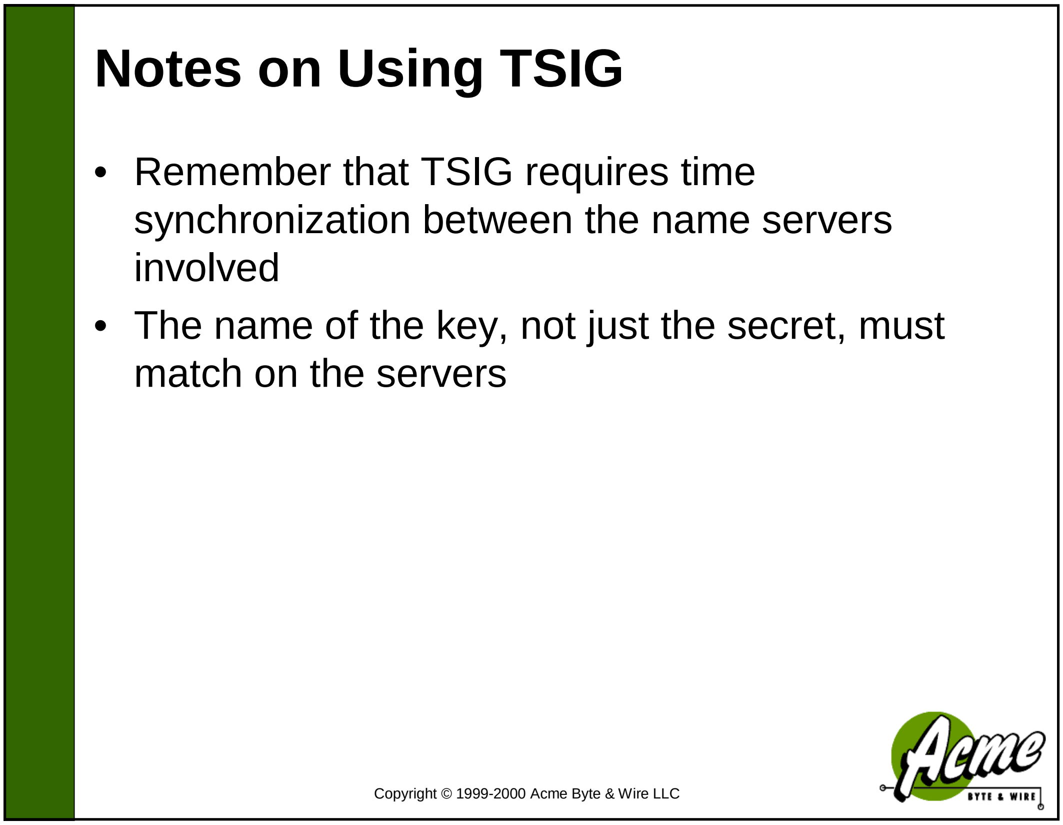
- This tells the name server to sign responses (including zone transfer) to the name server at 206.168.119.178 with the key called *huskymo-tornado.*, and to expect responses from 206.168.119.178 to be signed by the same key



Configuring TSIG with BIND 8.2+

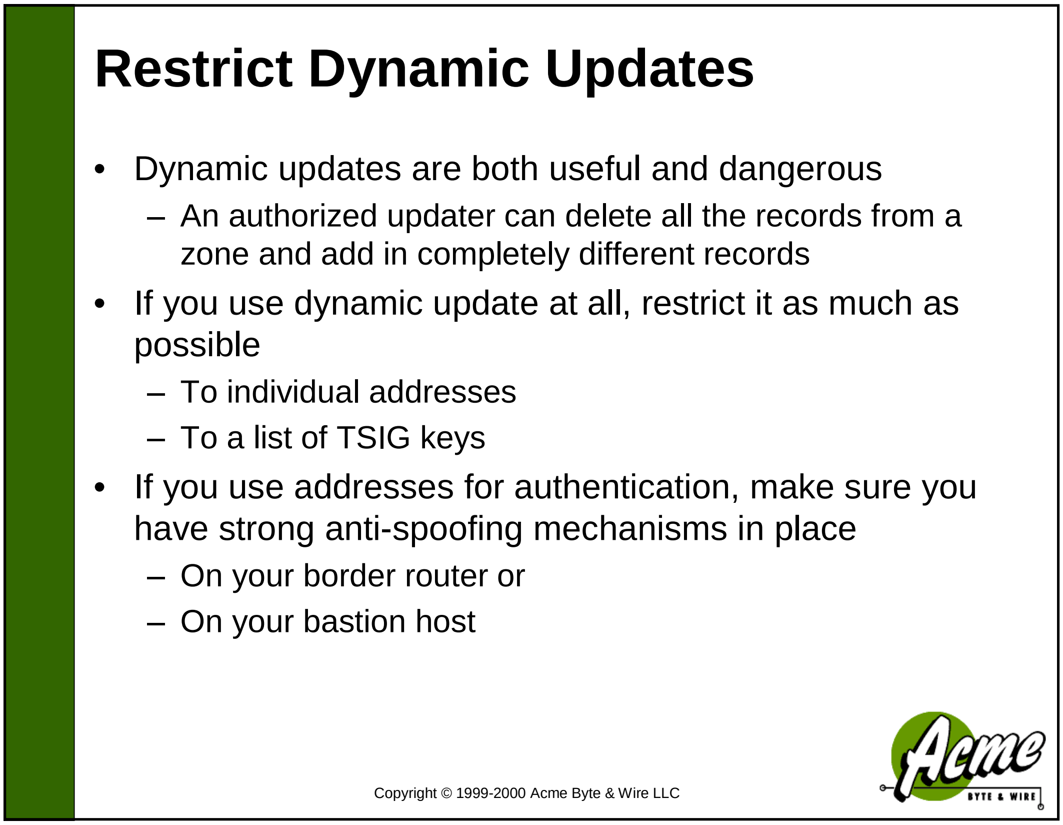
- Slave name server’s *named.conf:*

```
key huskymo-tornado. {
     algorithm hmac-md5;
     secret “mZiMNOUYQPMNwsDzrX2ENw==”;
};

server 208.8.5.250 {
     transfer-format many-answers;
     keys { huskymo-tornado.; };
};

zone “acmebw.com” {
     type slave;
     file “bak.acmebw.com”;
     allow-transfer { none; };
};
```



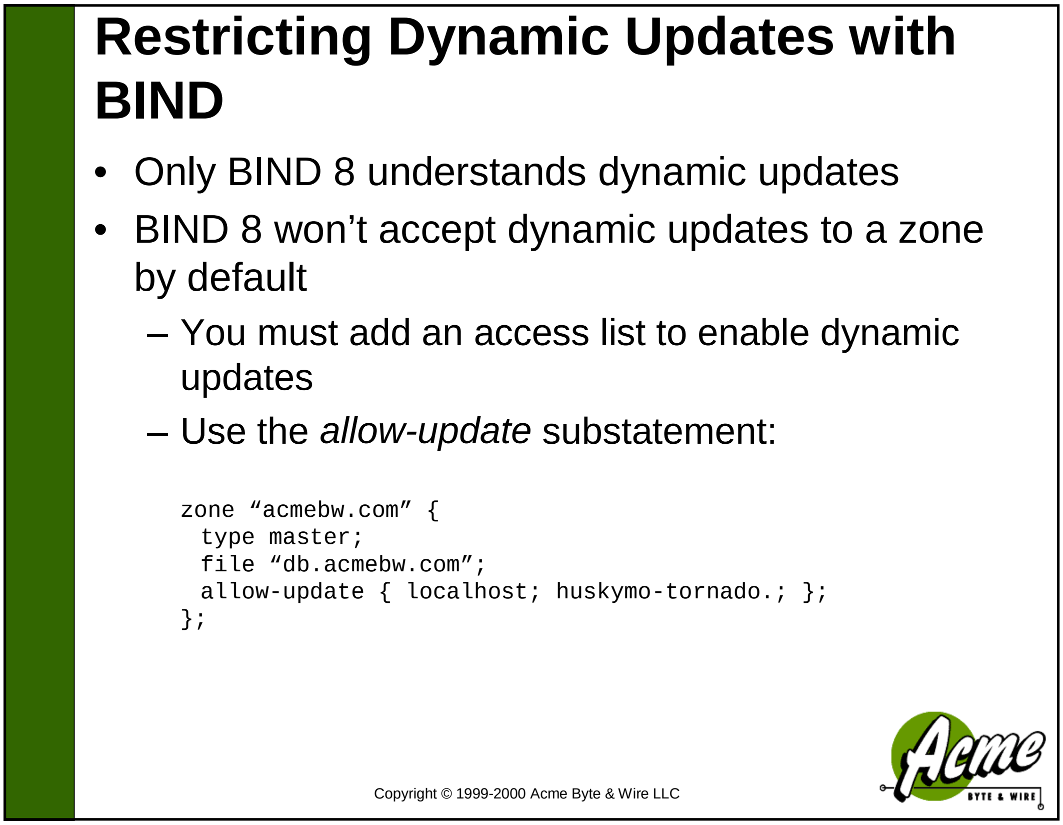
Notes on Using TSIG

- Remember that TSIG requires time synchronization between the name servers involved
- The name of the key, not just the secret, must match on the servers



Restrict Dynamic Updates

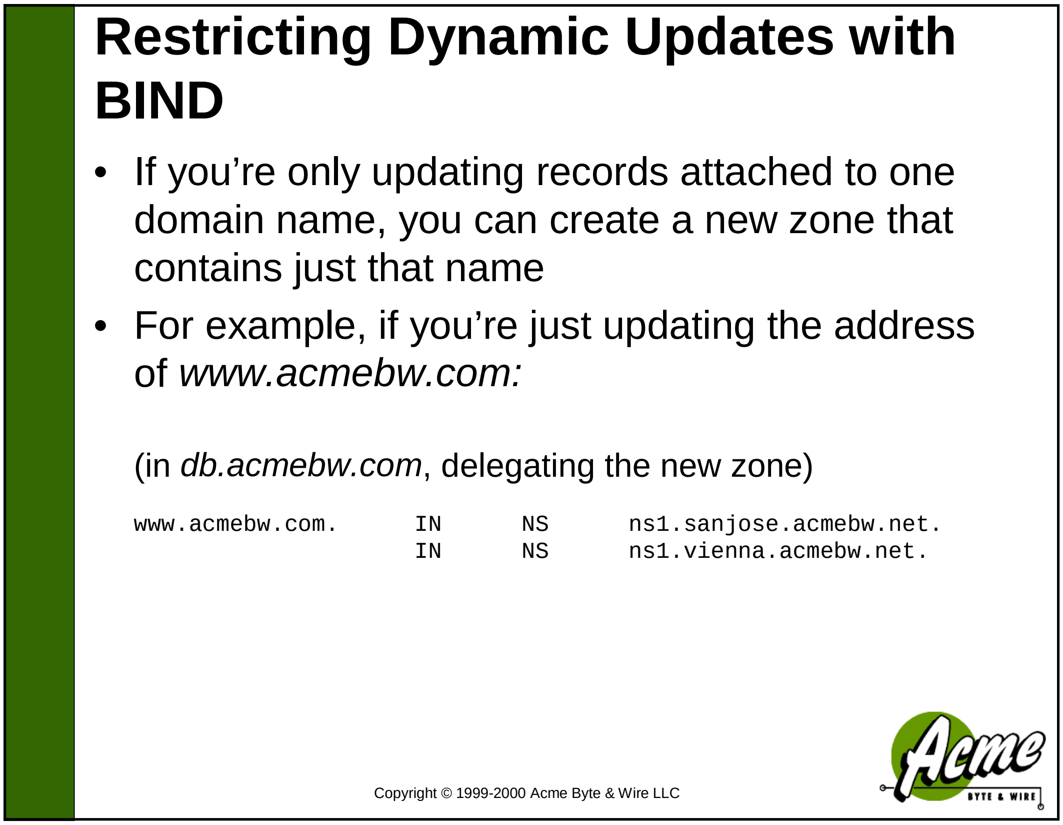
- Dynamic updates are both useful and dangerous
  - An authorized updater can delete all the records from a zone and add in completely different records
- If you use dynamic update at all, restrict it as much as possible
  - To individual addresses
  - To a list of TSIG keys
- If you use addresses for authentication, make sure you have strong anti-spoofing mechanisms in place
  - On your border router or
  - On your bastion host



Restricting Dynamic Updates with BIND

- Only BIND 8 understands dynamic updates
- BIND 8 won’t accept dynamic updates to a zone by default
  - You must add an access list to enable dynamic updates
  - Use the *allow-update* substatement:

```
zone “acmebw.com” {
  type master;
  file “db.acmebw.com”;
  allow-update { localhost; huskymo-tornado.; };
};
```

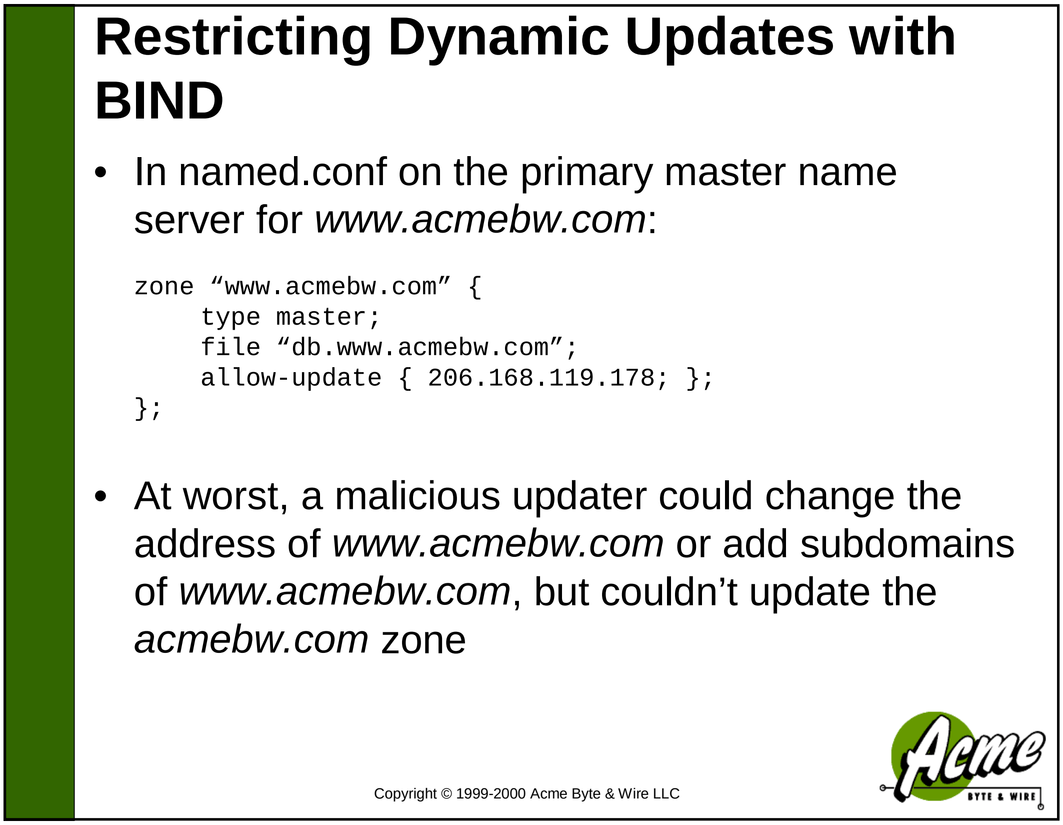


Restricting Dynamic Updates with BIND

- If you’re only updating records attached to one domain name, you can create a new zone that contains just that name
- For example, if you’re just updating the address of *www.acmebw.com:*

(in *db.acmebw.com*, delegating the new zone)

```
www.acmebw.com.      IN      NS      ns1.sanjose.acmebw.net.
                     IN      NS      ns1.vienna.acmebw.net.
```

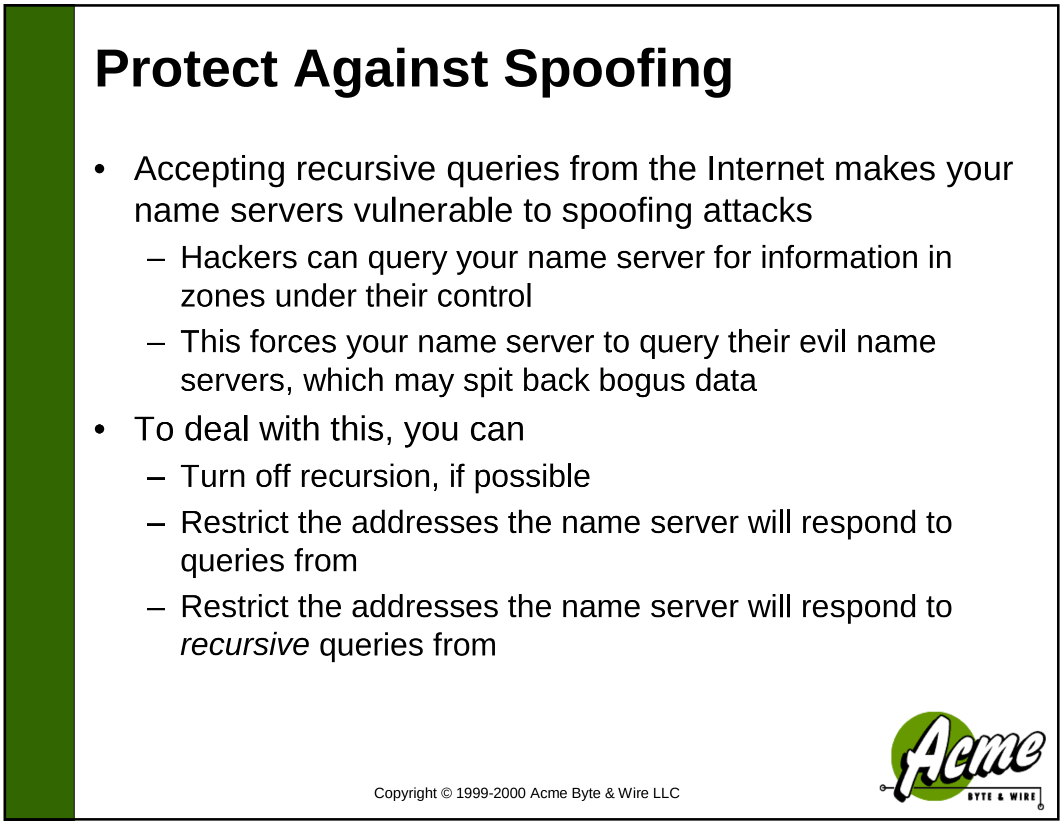


Restricting Dynamic Updates with BIND

- In named.conf on the primary master name server for *www.acmebw.com*:

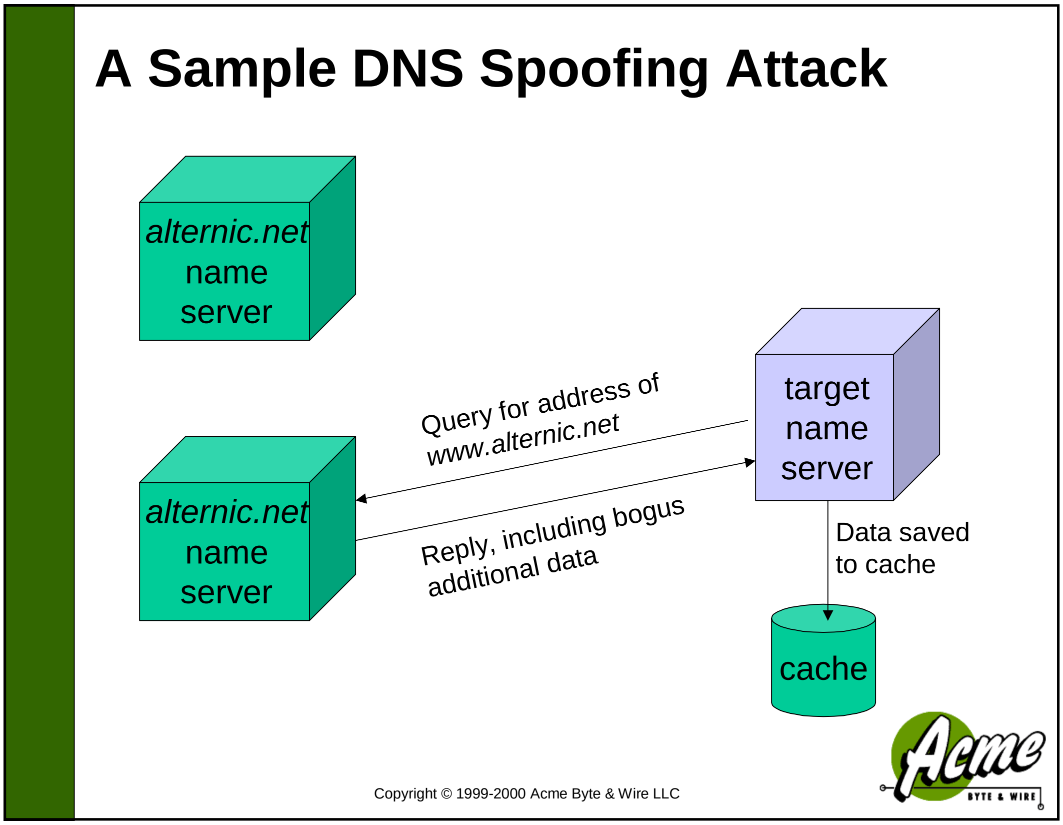
```
zone “www.acmebw.com” {
     type master;
     file “db.www.acmebw.com”;
     allow-update { 206.168.119.178; };
};
```

- At worst, a malicious updater could change the address of *www.acmebw.com* or add subdomains of *www.acmebw.com*, but couldn’t update the *acmebw.com* zone



Protect Against Spoofing

- Accepting recursive queries from the Internet makes your name servers vulnerable to spoofing attacks
  - Hackers can query your name server for information in zones under their control
  - This forces your name server to query their evil name servers, which may spit back bogus data
- To deal with this, you can
  - Turn off recursion, if possible
  - Restrict the addresses the name server will respond to queries from
  - Restrict the addresses the name server will respond to *recursive* queries from



A Sample DNS Spoofing Attack

alternic.net name server

alternic.net name server

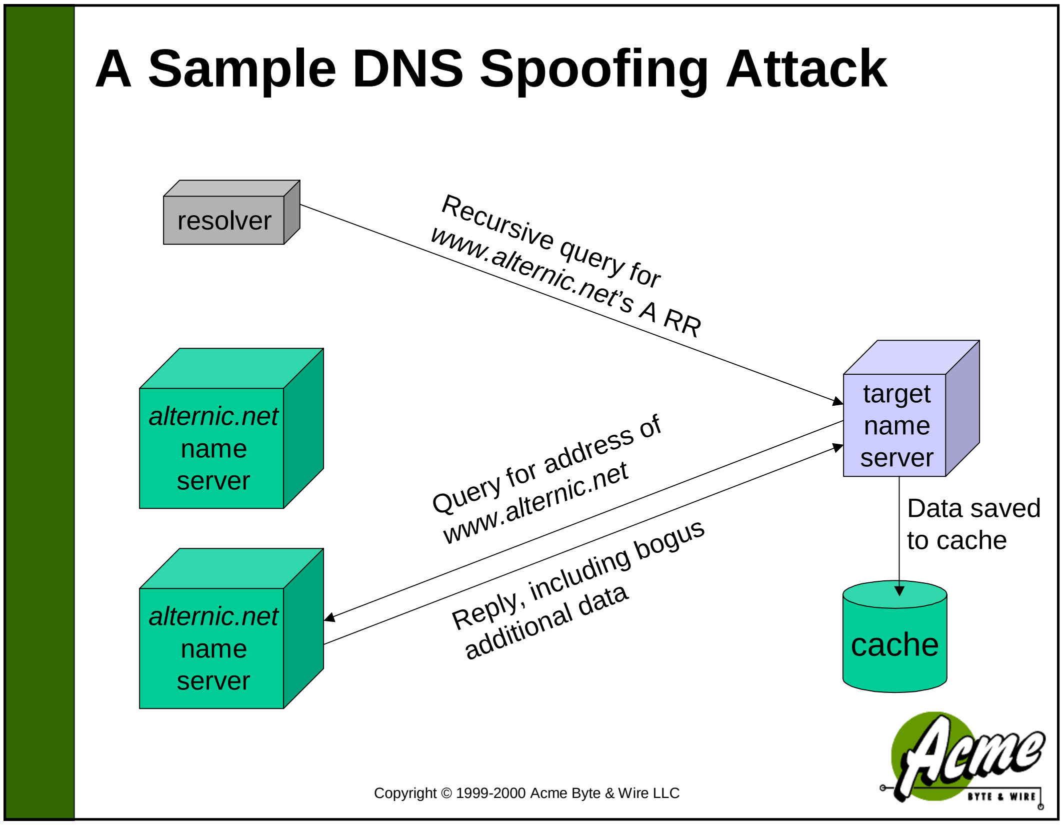
target name server

Query for address of *www.alternic.net*

Reply, including bogus additional data

Data saved to cache

cache



A Sample DNS Spoofing Attack

resolver

Recursive query for *www.alternic.net's A RR*

alternic.net name server

alternic.net name server

target name server

Query for address of *www.alternic.net*

Reply, including bogus additional data

Data saved to cache

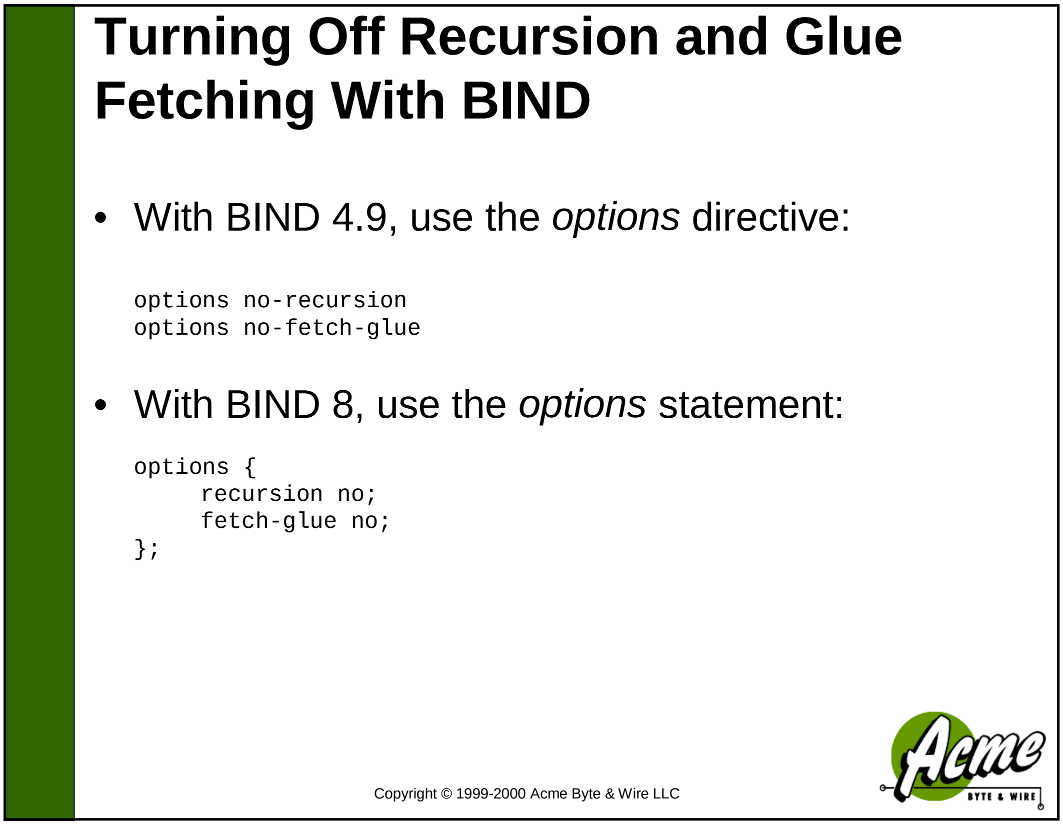
cache

Turning Off Recursion

- Disabling recursion puts your name servers into a passive mode, telling them never to send queries on behalf of other name servers or resolvers
  - A non-recursive name server is very difficult to spoof, since it doesn't send queries, and hence doesn't cache any data
  - You can't disable recursion on a name server if any legitimate resolvers query it, or if other name servers use it as a forwarder
  - If you can't disable recursion, restrict the queries the name server will accept, shown later

Turning Off Glue Fetching

- Normally a name server returning NS records for which it does not have A records will attempt to retrieve them
  - This is called *glue fetching*
  - A potential source of a spoofed response
- Turning off glue fetching prevents this lookup
  - The name server will never generate any queries
  - And will not build up a cache



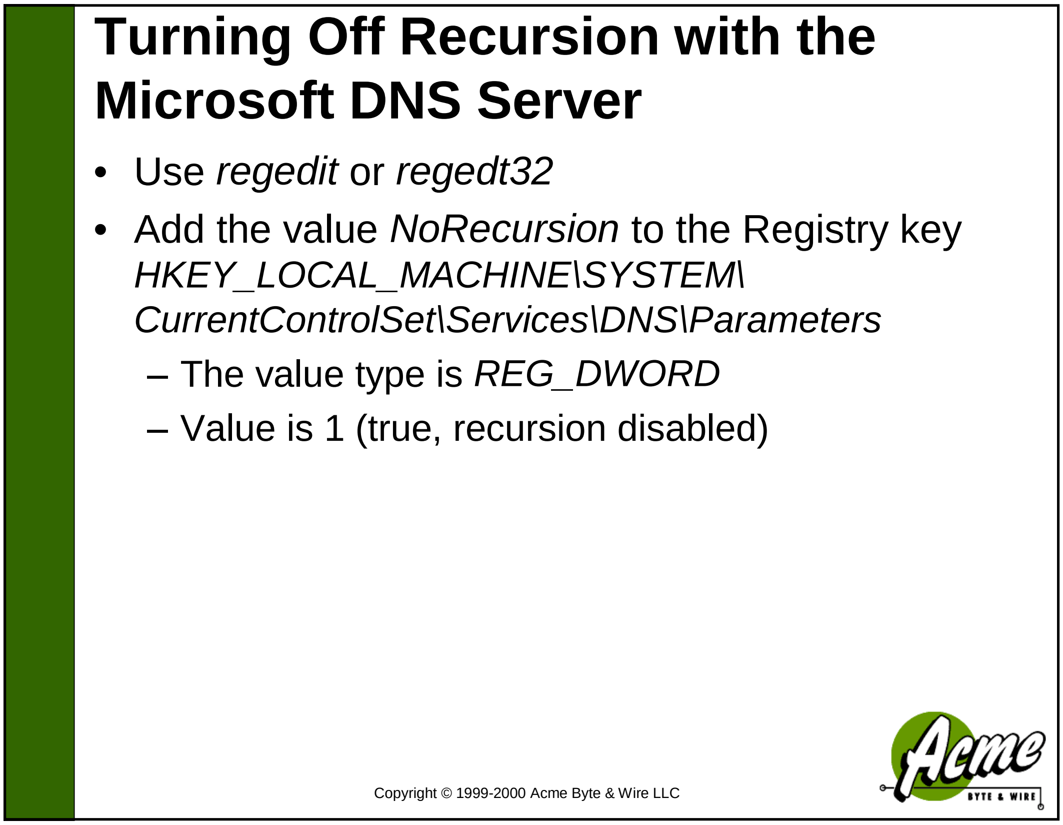
Turning Off Recursion and Glue Fetching With BIND

- With BIND 4.9, use the *options* directive:

```
options no-recursion
options no-fetch-glue
```

- With BIND 8, use the *options* statement:

```
options {
     recursion no;
     fetch-glue no;
};
```



Turning Off Recursion with the Microsoft DNS Server

- Use *regedit* or *regedt32*
- Add the value *NoRecursion* to the Registry key *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\DNS\Parameters*
  - The value type is *REG_DWORD*
  - Value is 1 (true, recursion disabled)

Restrict Queries

- If you can’t turn off recursion, restrict the queries that your name servers accept to:
  - The addresses they should come from
  - The zones they should ask about
- On most name servers
  - Queries for records in authoritative zones can come from anywhere, because the zones are delegated to the name server
  - Queries for records outside of authoritative zones should only come from internal addresses

Restricting Queries with BIND

- Unfortunately, only BIND 8 (i.e., not BIND 4 or the Microsoft DNS Server) will let you establish such fine-grained access
- Use the *allow-query* substatement:

```
acl internal { 206.168.119.176/29; };

options {
    directory “/var/named”;
    allow-query { internal; };
};

zone “acmebw.com” {
    type master;
    file “db.acmebw.com”;
    allow-query { any; };

};
```

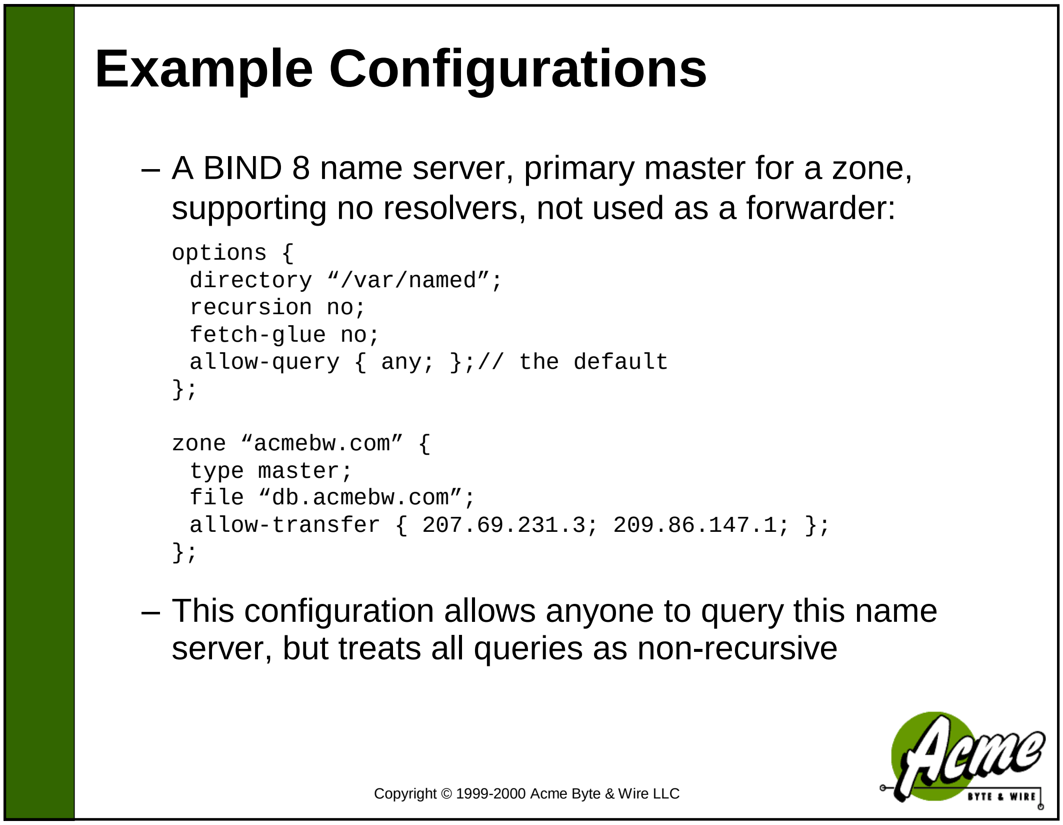
Restrict Recursive Queries

- BIND 8.2.1+ allows you to restrict the IP addresses you accept *recursive* queries from
  - Queriers from other IP addresses will have their recursive queries processed as non-recursive
- Use the *allow-recursion* substatement:

```
acl internal { 206.168.119.178/29; };

options {
     directory “/var/named”;
     allow-recursion { 206.168.119.176/29; };
};

zone “acmebw.com” {
     type master;
     file “db.acmebw.com”;
};
```

Example Configurations

- Here are some example configurations showing you how to put it all together

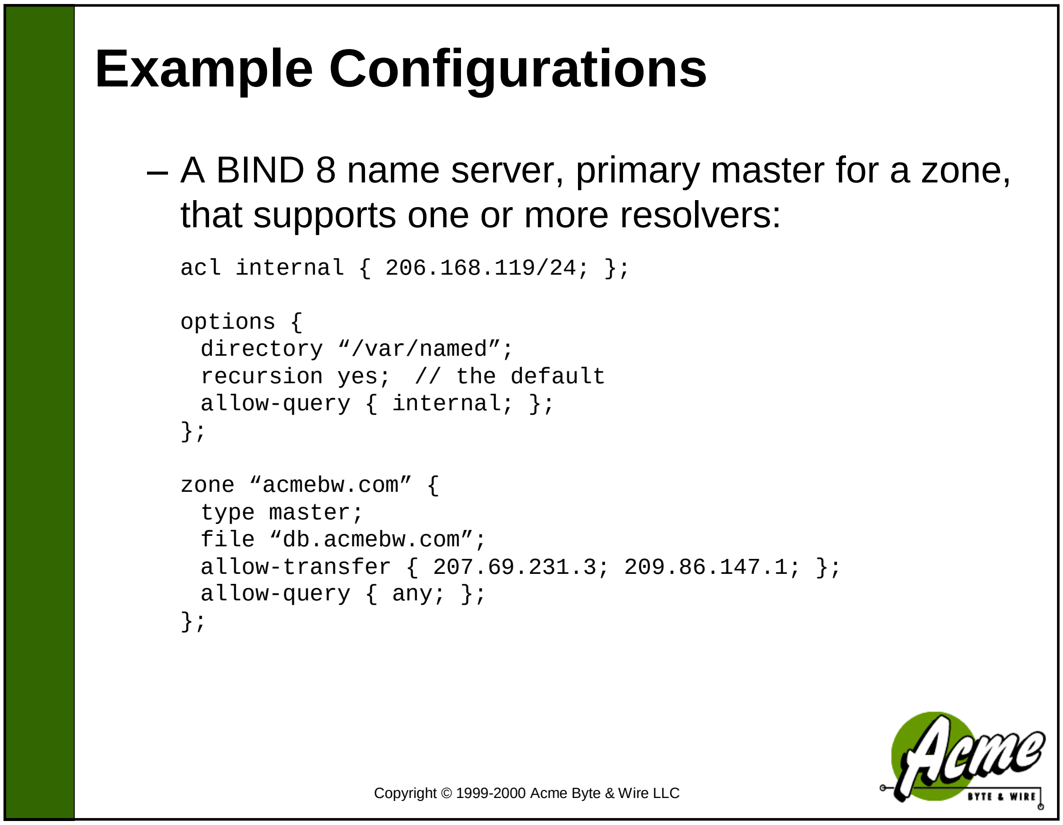


Example Configurations

– A BIND 8 name server, primary master for a zone, supporting no resolvers, not used as a forwarder:

```
options {
 directory “/var/named”;
 recursion no;
 fetch-glue no;
 allow-query { any; };// the default
};

zone “acmebw.com” {
 type master;
 file “db.acmebw.com”;
 allow-transfer { 207.69.231.3; 209.86.147.1; };
};
```

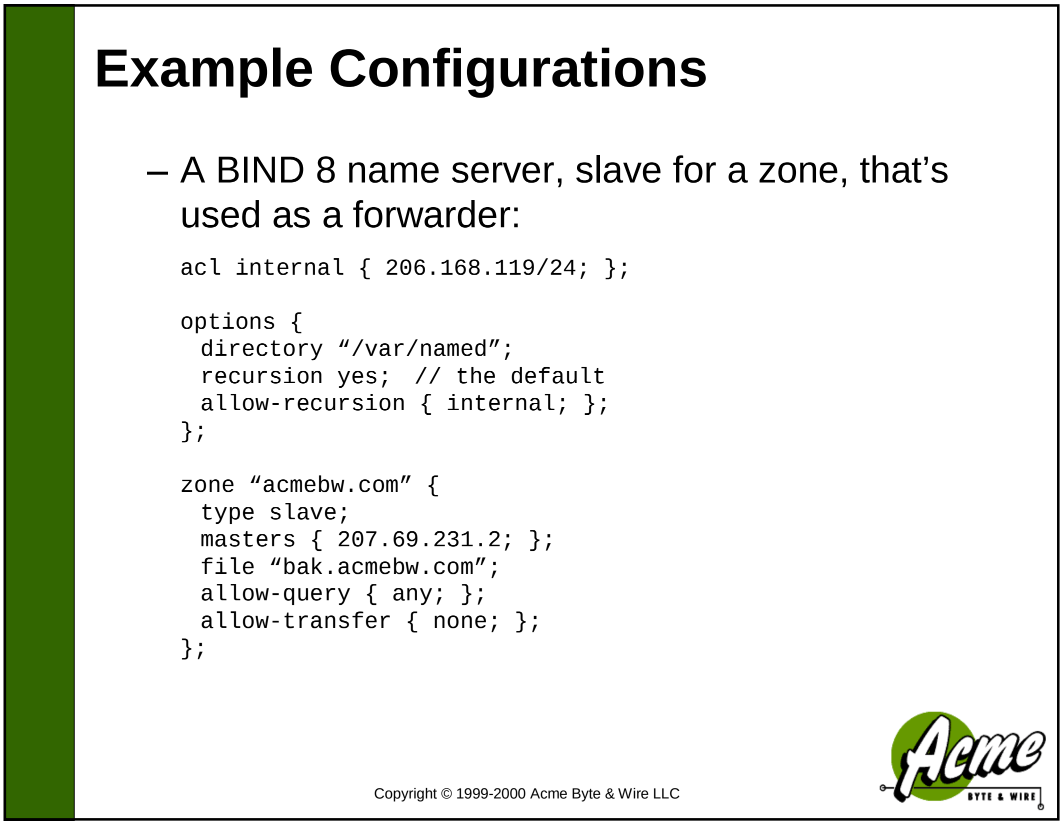
– This configuration allows anyone to query this name server, but treats all queries as non-recursive



Example Configurations

– A BIND 8 name server, primary master for a zone, that supports one or more resolvers:

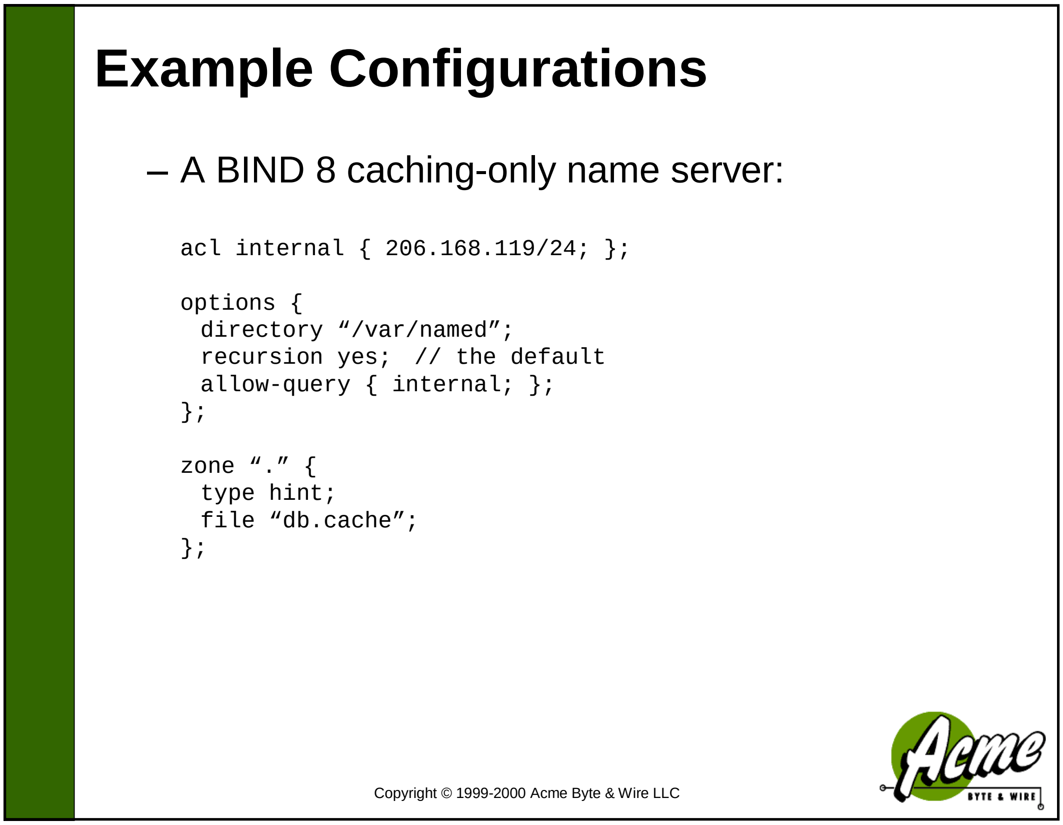
```
acl internal { 206.168.119/24; };

options {
 directory “/var/named”;
 recursion yes;  // the default
 allow-query { internal; };
};

zone “acmebw.com” {
 type master;
 file “db.acmebw.com”;
 allow-transfer { 207.69.231.3; 209.86.147.1; };
 allow-query { any; };
};
```



Example Configurations

– A BIND 8 name server, slave for a zone, that’s used as a forwarder:

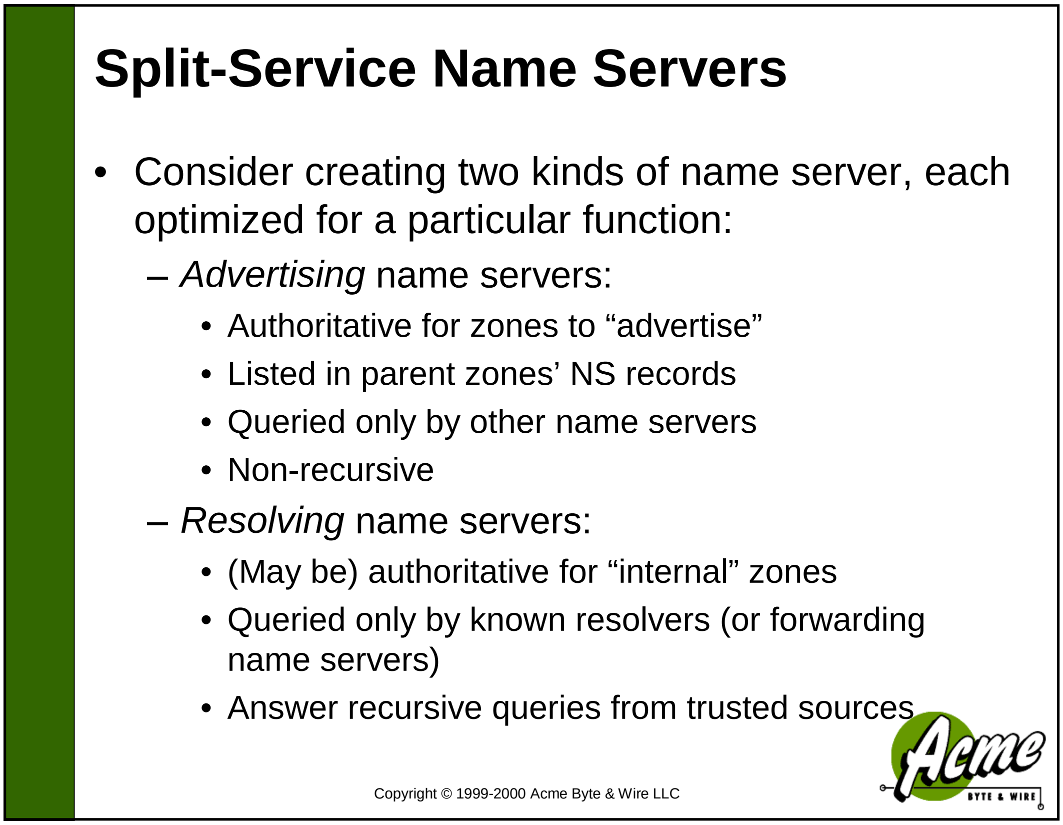
```
acl internal { 206.168.119/24; };

options {
 directory “/var/named”;
 recursion yes;  // the default
 allow-recursion { internal; };
};

zone “acmebw.com” {
 type slave;
 masters { 207.69.231.2; };
 file “bak.acmebw.com”;
 allow-query { any; };
 allow-transfer { none; };
};
```



Example Configurations

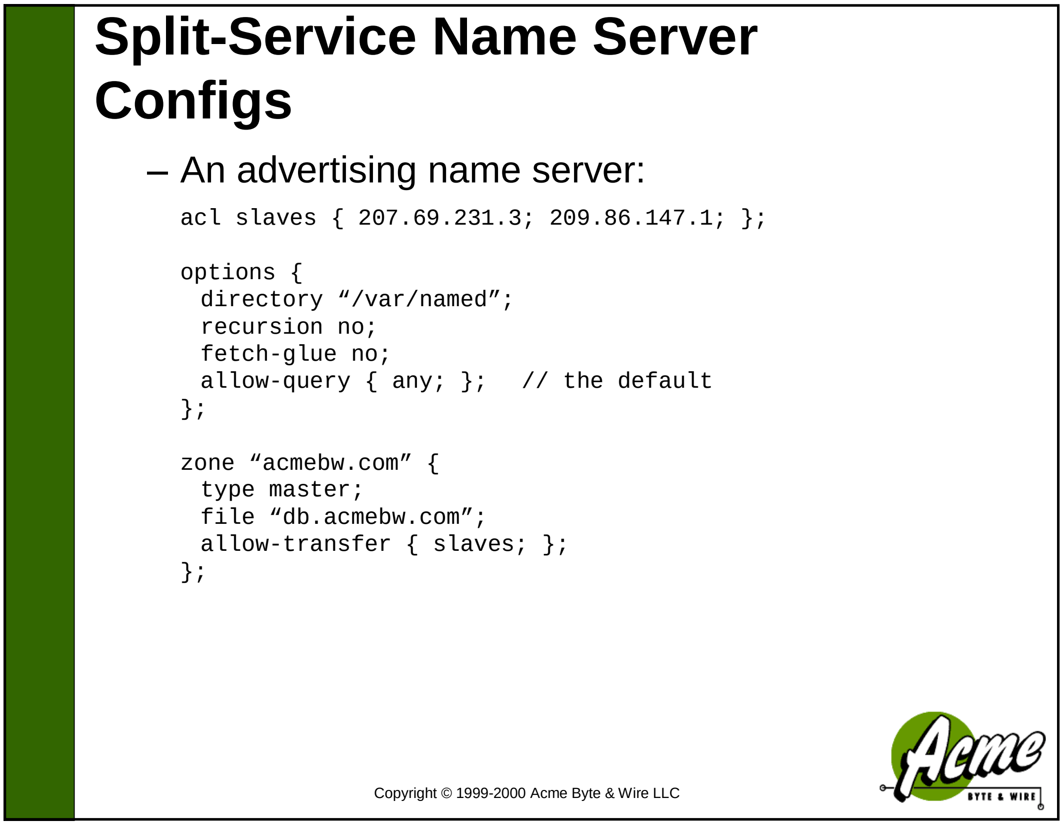
– A BIND 8 caching-only name server:

```
acl internal { 206.168.119/24; };

options {
 directory “/var/named”;
 recursion yes;  // the default
 allow-query { internal; };
};

zone “.” {
 type hint;
 file “db.cache”;
};
```



Split-Service Name Servers

- Consider creating two kinds of name server, each optimized for a particular function:
  - *Advertising* name servers:
    - Authoritative for zones to “advertise”
    - Listed in parent zones’ NS records
    - Queried only by other name servers
    - Non-recursive
  - *Resolving* name servers:
    - (May be) authoritative for “internal” zones
    - Queried only by known resolvers (or forwarding name servers)
    - Answer recursive queries from trusted sources



Split-Service Name Server Configs

- An advertising name server:

```
acl slaves { 207.69.231.3; 209.86.147.1; };

options {
  directory “/var/named”;
  recursion no;
  fetch-glue no;
  allow-query { any; };   // the default
};

zone “acmebw.com” {
  type master;
  file “db.acmebw.com”;
  allow-transfer { slaves; };
};
```

Split-Service Name Server Configs

– A resolving name server:

```
acl internal { 192.168.0/24; };

options {
 directory “/var/named”;
 recursion yes;     // the default
 allow-query { internal; };
};

zone “.” {
 type hint;
 file “db.cache”;
};

zone “acmebw.com” {
 type slave;
 masters { 207.69.231.2; };
 file “bak.acmebw.com”;
 allow-transfer { internal; };
};
```

Follow Relevant Newsgroups and Mailing Lists

- (Unfortunately) New vulnerabilities are found in name servers all the time
- (Fortunately) These vulnerabilities are usually patched quickly
- If you follow the relevant newsgroups and mailing lists closely, you’ll find out about the vulnerabilities and any necessary reconfiguration or patches quickly

Newsgroups and Mailing Lists Relevant to BIND

- *comp.protocols.dns.bind*
  - (and *bind-users@isc.org,* its mailing list equivalent; join by sending mail to *bind-users-request@isc.org*)
- *comp.protocols.tcp-ip.domains*
- The CERT mailing list (join by sending mail to *cert-advisory-request@cert.org*)
- Your UNIX vendor's security announcement mailing list

Newsgroups and Mailing Lists Relevant to Microsoft DNS Server

- *microsoft.public.windowsnt.dns*
- The CERT mailing list (join by sending mail to *cert-advisory-request@cert.org*)
- Microsoft's Product Security Notification Service (subscribe by sending mail to *microsoft_security-subscribe-request@announce.microsoft.com*)