BIND configuration

BIND

- BIND
  - the Berkeley Internet Name Domain system
- Main versions
  - BIND4
    - Announced in 1980s
    - Based on RFC 1034, 1035
    - Deprecated in 2001
  - BIND8
    - Released in 1997
    - Improvements including:
      - efficiency, robustness and security
    - Deprecated in 2007
  - BIND9
    - Released in 2000
    - Enhancements including:
      - multiprocessor support, DNSSEC, IPv6 support, etc

BIND – components

- Three major components
  - named
    - Daemon that answers the DNS query
    - Perform Zone transfer
  - Library routines
    - Routines that used to resolve host by contacting the servers of DNS distributed database
      - Ex: res_query, res_search, ...etc.
  - Command-line interfaces to DNS
    - Ex: nslookup, dig, hosts

named in FreeBSD

- startup
  - Edit /etc/rc.conf
    - named_enable="YES"
  - Manual utility command
    - % rndc {stop | reload | flush ...}
- Configuration files
  - /etc/namedb/named.conf (Configuration file)
  - /etc/namedb/named.root (DNS root server cache hint file)
  - Zone data files
- See your BIND version
  - % dig @127.0.0.1 version.bin txt chaos
    - version.bind. 0 CH TXT "9.3.3"

BIND Configuration – named.conf

- /etc/namedb/named.conf
  - Roles of this name server
    - Master, slave, or stub
  - Global options
  - Zone specific options

- named.conf is composed of following statements:
  - include, options, server, key, acl, zone, view, controls, logging, trusted-keys

BIND Configuration – named.conf address match list

- Address Match List
  - A generalization of an IP address that can include:
    - An IP address
      - Ex. 140.113.17.1
    - An IP network with CIDR netmask
      - Ex. 140.113/16
    - The ! character to do negate
    - The name of a previously defined ACL
    - A cryptographic authentication key
  - First match
  - Example:
    - {!1.2.3.4; 1.2.3/24;};
    - {128.138/16; 198.11.16/24; 204.228.69/24; 127.0.0.1;};

BIND Configuration – named.conf acl

- The "acl" statement
  - Define a class of access control
  - Define before they are used
  - Syntax

```
acl acl_name {
        address_match_list
};
```

  - Predefined acl classes
    - any, localnets, localhost, none
  - Example

```
acl CSnets {
        140.113.235/24; 140.113.17/24; 140.113.209/24; 140.113.24/24;
};
acl NCTUnets {
        140.113/16; 10.113/16; 140.126.237/24;
};
allow-transfer {localhost; CSnets; NCTUnets};
```

BIND Configuration – named.conf key

- The "key" statement
  - Define a encryption key used for authentication with a particular server
  - Syntax

```
key key-id {
        algorithm string;
        secret string;
}
```

  - Example:

```
key serv1-serv2 {
        algorithm hmac-md5;
        secret "ibkAlUA0XXAXDxWRTGeY+d4CGbOgOIr7n63eizJFHQo="
}
```

  - This key is used to
    - Sign DNS request before sending to target
    - Validate DNS response after receiving from target

BIND Configuration – named.conf include

- The "include" statement
  - Used to separate large configuration file
  - Another usage is used to separate cryptographic keys into a restricted permission file
  - Ex:

```
include "/etc/namedb/rndc.key";
```

```
-rw-r--r--  1 root  wheel  4947 Mar  3  2006 named.conf
-rw-r-----  1 bind  wheel    92 Aug 15  2005 rndc.key
```

  - If the path is relative
    - Relative to the directory option

BIND Configuration – named.conf option (1)

- The "option" statement
  - Specify global options
  - Some options may be overridden later for specific zone or server
  - Syntax:

```
options {
        option;
        option;
};
```

- There are about 50 options in BIND9
  - version "There is no version."; [real version num]
    - version.bind. 0 CH TXT "9.3.3"
    - version.bind. 0 CH TXT "There is no version."
  - directory "/etc/namedb/db";
    - Base directory for relative path and path to put zone data files

BIND Configuration – named.conf option (2)

- notify yes | no [yes]
  - Whether notify slave sever when relative zone data is changed
- also-notify 140.113.235.101; [empty]
  - Also notify this non-advertised NS server
- recursion yes | no [yes]
  - Recursive name server
- allow-recursion {address_match_list }; [all]
  - Finer granularity recursion setting
- check-names {master|slave|response action};
  - check hostname syntax validity
    - Letter, number and dash only
    - 64 characters for each component, and 256 totally
  - Action:
    - ignore: do no checking
    - warn: log bad names but continue
    - fail: log bad names and reject
  - default action
    - master fail
    - slave warn
    - response ignore

BIND Configuration – named.conf option (3)

- listen-on port ip_port address_match_list; [53, all]
  - NIC and ports that named listens for query
  - Ex: listen-on port 5353 {192.168.1/24;};
- query-source address ip_addr port ip_port; [random]
  - NIC and port to send DNS query
- forwarders {in_addr; …}; [empty]
  - Often used in cache name server
  - Forward DNS query if there is no answer in cache
- forward only | first; [first]
  - If forwarder does not response, queries for forward only server will fail
- allow-query address_match_list; [all]
  - Specify who can send DNS query to you
- allow-transfer address_match_list; [all]
  - Specify who can request zone transfer of your zone data
- blackhole address_match_list; [empty]
  - Reject queries and would never ask them for answers

BIND Configuration – named.conf option (4)

- transfer-format one-answer | many-answers; [many-answers]
  - Ways to transfer data records from master to slave
  - How many data records in single packet
  - Added in BIND 8.1
- transfers-in num; [10]
- transfers-out num; [10]
  - Limit of the number of inbound and outbound zone transfers concurrently
- transfers-per-ns num; [2]
  - Limit of the inbound zone transfers concurrently from the same remote server
- transfer-source IP-address;
  - IP of NIC used for inbound transfers

BIND Configuration – named.conf server

- The "server" statement
  - Tell named about the characteristics of its remote peers
  - Syntax

```
server ip_addr {
        bogus no|yes;
        provide-ixfr yes|no;        (for master)
        request-ixfr yes|no;        (for slave)
        transfers num;
        transfer-format many-answers|one-answer;
        keys { key-id; key-id};
};
```

  - ixfr
    - Incremental zone transfer
  - transfers
    - Limit of number of concurrent inbound zone transfers from that server
    - Server-specific transfers-in
  - keys
    - Any request sent to the remote server is signed with this key

BIND Configuration – named.conf zone (1)

- The "zone" statement
  - Heart of the named.conf that tells named about the zones that it is authoritative
  - zone statement format varies depending on roles of named
    - Master or slave
  - The zone file is just a collection of DNS resource records
  - Basically

```
Syntax:
zone "domain_name" {
        type master | slave| stub;
        file "path";
        masters {ip_addr; ip_addr;};
        allow-query {address_match_list};       [all]
        allow-transfer { address_match_list};   [all]
        allow-update {address_match_list};      [empty]
};
```

allow-update cannot be used for a slave zone

BIND Configuration – named.conf zone (2)

- Master server zone configuration

```
zone "ce.nctu.edu.tw" IN {
    type master;
    file "named.hosts";
    allow-query { any; };
    allow-transfer { localhost; CS-DNS-Servers; };
    allow-update { none; };
};
```

- Slave server zone configuration

```
zone "cs.nctu.edu.tw" IN {
    type slave;
    file "cs.hosts";
    masters { 140.113.235.107; };
    allow-query { any; };
    allow-transfer { localhost; CS-DNS-Servers; };
};
```

BIND Configuration – named.conf zone (3)

- Forward zone and reverse zone

```
zone "cs.nctu.edu.tw" IN {
    type master;
    file "named.hosts";
    allow-query { any; };
    allow-transfer { localhost; CS-DNS-Servers; };
    allow-update { none; };
};
```

```
zone "235.113.140.in-addr.arpa" IN {
    type master;
    file "named.235.rev";
    allow-query { any; };
    allow-transfer { localhost; CS-DNS-Servers; };
    allow-update { none; };
};
```

BIND Configuration – named.conf zone (4)

- Example
  - In named.hosts, there are plenty of A or CNAME records

```
...
bsd1            IN      A       140.113.235.131
csbsd1          IN      CNAME   bsd1
bsd2            IN      A       140.113.235.132
bsd3            IN      A       140.113.235.133
bsd4            IN      A       140.113.235.134
bsd5            IN      A       140.113.235.135
...
```

  - In named.235.rev, there are plenty of PTR records

```
...
131.235.113.140     IN     PTR    bsd1.cs.nctu.edu.tw.
132.235.113.140     IN     PTR    bsd2.cs.nctu.edu.tw.
133.235.113.140     IN     PTR    bsd3.cs.nctu.edu.tw.
134.235.113.140     IN     PTR    bsd4.cs.nctu.edu.tw.
135.235.113.140     IN     PTR    bsd5.cs.nctu.edu.tw.
...
```

BIND Configuration – named.conf zone (5)

- Setting up root hint
  - A cache of where are the DNS root servers

```
zone "." IN {
    type hint;
    file "named.root";
};
```

- Setting up forwarding zone
  - Forward DNS query to specific name server, bypassing the standard query path

```
zone "nctu.edu.tw" IN {
    type forward;
    forward first;
    forwarders { 140.113.250.135; 140.113.1.1; };
};

zone "113.140.in-addr.arpa" IN {
    type forward;
    forward first;
    forwarders { 140.113.250.135; 140.113.1.1; };
};
```

BIND Configuration – named.conf view (1)

- The "view" statement
  - Create a different view of DNS naming hierarchy for internal machines
    - Restrict the external view to few well-known servers
    - Supply additional records to internal users
  - Also called "split DNS"
  - In-order processing
    - Put the most restrictive view first
  - All-or-nothing
    - All zone statements in your named.conf file must appear in the content of view

BIND Configuration – named.conf view (2)

- Syntax

```
view view-name {
        match_clients {address_match_list};
        view_options;
        zone_statement;
};
```

- Example

```
view "internal" {
        match-clients {our_nets;};
        recursion yes;
        zone "cs.nctu.edu.tw" {
        type master;
        file "named-internal-cs";
        };
};
view "external" {
 match-clients {any;};
        recursion no;
        zone "cs.nctu.edu.tw" {
        type master;
        file "named-external-cs";
        };
};
```

BIND Configuration – named.conf controls

- The "controls" statement
  - Specify how the named server listens for control message
  - Syntax

```
controls {
        inet ip_addr allow {address_match_list} keys {key-id;};
};
```

  - Example:

```
include "/etc/named/rndc.key";
controls {
        inet 127.0.0.1  allow {127.0.0.1;}   keys {rndc_key;};
}
```

```
key "rndc_key" {
    algorithm       hmac-md5;
    secret "GKnELuie/G99NpOC2/AXwA==";
};
```

```
SYNOPSIS
       rndc [-c config-file] [-k key-file] [-s server] [-p port] [-V]
            [-y key_id] {command}
```

BIND Configuration – rndc

- RNDC – remote name daemon control
  - reload, restart, status, dumpdb, …..
  - rndc-confgen

```
# Start of rndc.conf
key "rndc-key" {
        algorithm hmac-md5;
        secret "ayVEG7gJJdx+AMhA8+9jbg==";
};

options {
        default-key "rndc-key";
        default-server 127.0.0.1;
        default-port 953;
};
# End of rndc.conf
```

```
SYNOPSIS
       rndc [-c config-file] [-k key-file] [-s server] [-p port] [-V]
            [-y key_id] {command}
```

Updating zone files

- Master
  - Edit zone files
    - Serial number
    - Forward and reverse zone files for single IP
  - Do “rndc reload”
    - "notify" is on, slave will be notify about the change
    - "notify" is off, refresh timeout, or do "rndc reload" in slave
- Zone transfer
  - DNS zone data synchronization between master and slave servers
  - AXFR (all zone data are transferred at once, before BIND8.2)
  - IXFR (incremental updates zone transfer)
  - TCP port 53

Dynamic Updates

- The mappings of name-to-address are relatively stable
- DHCP will dynamically assign IP addresses to the hosts
  - Hostname-based logging or security measures become very difficulty

```
dhcp-host1.domain       IN   A    192.168.0.1
dhcp-host2.domain       IN   A    192.168.0.2
```

- Dynamic updates
  - BIND allows the DHCP daemon to notify the updating RR contents
  - Using allow-update
  - nsupdate
  - DDNS – dynamic DNS

Non-byte boundary (1)

- In normal reverse configuration:
  - named.conf will define a zone statement for each reverse subnet zone and
  - Your reverse db will contains lots of PTR records
  - Example:

```
zone "1.168.192.in-addr.arpa." {
    type master;
    file "named.rev.1";
    allow-query {any;};
    allow-update {none;};
    allow-transfer {localhost;};
};
```

```
$TTL    3600
$ORIGIN 1.168.192.in-addr.arpa.
@       IN      SOA     chwong.csie.net chwong.chwong.csie.net.   (
                        2007050401      ; Serial
                        3600            ; Refresh
                        900             ; Retry
                        7D              ; Expire
                        2H )            ; Minimum
        IN      NS      ns.chwong.csie.net.
254     IN      PTR     ns.chwong.csie.net.
1       IN      PTR     www.chwong.csie.net.
2       IN      PTR     ftp.chwong.csie.net.
…
```

Non-byte boundary (2)

- What if you want to delegate 192.168.2.0 to another sub-domain
  - Parent
    - **Remove** forward db about 192.168.2.0/24 network
      - Ex:

        pc1.chwong.csie.net. IN A 192.168.2.35
        pc2.chwong.csie.net. IN A 192.168.2.222
        …
    - **Remove** reverse db about 2.168.192.in-addr.arpa
      - Ex:

        35.2.168.192.in-addr.arpa. IN PTR pc1.chwong.csie.net.
        222.2.168.192.in-addr.arpa. IN PTR pc2.chwong.csie.net.
        …
    - Add glue records about the name servers of sub-domain
      - Ex: in zone db of "chwong.csie.net"

        sub1 IN NS ns.sub1.chwong.csie.net.
        ns.sub1 IN A 192.168.2.1
      - Ex: in zone db of "168.192.in-addr.arpa."

        2 IN NS ns.sub1.chwong.csie.net.
        1.2 IN PTR ns.sub1.chwong.csie.net

Non-byte boundary (3)

- What if you want to delegate 192.168.3.0 to four sub-domains (a /26 network)
  - 192.168.3.0 ~ 192.168.3.63
    - ns.sub1.chwong.csie.net.
  - 192.168.3.64 ~ 192.168.3.127
    - ns.sub2.chwong.csie.net.
  - 192.168.3.128 ~ 192.168.3.191
    - ns.sub3.chwong.csie.net.
  - 192.168.3.192 ~ 192.168.3.255
    - ns.sub4.chwong.csie.net.

- It is easy for forward setting
  - In zone db of chwong.csie.net

sub1	IN	NS	ns.sub1.chwong.csie.net.
ns.sub1	IN	A	1921.68.3.1
sub2	IN	NS	ns.sub2.chwong.csie.net.
ns.sub2	IN	A	192.168.3.65
...			

Non-byte boundary (4)

- Non-byte boundary reverse setting
  - Method1

```
$GENERATE 0-63      $.3.168.192.in-addr.arpa. IN  NS    ns.sub1.chwong.csie.net.
$GENERATE 64-127    $.3.168.192.in-addr.arpa. IN  NS    ns.sub2.chwong.csie.net.
$GENERATE 128-191   $.3.168.192.in-addr.arpa. IN  NS    ns.sub3.chwong.csie.net.
$GENERATE 192-255   $.3.168.192.in-addr.arpa. IN  NS    ns.sub4.chwong.csie.net.
```

And

```
zone "1.3.168.192.in-addr.arpa." {
        type master;
        file "named.rev.192.168.3.1";
};

; named.rev.192.168.3.1
@   IN  SOA     sub1.chwong.csie.net. root.sub1.chwong.csie.net. (1;3h;1h;1w;1h)
    IN  NS      ns.sub1.chwong.csie.net.
```

Non-byte boundary (5)

- Method2

```
$ORIGIN  3.168.192.in-addr.arpa.
$GENERATE  1-63       $      IN  CNAME   $.0-63.3.168.192.in-addr.arpa.
0-63.3.168.192.in-addr.arpa.          IN  NS          ns.sub1.chwong.csie.net.
$GENERATE  65-127     $      IN  CNAME   $.64-127.3.168.192.in-addr.arpa.
64-127.3.168.192.in-addr.arpa.        IN  NS          ns.sub2.chwong.csie.net.
$GENERATE  129-191    $      IN  CNAME   $.128-191.3.168.192.in-addr.arpa.
128-191.3.168.192.in-addr.arpa.       IN  NS          ns.sub3.chwong.csie.net.
$GENERATE  193-255    $      IN  CNAME   $.192-255.3.168.192.in-addr.arpa.
192-255.3.168.192.in-addr.arpa.       IN  NS          ns.sub4.chwong.csie.net.

zone "0-63.3.168.192.in-addr.arpa." {
        type master;
        file "named.rev.192.168.3.0-63";
};
```

```
        ; named.rev.192.168.3.0-63
        @   IN   SOA   sub1.chwong.csie.net. root.sub1.chwong.csie.net. (1;3h;1h;1w;1h)
            IN   NS    ns.sub1.chwong.csie.net.
1     IN   PTR  www.sub1.chwong.csie.net.
2     IN   PTR  abc.sub1.chwong.csie.net.
…
```

BIND Debugging and Logging

Logging (1)

- Terms
  - Channel
    - A place where messages can go
    - Ex: syslog, file or /dev/null
  - Category
    - A class of messages that named can generate
    - Ex: answering queries or dynamic updates
  - Module
    - The name of the source module that generates the message
  - Facility
    - syslog facility name
  - Severity
    - Priority in syslog
- Logging configuration
  - Define what are the channels
  - Specify where each message category should go
- When a message is generated
  - It is assigned a “category”, a “module”, a “severity”
  - It is distributed to all channels associated with its category

Logging (2)

- The "logging" statement
  - Either "file" or "syslog" in channel sub-statement
    - size:
      - ex: 2048, 100k, 20m, 15g, unlimited, default
    - facility:
      - ex: daemon, local0 ~ local7
    - severity:
      - critical, error, warning, notice, info, debug (with an optional numeric level), dynamic
      - Dynamic is recognized and matches the server's current debug level

```
logging {
    channel_def;
    channel_def;
    …
    category category_name {
        channel_name;
        channel_name;
        …
    };
};
```

```
channel channel_name {
    file path [versions num|unlimited] [size siznum];
    syslog facility;

    severity severity;
    print-category yes|no;
    print-severity yes|no;
    print-time yes|no;
};
```

Logging (3)

- Predefined channels

default_syslog	Sends severity info and higher to syslog with facility daemon
default_debug	Logs to file "named.run", severity set to dynamic
default_stderr	Sends messages to stderr or named, severity info
null	Discards all messages

- Available categories

default	Categories with no explicit channel assignment
general	Unclassified messages
config	Configuration file parsing and processing
queries/client	A short log message for every query the server receives
dnssec	DNSSEC messages
update	Messages about dynamic updates
xfer-in/xfer-out	zone transfers that the server is receiving/sending
db/database	Messages about database operations
notify	Messages about the "zone changed" notification protocol
security	Approved/unapproved requests
resolver	Recursive lookups for clients

Logging (4)

- Example of logging statement

```
logging {
    channel security-log {
        file "/var/named/security.log" versions 5 size 10m;
        severity info;
        print-severity yes;
        print-time yes;
    };
    channel query-log {
        file "/var/named/query.log" versions 20 size 50m;
        severity info;
        print-severity yes;
        print-time yes;
    };
    category default        { default_syslog; default_debug; };
    category general        { default_syslog; };
    category security       { security-log; };
    category client         { query-log; };
    category queries        { query-log; };
    category dnssec         { security-log; };
};
```

Debug

- Named debug level
  - From 0 (debugging off) ~ 11 (most verbose output)
  - % named –d2 (start named at level 2)
  - % rndc trace (increase debugging level by 1)
  - % rndc trace 3 (change debugging level to 3)
  - % rndc notrace (turn off debugging)

- Debug with "logging" statement
  - Define a channel that include a severity with "debug" keyword
    - Ex: severity debug 3
    - All debugging messages up to level 3 will be sent to that particular channel

Tools

Tools
– nslookup

- Interactive and Non-interactive
  - Non-Interactive
    - % nslookup cs.nctu.edu.tw.
    - % nslookup –type=mx cs.nctu.edu.tw.
    - % nslookup –type=ns cs.nctu.edu.tw. 140.113.1.1
  - Interactive
    - % nslookup
    - > set all
    - > set type=any
    - > server host
    - > lserver host
    - > set debug
    - > set d2

```
csduty [/u/dcs/94/9455832] -chwong- nslookup
> set all
Default server: 140.113.235.107
Address: 140.113.235.107#53
Default server: 140.113.235.103
Address: 140.113.235.103#53

Set options:
  novc                  nodebug          nod2
  search                recurse
  timeout = 0           retry = 3        port = 53
  querytype = A         class = IN
  srchlist = cs.nctu.edu.tw/csie.nctu.edu.tw
>
```

Tools
– dig

- Usage
  - % dig cs.nctu.edu.tw
  - % dig cs.nctu.edu.tw mx
  - % dig @ns.nctu.edu.tw cs.nctu.edu.tw mx
  - % dig -x 140.113.209.3
    - ➢ Reverse query
- Find out the root servers
  - % dig @a.root-servers.net . ns

Tools
– host

- host command
  - % host cs.nctu.edu.tw.
  - % host –t mx cs.nctu.edu.tw.
  - % host 140.113.1.1
  - % host –v 140.113.1.1

DNS Security

Security
– named.conf security configuration

- Security configuration

Feature	Config. Statement	comment
allow-query	options, zone	Who can query
allow-transfer	options, zone	Who can request zone transfer
allow-update	zone	Who can make dynamic updates
blackhole	options	Which server to completely ignore
bogus	server	Which servers should never be queried

Security – With TSIG (1)

- TSIG (Transaction SIGnature)
  - Developed by IETF (RFC2845)
  - Symmetric encryption scheme to sign and validate DNS requests and responses between servers
  - Algorithm in BIND9
    - HMAC-MD5, HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512
  - Usage
    - Prepare the shared key with dnssec-keygen
    - Edit "key" statement
    - Edit "server" statement to use that key
    - Edit "zone" statement to use that key with:
      - allow-query
      - allow-transfer
      - allow-update

Security – With TSIG (2)

- TSIG example (dns1 with dns2)

1. % dnssec-keygen –a HMAC-MD5 –b 128 –n HOST cs

```
% dnssec-keygen -a HMAC-MD5 -b 128 -n HOST cs
Kcs.+157+35993
% cat Kcs.+157+35993.key
cs.  IN KEY 512 3 157 oQRab/QqXHVhkyXi9uu8hg==
```

```
% cat Kcs.+157+35993.private
Private-key-format: v1.2
Algorithm: 157 (HMAC_MD5)
Key: oQRab/QqXHVhkyXi9uu8hg==
```

2. Edit /etc/named/dns1-dns2.key

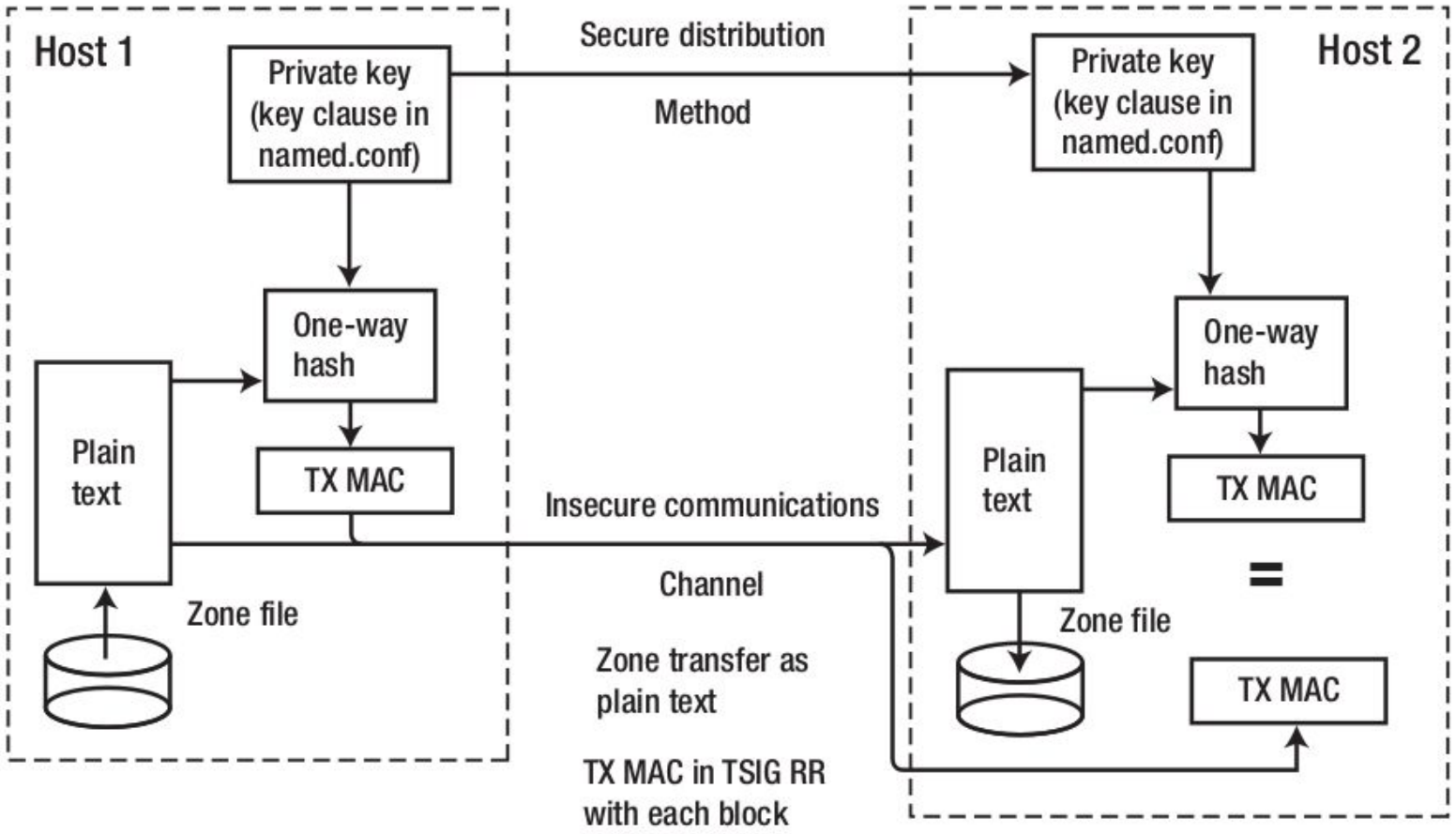
```
key dns1-dns2 {
    algorithm hmac-md5;
    secret "oQRab/QqXHVhkyXi9uu8hg==";
};
```

3. Edit both named.conf of dns1 and dns2
   - Suppose dns1 = 140.113.235.107 dns2 = 140.113.235.103

```
include "dns1-dns2.key";
server 140.113.235.103 {
    keys {dns1-dns2;};
};
```

```
include "dns1-dns2.key";
server 140.113.235.107 {
    keys {dns1-dns2;};
};
```

Security – With TSIG (3)



Security
– Securing zone transfer

- Securing zone transfer with ACL

```
zone “example.com” in {
        type master;
        file “host”;
        allow-transfer { trusted; 192.168.10.2; };
};
```

Security
– Securing zone transfer

- Securing zone transfer with Key (***Master***)

```
include "keys/example.com.key"; // include the key clause
// server clause references the key clause included above
server 10.1.2.3 {
   keys {"example.com";}; // name used in key clause
};
....
zone "example.com" in{
   type master;
   file "master.example.com";
   // allow transfer only if key (TSIG) present
   allow-transfer {key "example.com";};
};
....
```

Security
– Securing zone transfer

- Securing zone transfer with TSIG (***Slave***)

```
// named.conf example.com slave fragment
options {
    ....
    directory "/var/named";
    dnssec-enable yes;
    ....
};
include "keys/example.com.key"; // include the key clause
server 10.1.2.5 {
    keys {"example.com";}; // name used in key clause
};
....
zone "example.com" in{
    type slave;
    file "slave.example.com";
    masters {10.1.2.5;};
};
```

Security
– Securing dynamic update

- Securing dynamic update with ACL

```
options {
....
};
....
zone "example.com in{
....
    allow-update {10.1.2.5;}; // this zone only
....
};
```

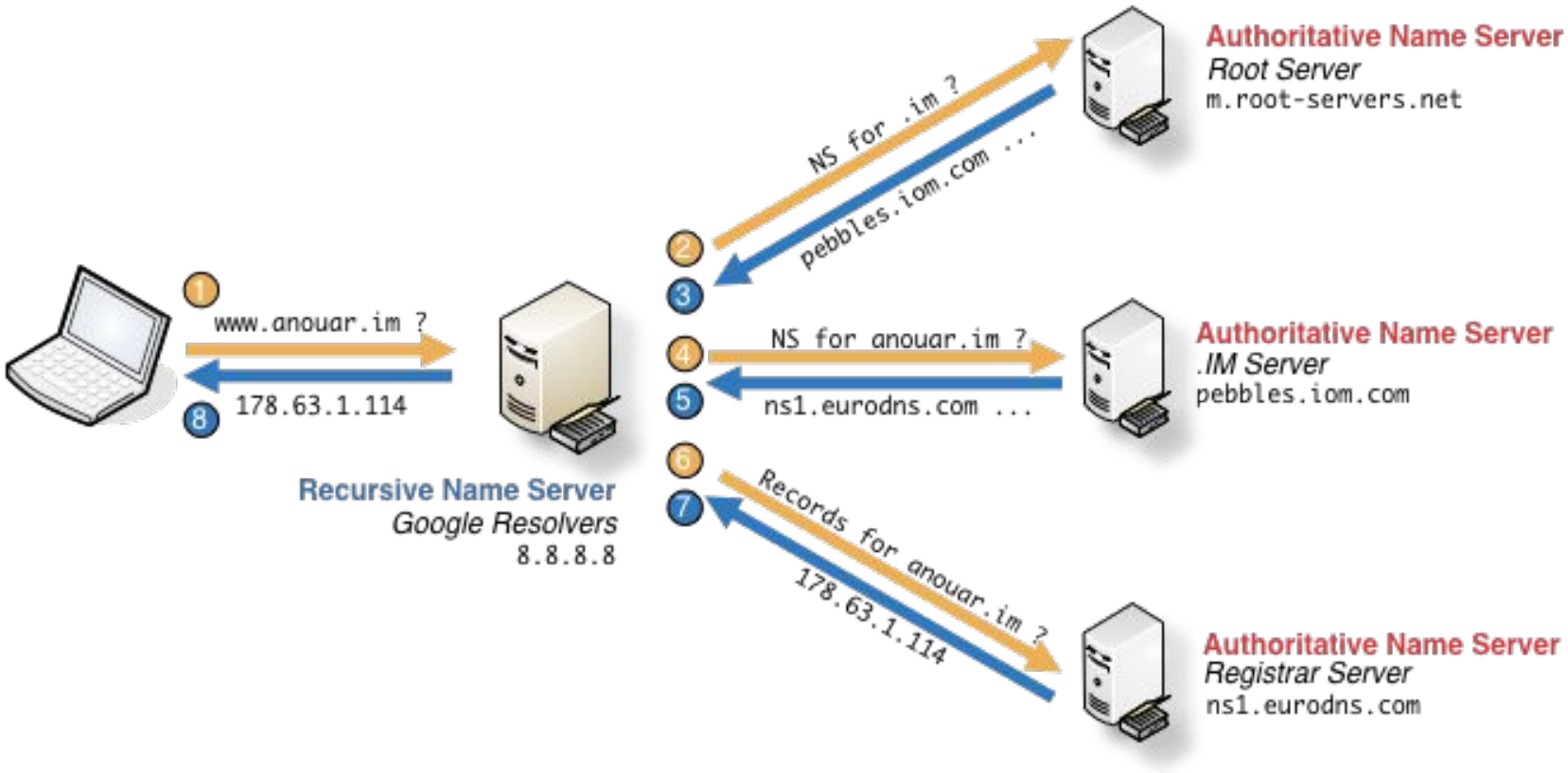
Security
– Securing dynamic update

- Securing dynamic update with TSIG

```
include "keys/example.com.key"; // include the key clause
server 10.1.2.3 {
   keys {"example.com";}; // name used in key clause
};
....
zone "example.com" in{
   type master;
   file "master.example.com";
   allow-update {key "example.com";};
};
....
zone "example.net" in{
   type master;
   file "master.example.net";
   update-policy { grant example.com subdomain example.net ANY;};
   update-policy { grant * self * A;};
   update-policy { grant update-mx name example.net MX;};
};
....
```

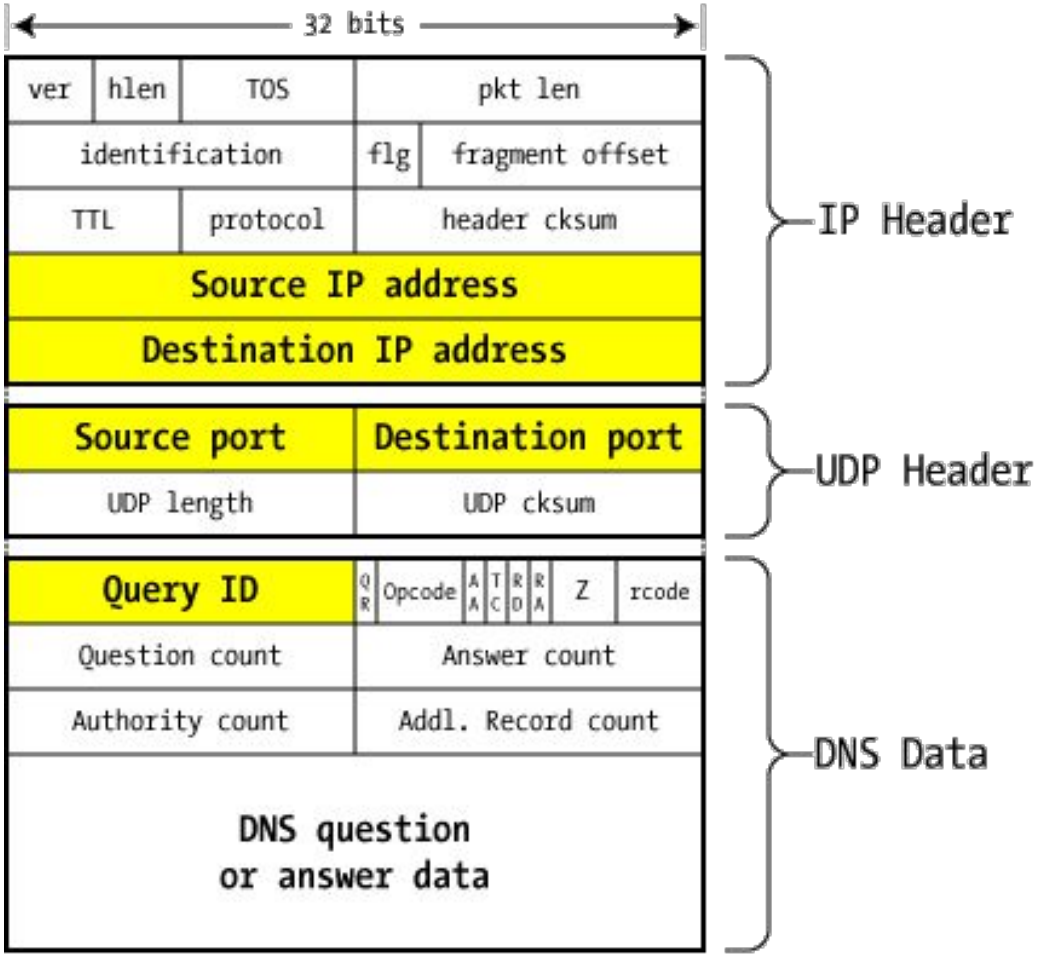
Security
– Cache poisoning

- A Normal Resolving Process



Security
– Cache poisoning

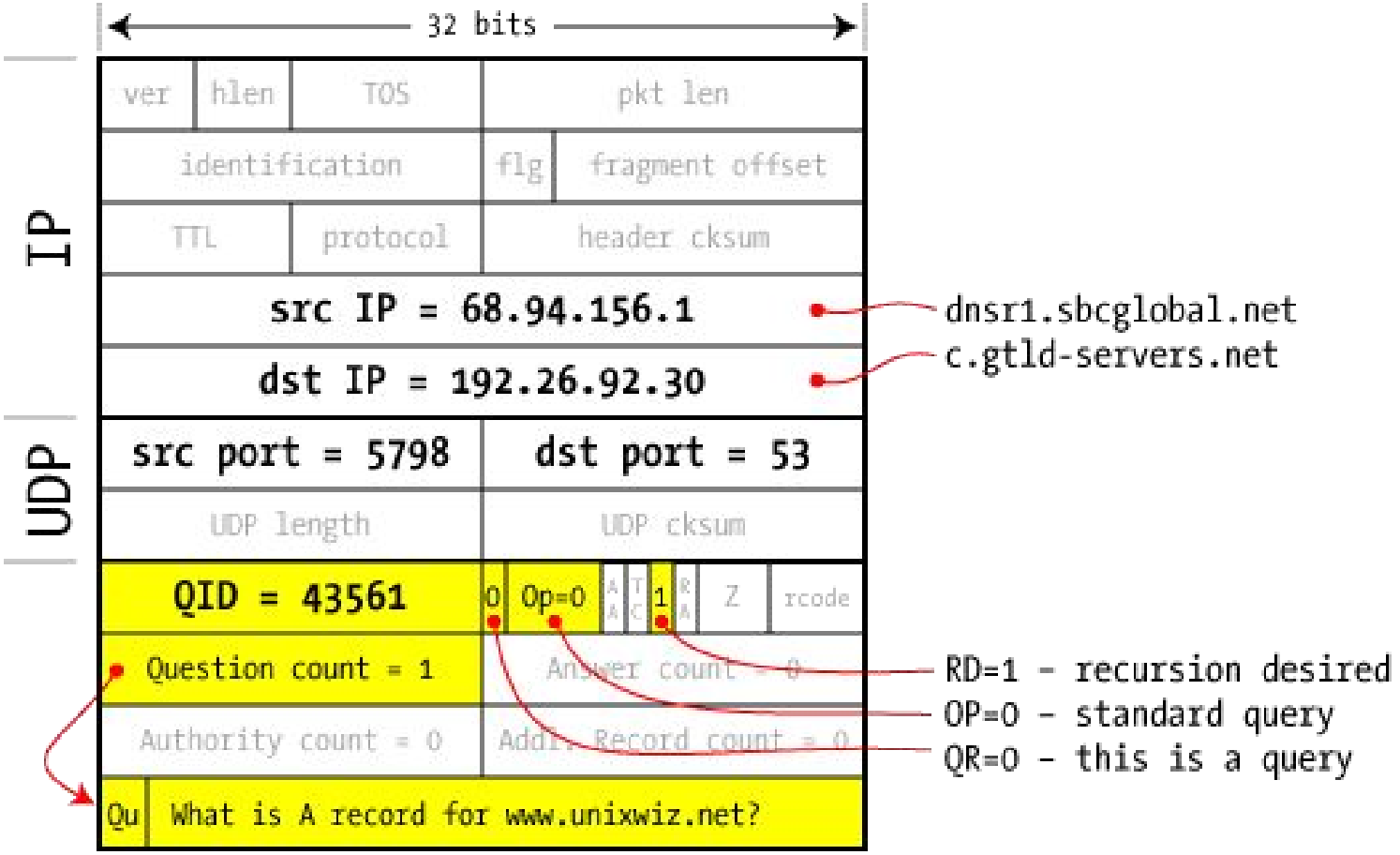
- DNS packet on the wire



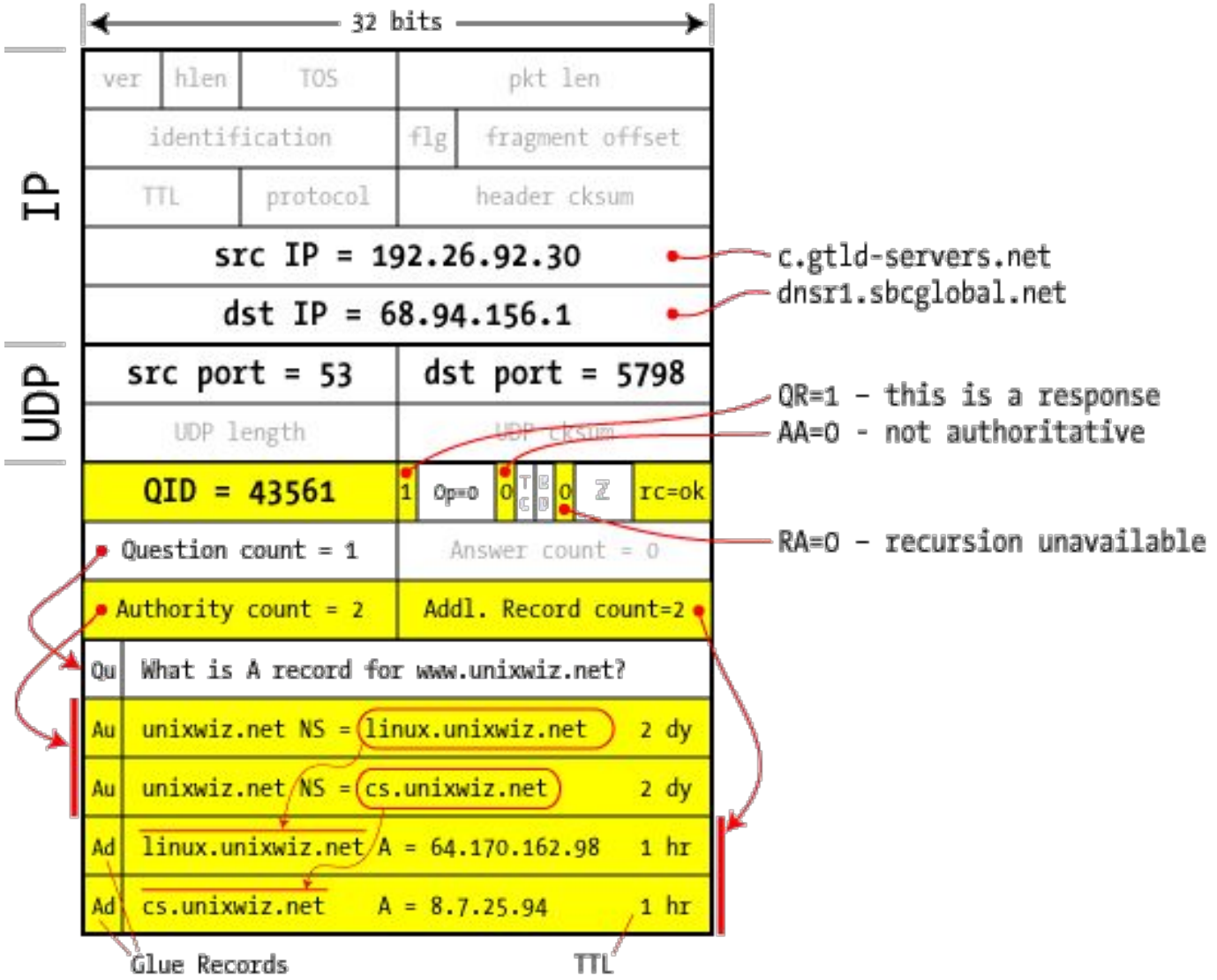
DNS packet on the wire

Security – Cache poisoning

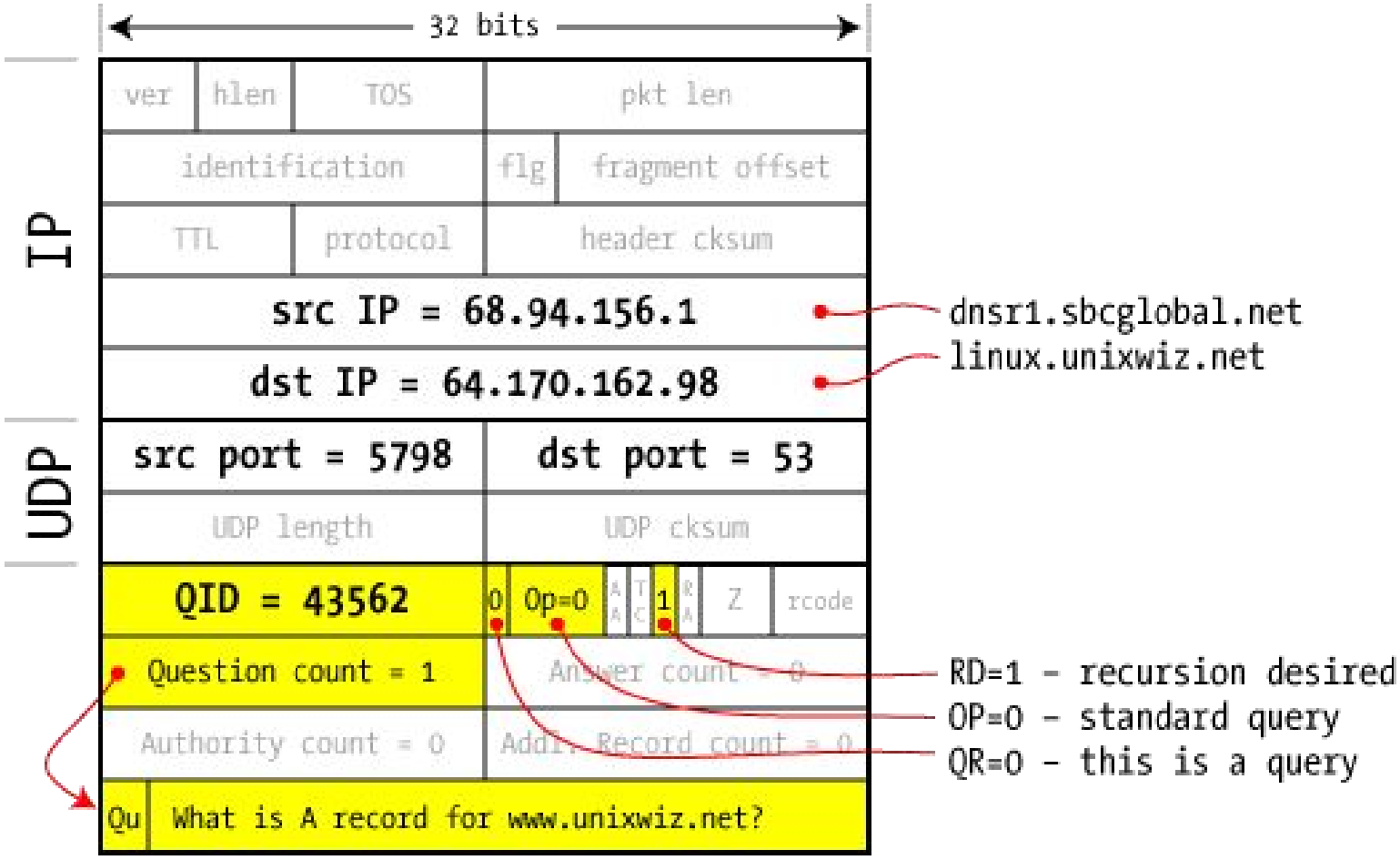
- Query from resolver to NS



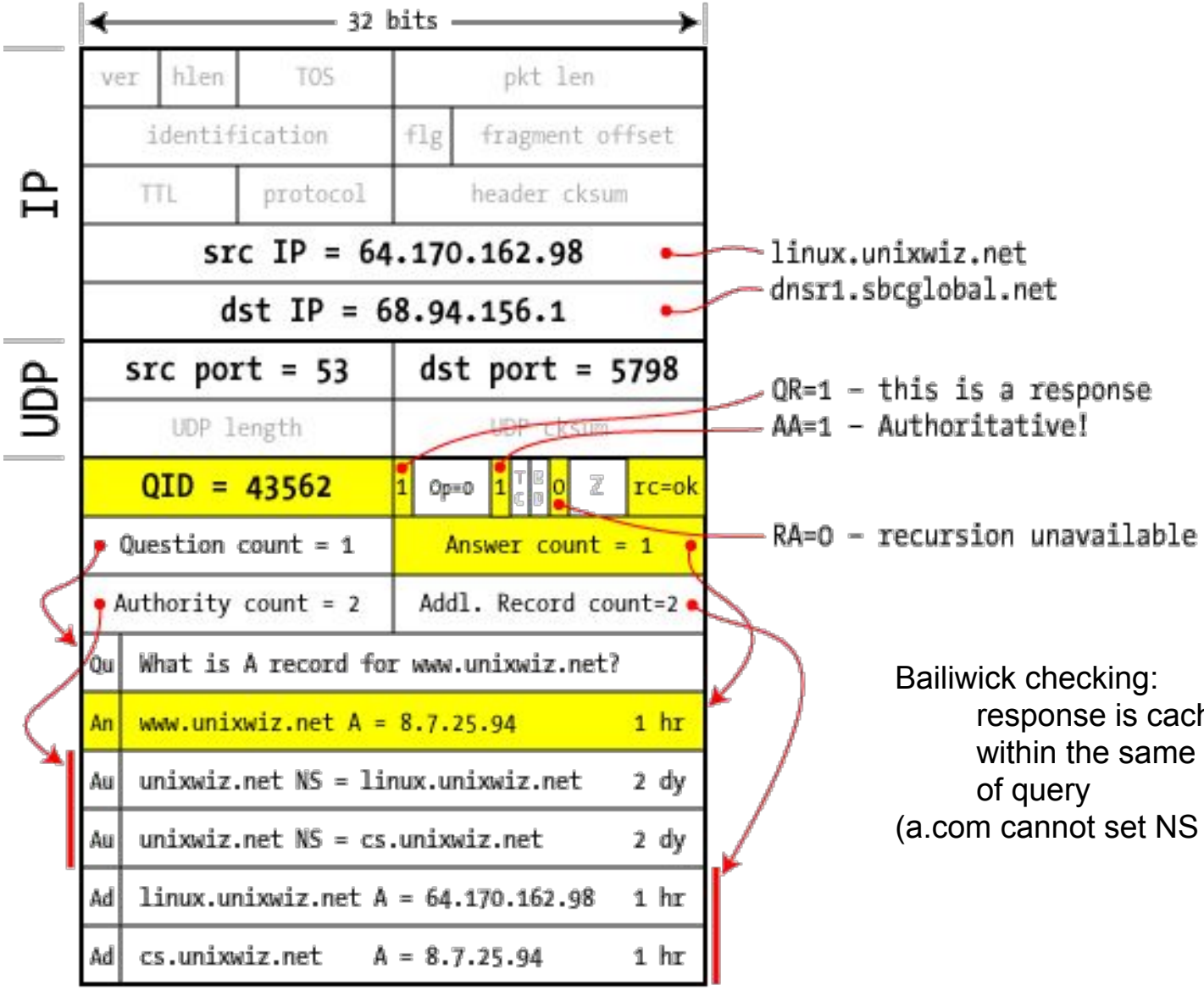
Security
– Cache poisoning



Security
– Cache poisoning



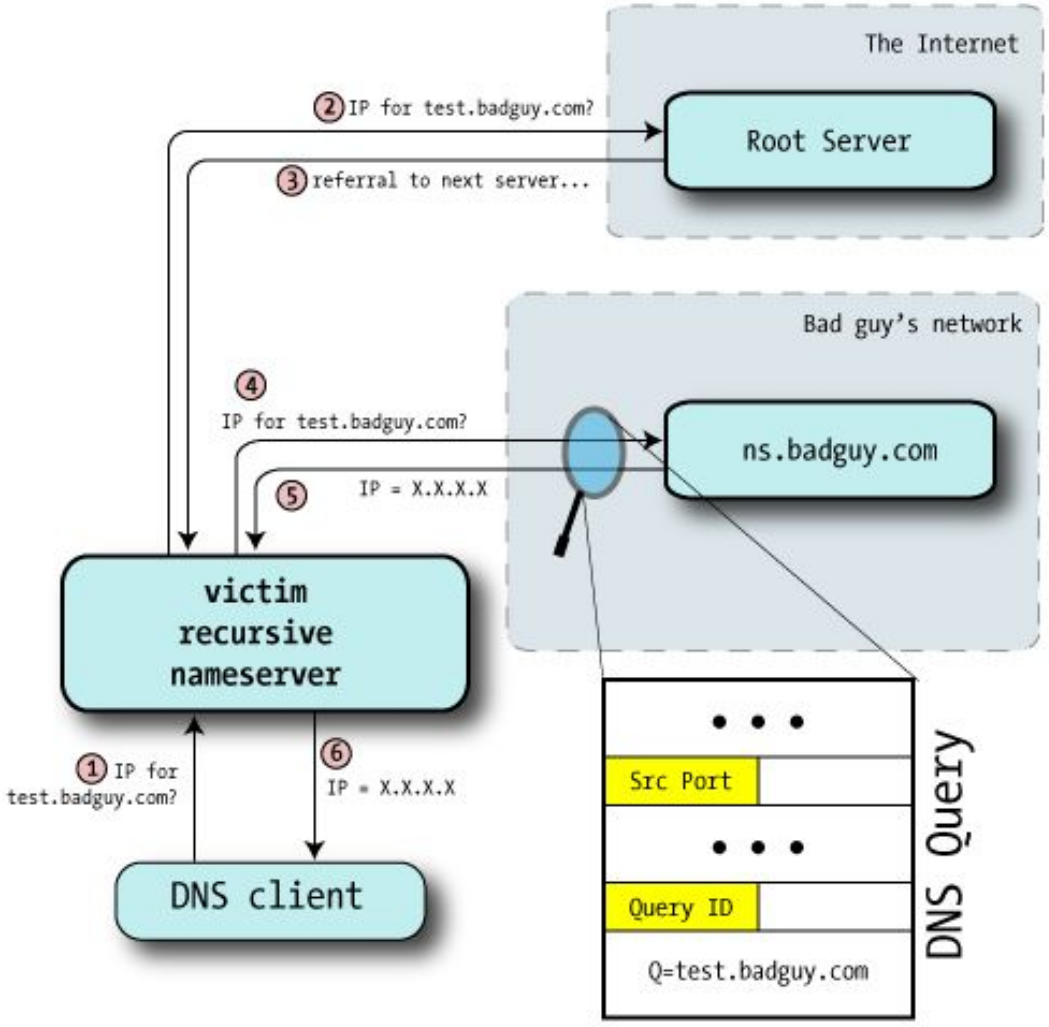
Security
– Cache poisoning



Bailiwick checking:
response is cached if it is within the same domain of query
(a.com cannot set NS for b.com)

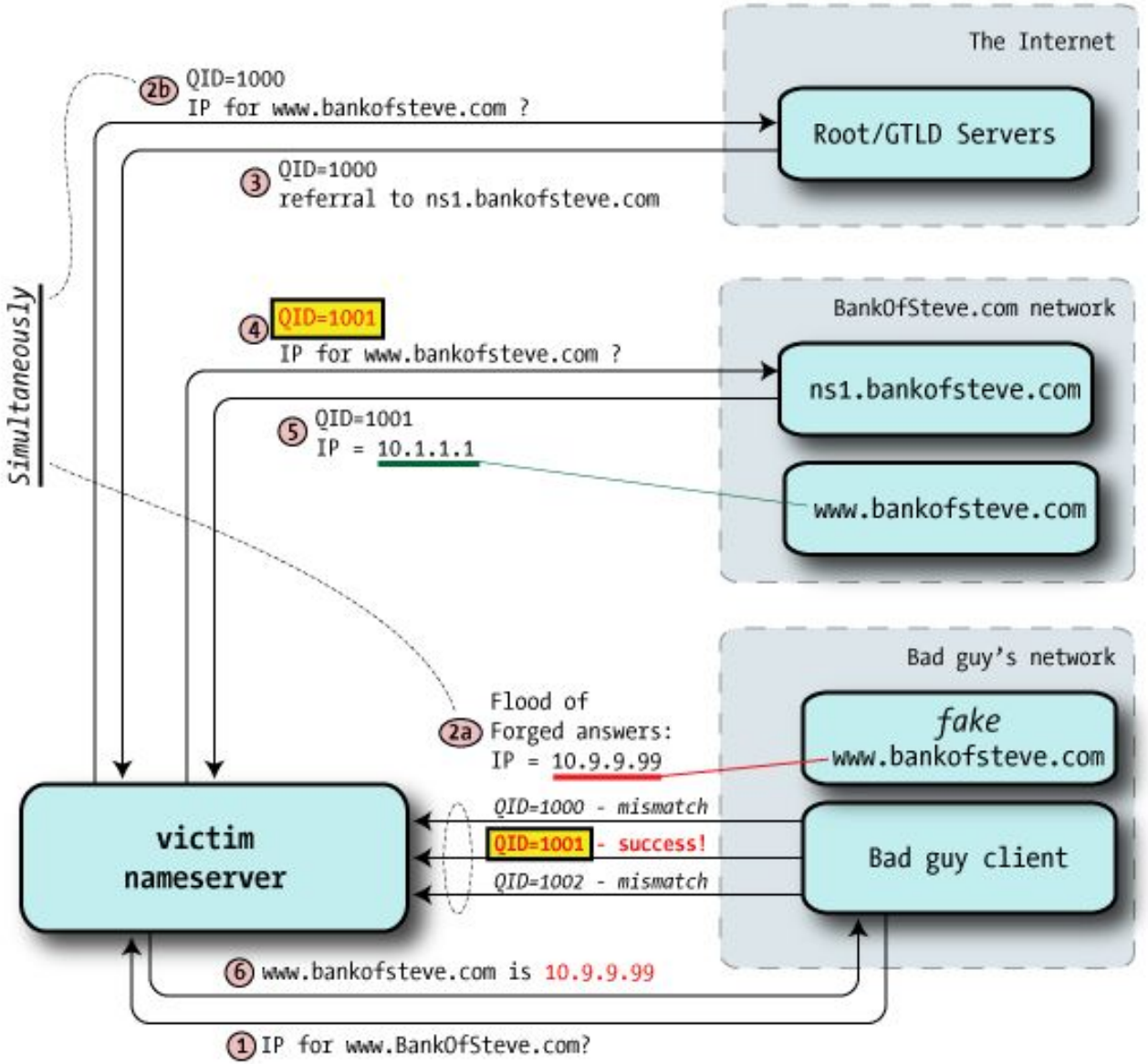
Security
– Cache poisoning

Guessing Query ID



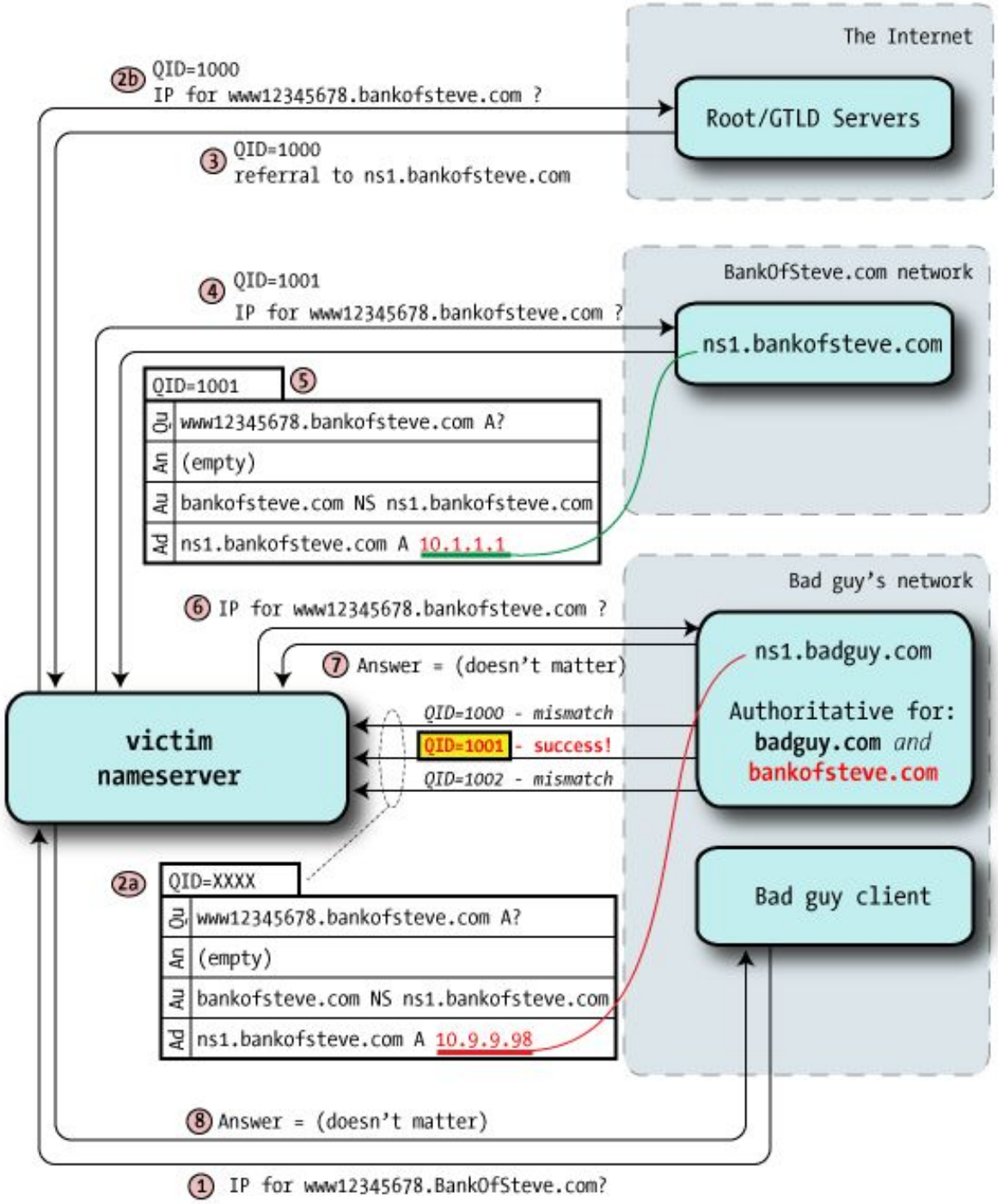
Security
– Cache poisoning

Flooding



Security
– Cache poisoning

- Kaminsky Attack
  - Poison cache for NS record instead
  - Take over all of second level domain


The Internet
Root/GTLD Servers
2b QID=1000
IP for www12345678.bankofsteve.com ?
3 QID=1000
referral to ns1.bankofsteve.com
BankOfSteve.com network
ns1.bankofsteve.com
4 QID=1001
IP for www12345678.bankofsteve.com ?
QID=1001
5
Qu www12345678.bankofsteve.com A?
An (empty)
Au bankofsteve.com NS ns1.bankofsteve.com
Ad ns1.bankofsteve.com A 10.1.1.1
Bad guy's network
6 IP for www12345678.bankofsteve.com ?
7 Answer = (doesn't matter)
ns1.badguy.com
Authoritative for:
badguy.com and
bankofsteve.com
victim
nameserver
QID=1000 - mismatch
QID=1001 - success!
QID=1002 - mismatch
Bad guy client
2a QID=XXXX
Qu www12345678.bankofsteve.com A?
An (empty)
Au bankofsteve.com NS ns1.bankofsteve.com
Ad ns1.bankofsteve.com A 10.9.9.98
8 Answer = (doesn't matter)
1 IP for www12345678.BankOfSteve.com?

Security
– Cache poisoning

- Defense
  - Randomized query ID
  - Randomized UDP port
  - **DNSSEC**
    - Cryptographically sign DNS responses

Security – DNSSEC

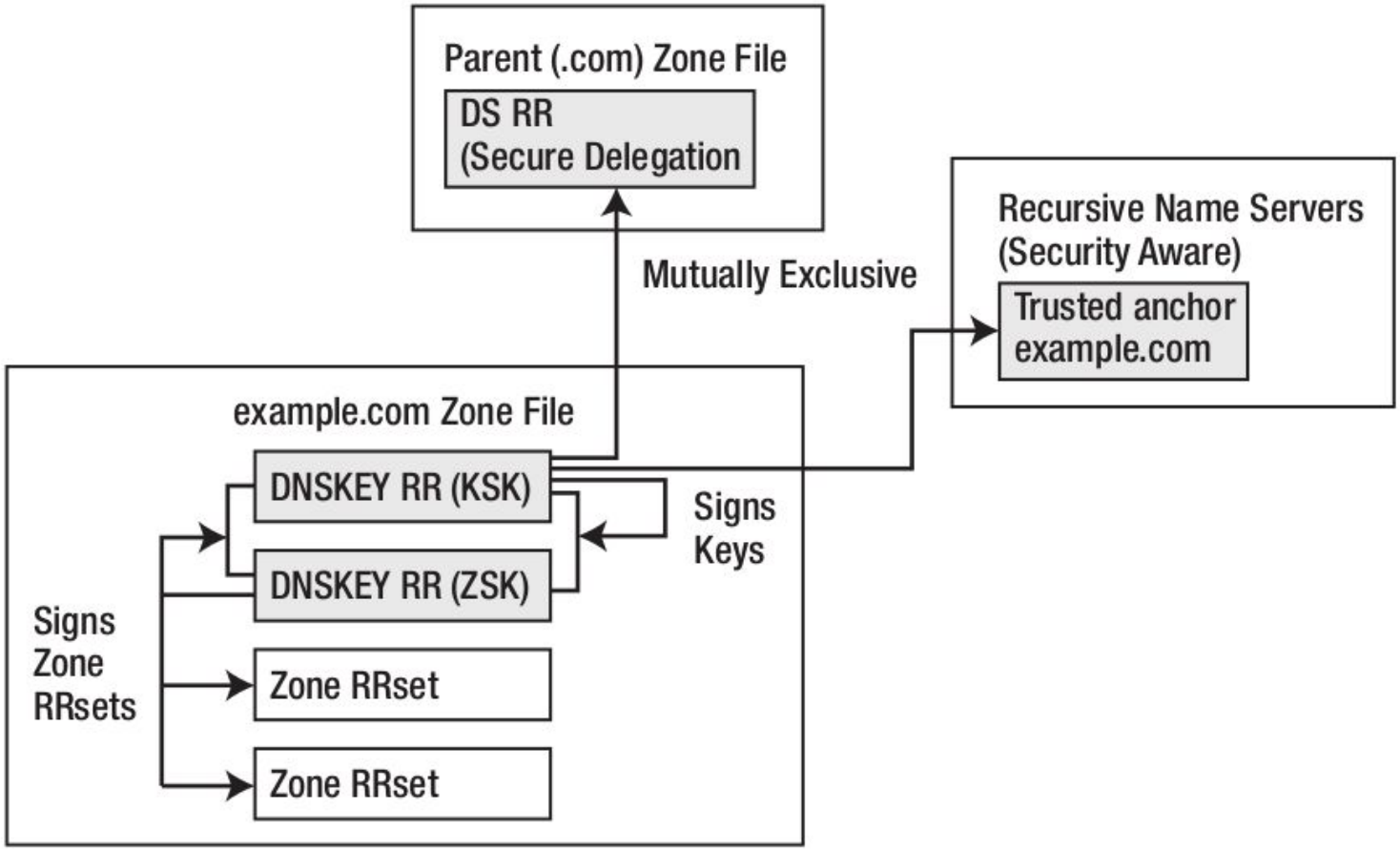
- What is DNSSEC?
  - Using Public-key crypto (asymmetric)
  - Follow the delegation of authority model
  - Data authenticity and integrity
    - Signing the RRSets with private key
    - Public DNSKEYs are published, used to verify RRSIGs
    - Children sign their zones with private key
      - The private key is authenticated by parent's signing hash(DS) of the child zone's key

Security – DNSSEC

- Resource Records
  - RRSIG
    - Crypto signatures for A, AAAA, NS, etc.
    - Tracks the type and number at each node.
  - NSEC/NSEC3
    - Confirms the NXDOMAIN response.
  - DNSKEY
    - Public keys for the entire zone.
    - Private side is used generate RRSIGs
  - DS Record
    - Handed up to parent zone to authenticate the NS record

Security – DNSSEC

- ZSK and KSK



Security – DNSSEC Implementation

- Generate ZSK (Zone signing key)

```
$dnssec-keygen -a rsasha256 -b 2048 -n zone \
        example.com
Kexample.com.+008+27228
```

- Generate KSK (Key signing key)

```
$dnssec-keygen -a rsasha256 -b 2048 -f KSK -n zone \
        example.com
Kexample.com.+008+34957
```

Security – DNSSEC Implementation

- In zone file

```
$TTL 86400 ; 1 day
$ORIGIN example.com.
@           IN SOA ns1.example.com. hostmaster.example.com. (
                           2010121500 ; serial
                           43200      ; refresh (12 hours)
                           600        ; retry (10 minutes)
                           604800     ; expire (1 week)
                           10800      ; nx (3 hours)
                           )
            IN  NS ns1.example.com.
            IN  NS ns2.example.com.
            IN  MX 10 mail.example.com.
            IN  MX 10 mail1.example.com.
_ldap._tcp  IN  SRV 5 2 235 www
ns1         IN  A  192.168.2.6
ns2         IN  A  192.168.23.23
www         IN  A  10.1.2.1
            IN  A  172.16.2.1
mail        IN  A  192.168.2.3
mail1       IN  A  192.168.2.4
$ORIGIN sub.example.com.
@           IN  NS ns3.sub.example.com.
            IN  NS ns4.sub.example.com.
ns3         IN  A  10.2.3.4 ; glue RR
ns4         IN  A  10.2.3.5 ; glue RR
$INCLUDE keys/Kexample.com.+008+34957.key ; KSK
$INCLUDE keys/Kexample.com.+008+27228.key ; ZSK
```

Security – DNSSEC Implementation

- Signing the zone

```
# dnssec-signzone -o example.com -t -k Kexample.com.+008+34957
master.example.com Kexample.com.+008+27228
Verifying the zone using the following alogoriths: RSASHA256
Algorithm: RSASHA256 KSKs: 1 active, 0 stand-by, 0 revoked
                     ZSKs: 1 active, 0 stand-by, 0 revoked
master.example.com.signed
Signatures generated:                       21
Signatures retained:                         0
Signatures dropped:                          0
Signatures successfully verified:            0
Signatures unsuccessfully verified:          0
Runtime in seconds:                      0.227
Signatures per second:                  92.327n
```

When signing the zone with only ZSK, just omit the -k parameter

Security – DNSSEC Implementation

- Signing the zone (example.com.signed)

```
; File written on Sat Dec 18 21:31:01 2010
; dnssec_signzone version 9.7.2-P2
example.com.  86400 IN SOA ns1.example.com. hostmaster.example.com. (
                          2010121500 ; serial
                          43200      ; refresh (12 hours)
                          600        ; retry (10 minutes)
                          604800     ; expire (1 week)
                          10800      ; minimum (3 hours)
                          )
                  86400    RRSIG  SOA 8 2 86400 20110118013101 (
                           20101219013101 27228 example.com.
                           Mnm5RaKEFAW4V5dRhP7OxLtGAFMb/Zsej2vH
                           mK5O7zHL+U2Hbx+arMMoA/aOxtp6JxpOFWM3
                           67VHclTjjGX9xf++6qvA65JHRNvKoZgXGtXI
                           VGG6ve8A8J9LRePtCKwo3WfhtLEMFsd1KI6o
                           JTViPzs3UDEqgAvy8rgtvwr8Oa8= )
                  86400   NS              ns1.example.com.
                  86400   NS              ns2.example.com.
                  86400   RRSIG   NS 8 2 86400 20110118013101 (
                           20101219013101 27228 example.com.
                           ubbRJV+DiNmgQITtncLOCjIw4cfB4qnC+DX8
                           ....
                           S78T5Fxh5SbLBPTBKmlKvKxcx6k= )
```

Security – DNSSEC Implementation

- Update the Zone clause to use the signed zone

```
zone "example.com" {
        type master;
        file "example.com.signed";
        masters {ip_addr; ip_addr;};
        allow-query {address_match_list};
        allow-transfer { address_match_list};
        allow-update {address_match_list};
};
```

Security – DNSSEC Implementation

- Create Trust Anchor

```
86400 DNSKEY 257 3 8 (
        5Jq6Dp+JyHNO3OHqgHv2KrRuvUOXV+8l

        ); key id = 34957
```

```
options {
   ....
   directory "/var/named";
   dnssec-enable yes;
   dnssec-validation yes;
      allow-recursion {10.2/16; 192.168.2/24;}; // recursion limits - closes resolver
   ....
};
trusted-keys{
   "example.com" 257 3 8 "5Jq6Dp+JyHNO3OHqgHv2KrRuvUOXV+8l
";
};
....
```

Security – DNSSEC Implementation

- Create Chain of Trust
  - Extract DNSKEY RR and use dnssec-dsfromkey
  - Add -g parameter when signing zone using dnssec-signzone
  - dnssec-signzone -g ….
    - ds-set.example.com
      - contains DS record that you should hand to parent

Security
– DNSSEC Implementation