

Internet Operation (7)

# DNS Operation

(Advanced Internet Technology-II by SOI Asia)

JINMEI, Tatuya

Toshiba Corporation / The KAME Project

jinmei@{isl.rdc.toshiba.co.jp, kame.net}

# Self-introduction

- Name: JINMEI, Tatuya
- Affiliate: R&D Center, Toshiba Corporation
- Working on/for/as
  - various fields on the Internet
    - IPv6, multicasting, DNS
  - the KAME Project
    - <http://www.kame.net/>
    - reference implementation of IPv6/IPsec
  - an "occasional developer" of BIND
    - mainly on IPv6, server performance
- Contact points
  - [jinmei@isl.rdc.toshiba.co.jp](mailto:jinmei@isl.rdc.toshiba.co.jp)
  - <http://www.jinmei.org/>

# Overview of this lecture

## □ Prerequisites

- a brief understanding of DNS, IP, and IPv6
- assume basic DNS notions/terminologies
  - (interrupts for questions are welcome)

## □ Contents

- a brief review of the DNS protocol
- notes on key points for solid operation
  - SOA, CNAME, IPv6
- introduction to BIND operation
- other operational issues

## □ Goal

- establish a base for your own operation
  - minimum understanding of protocol basics
  - hints for diagnosing problems

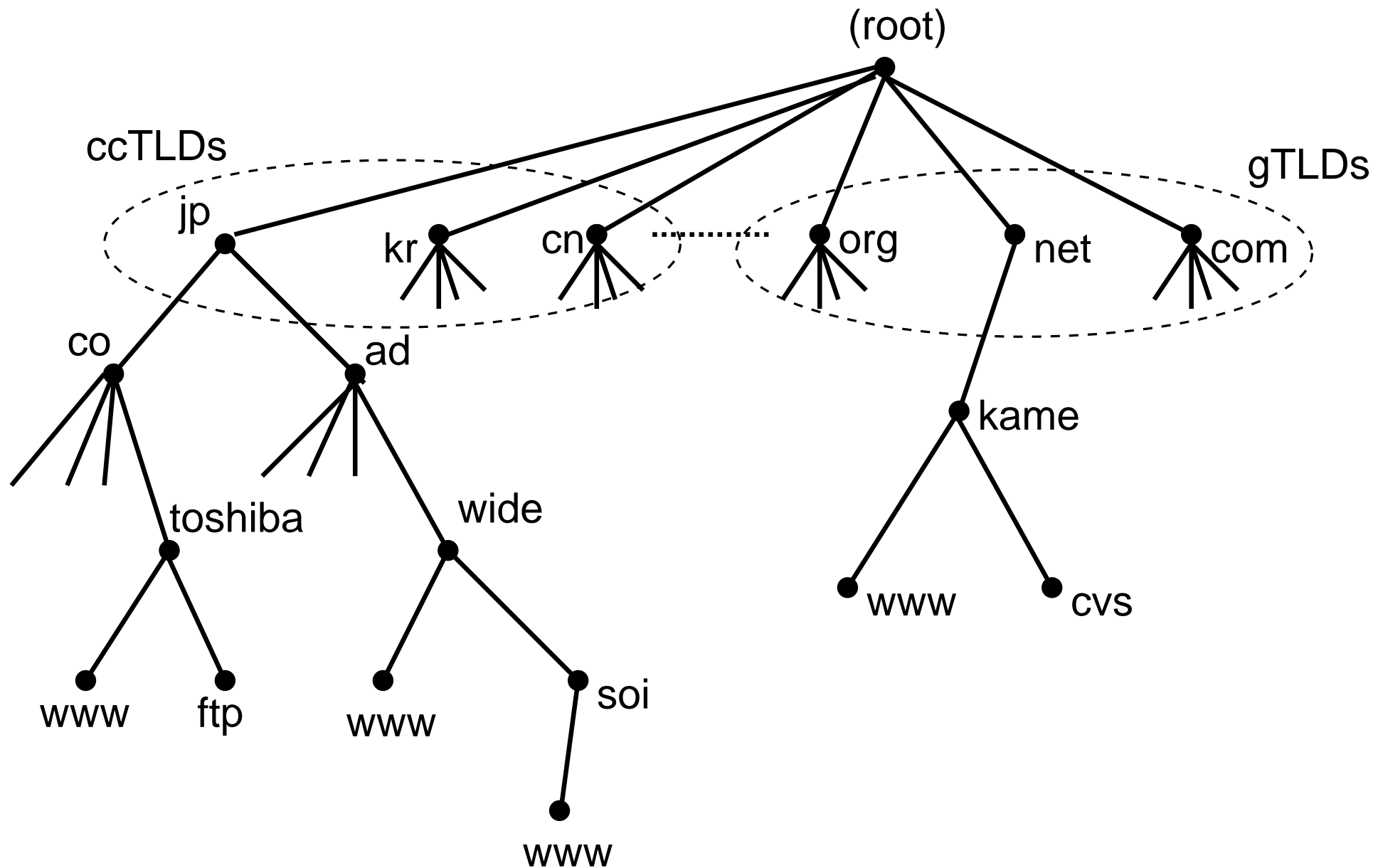
# A brief review of DNS

- Basic notion
- Resource records
- IPv6 related points

# A brief review of DNS (1/2)

- DNS (Domain Name System)
  - mapping between host names and IP addresses
  - www.toshiba.co.jp. <-> 211.7.133.18
  
- Hierarchical distributed database
  - coherent and load-balanced
  - tree-based structure with authority delegation
    - ▷ rooted at "root" zone
    - ▷ 2nd level: ccTLDs, gTLDs
  - efficient lookup by caching

# Domain Name Space Structure

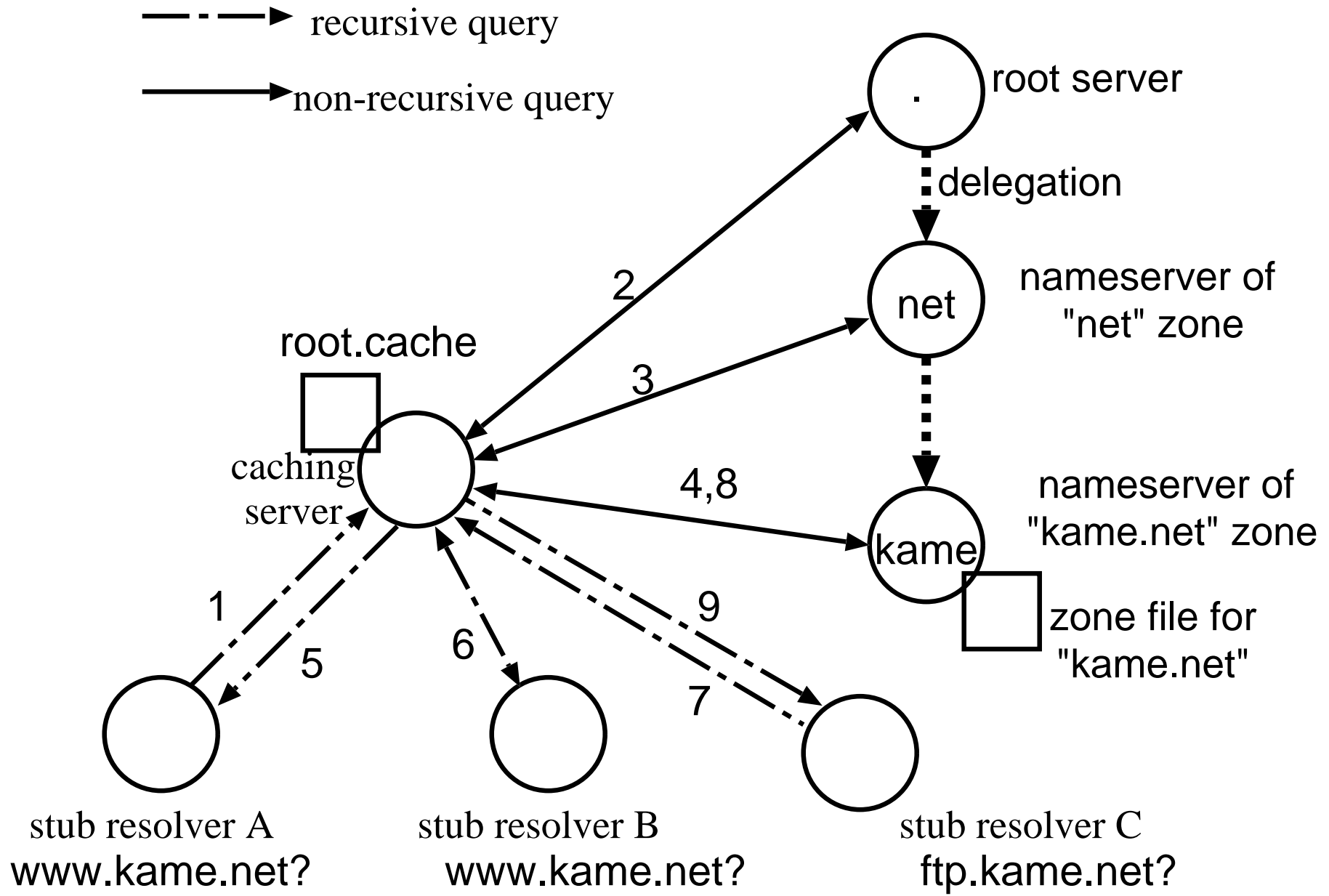


# A brief review of DNS (2/2)

## □ Server-client model

- authoritative servers manage DNS DB
  - ▷ aka "nameservers"
  - ▷ primary (master) server maintains master DB
  - ▷ secondary (slave) servers have a copy of DB for load-balancing
  - ▷ primary and secondary are synchronized by "zone transfer"
  - ▷ note: both primary and secondary accept queries as working servers ("slave" does not mean backup)
- caching servers perform DB lookup
- (stub) resolvers are end clients
- communicate over {UDP,TCP}/{IPv4,IPv6}, port 53

# DNS data lookup (name resolution)



# Resource Records (RRs)

- Entity of DNS DB (record)
  
- `<name> <tTL> <class> <type> <data>`
  - name: a domain name = lookup key
  - tTL: lifetime of cached RRs in seconds
    - (don't confuse it with TTL field of IP Header)
  - class: typically IN (Internet Domain)
  - type: type of record (A, NS, MX, PTR, SOA...)
  - data: data depending on class/type
    - e.g. an IPv4 address for A RR
    - can consist of multiple fields

# Major resource records (1/2)

- SOA (Start of Authority) RR
  - mandatory RR for every zone
  - provides several configuration parameters
- A (Address) RR
  - provides hostname to IPv4 address mapping
- AAAA (IPv6 Address) RR
  - provides hostname to IPv6 address mapping
- NS (name server) RR
  - specifies the name of nameserver
  - defines zone delegation
    - both parent and child maintain NS RRsets
    - parent's set must be a subset of (or equal to) child's set (cause "lame delegation" otherwise)

# Major resource records (2/2)

- PTR (Pointer) RR
  - provides IP address to hostname mapping
  - for both IPv4 and IPv6
- MX (Mail Exchange) RR
  - provides the name of a mail server for a domain
- CNAME (Canonical Name) RR
  - provides an alias of a hostname

# SOA Parameters

- Very poorly understood
  - many operators simply copy-and-paste it
  - causing troubles in some cases

```
@ 1D IN SOA <primary_server> <admin_mail_address> (  
    2004111905      ; serial number  
    3600           ; refresh interval  
    600            ; retry interval  
    2419200        ; expiration limit  
                   ; (4 weeks in this case)  
    1200           ; "minimum" TTL  
)
```

# Important SOA Parameters (1/2)

- <primary\_server>
  - e.g. "ns.wide.ad.jp."
  - should properly be configured for dynamic update
  
- <admin\_mail\_address>
  - use "." instead of "@"
    - e.g. jinmei@kame.net -> jinmei.kame.net.
  - should be accurate in case of trouble
    - someone may help you with a valid contact point
    - do not simply copy it from other configurations

# Important SOA Parameters (2/2)

- **Serial number**
  - specifies whether zone transfer is necessary
  - a popular cause of DNS trouble
    - do not forget updating it when modifying a zone file
  
- **"Minimum" TTL**
  - previously used for indicating the "default" TTL
    - tend to be large (e.g., 1-2 days)
  - RFC 2308 update: TTL for negative caching
    - recommended value: 10-30 minutes
  - **revisit your configuration**
    - stick to the RFC 2308 definition
    - use "\$TTL" for the default TTL

# Notes on CNAME (1/2)

- Another major source of DNS troubles
  
- Note 1: avoid CNAME chain  
foo.example. CNAME bar.example.  
bar.example. CNAME baz.example.  
    ○ not prohibited, but troublesome
  - ▷ loop-prone
  - ▷ may affect lookup performance
  
- Note 2: do not use CNAME for NS/MX names  
kame.example. NS ns.kame.example.  
ns.kame.example. CNAME cname.kame.example.  
cname.kame.example. A 192.0.2.1

# Notes on CNAME (2/2)

- Note 3: do not specify multiple CNAMEs

dup.example. CNAME foo.example.

dup.example. CNAME bar.example.

- Note 4: do not mix CNAME with other RRs

dup.example. CNAME foo.example.

dup.example. A 192.0.2.2

- Recent BIND does not work with the invalid configurations

- will be ignored or rejected

# Sample of complete zone file

## □ "jinmei.org." zone

\$TTL 1D

```
@      1D IN  SOA   ns.jinmei.org.  root.jinmei.org. (  
                2004111903    ; serial  
                3600          ; refresh  
                600           ; retry  
                2419200       ; expire  
                1200 )        ; minimum
```

```
      IN  NS   ns  
      IN  MX   10  mail
```

```
ns    IN  A    203.178.141.194  
mail  IN  A    203.178.141.195
```

```
www   IN  CNAME ns
```

```
child IN  NS   ns.child  
      IN  NS   ns.kame.net. ; don't forget terminating "period"  
ns.child IN  A    203.178.141.202
```

# IPv6 related RRs

- RFC 3596

- Forward lookup: AAAA RR

www.kame.net. IN AAAA =>

2001:200:0:8002:203:47ff:fea5:3085

- c.f. similar to IPv4

www.kame.net. IN A 203.178.141.194

- Reverse lookup: PTR RR

- TLD: ip6.arpa.

- 1 label for 4 bits, 32 labels, lower to upper

;;for 2001:0200:0000:4819:0280:adff:fe71:81fc

\$ORIGIN 9.1.8.4.0.0.0.0.0.2.0.1.0.0.2.ip6.arpa.

c.f. 1.8.1.7.e.f.f.f.d.a.0.8.2.0 IN PTR www.kame.net.

# IPv6 transport for DNS

- Use IPv6 to send DNS query/response
- Some TLD servers now support IPv6 transport
  - root: planning
    - AAAA glues can be registered
  - gTLD: {A, B}.gtld-servers.net
  - ccTLD: 135 domains, 43 nameservers
    - out of 243 domains, 715 servers
- Implementation
  - servers: almost ready
  - resolvers: being deployed

# Obsoleted standards on IPv6 DNS

- RFC3363 obsoletes:
  - A6 RR
  - DNAME RR for IPv6 reverse lookup
  - bitstring labels
  
- Some old books still mention those, but just forget them
  - they will never revive

# Introduction to DNS operation with BIND

- BIND overview
- BIND tools
- How to use dig

# BIND

- **Berkeley Internet Name Domain**
  - most widely-used DNS implementation
  - developed by ISC (Internet Systems Consortium)
    - ▷ <http://www.isc.org/>
  
- **How to get it**
  - <ftp://ftp.isc.org/isc/bind/>
  - <ftp://ftp.isc.org/isc/bind9/>
  
- **Recommended version: BIND9**
  - safe, robust, protocol conformance
  - supporting latest protocol standards
  - BIND8 may still be okay, but at least drop BIND4

# Useful tools for DNS operation with BIND

- Configuration checkers
  - named-checkzone, named-checkconf
  - perform those before reloading server
- rndc
  - control command for BIND
    - (was "ndc" in BIND8)
  - reload: reloading server with new config
  - dumpdb: dump internal cache DB to a file
  - querylog: enable per-query logging
  - use "rndc-confgen" for setup
- dig/host/nslookup
  - powerful diagnostic tools

# dig

- Domain Information Groper
  - generates a DNS query
  - prints the response "as is"
- Basic command line description
  - dig [`@server_address`] [`RR_type`] `domain_name`
  - `server_address`
    - `/etc/resolv.conf` will be used when unspecified
    - use numeric address to avoid confusion
  - `RR_type`
    - A RR by default
    - specify "AAAA" for looking up IPv6 addresses
- Important output fields
  - "ANSWER SECTION"
  - "status" and "flag" in HEADER

# Output example of dig

```
; <<>> DiG 9.4.0a0 <<>> @203.178.141.194 orange.kame.net
; (1 server found)
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15195
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 1

;; QUESTION SECTION:
;orange.kame.net. IN A

;; ANSWER SECTION:
orange.kame.net. 86400 IN A 203.178.141.194

;; AUTHORITY SECTION:
kame.net. 86400 IN NS orange.kame.net.
kame.net. 86400 IN NS ns1.itojun.org.

;; ADDITIONAL SECTION:
orange.kame.net. 86400 IN AAAA 2001:200:0:8002:203:47ff:fea5:3085

;; Query time: 6 msec
;; SERVER: 203.178.141.194#53(203.178.141.194)
;; WHEN: Wed Nov 17 00:28:13 2004
;; MSG SIZE rcvd: 119
```

# dig tips

- -x: a short-cut for reverse lookup

dig -x 203.178.141.194

=> dig 194.141.178.203.in-addr.arpa. ptr

- be sure to use latest dig (9.3.0 or 9.2.4) for IPv6 reverse lookup

▷ older versions have some transition issues

- Terminating period for confusing names

- "dig in" means dig -c IN .

- "dig in." specifies ccTLD of India

# DNS operational considerations

- Lame delegation
- EDNS0 and packet size issue
- Number/location of servers
- Caching/authoritative separation

# Lame delegation

- A typical DNS misconfiguration
  - parent specifies a nameserver for a child domain
  - the server is not configured to have the authority
    - can easily happen since no automatic mechanism guarantees consistency
    - you'll see this in BIND log:  
info: lame server resolving 'foo.lame.example'  
(in 'lame.example?'): 192.0.2.3#53
  - "lame" servers tend to be left alone
    - other working nameservers can "hide" it
- Why bad
  - make lookup slow
  - hit software bug in some cases
  - increase network traffic and server load
    - can be an "attack" for TLD servers

# Detect lame delegation with dig

## □ 1. Get delegation from parent

- with "+norecurse" to make the intent clear

```
% dig @203.178.141.194 www.lame.jinmei.org +norecurse
```

```
;; AUTHORITY SECTION:
```

```
lame.jinmei.org.      86400 IN    NS     ns.lame.jinmei.org.
```

```
;; ADDITIONAL SECTION:
```

```
ns.lame.jinmei.org.  86400 IN    A     202.249.10.124
```

## □ 2. Send query to the lame server

- check if "aa" flag is set in the response

```
% dig @202.249.10.124 www.lame.jinmei.org +norecurse
```

```
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11840
```

```
;; flags: qr ra;  QUERY: 1, ANSWER: 0, AUTHORITY: 2, ADDITIONAL: 2
```

```
;; QUESTION SECTION:
```

```
;www.lame.jinmei.org.      IN    A
```

```
;; AUTHORITY SECTION:
```

```
org.      40698 IN    NS     TLD1.ULTRADNS.NET.
```

```
org.      40698 IN    NS     TLD2.ULTRADNS.NET.
```

```
...
```

# EDNS0 and packet size issue

- RFC 2671: provide the ability to negotiate UDP receive buffer size
  - the max response size was 512 bytes
  
- Important for IPv6 support at TLDs
  - we could not add AAAA glues without EDNS0
  - see the difference in the additional section between  
dig @202.12.27.33 toshiba.com +bufsize=1024  
dig @202.12.27.33 toshiba.com  
(202.12.27.33 = m.root-servers.net)
  
- Firewall and EDNS0
  - some FWs drop "large" DNS packets
  - can be crucial with EDNS0
  - check your FW behavior/setting

# Number/location of nameservers

- How many are suitable for your zone?
  - robustness vs management cost
  - large number of servers cause large packets
    - hitting packet size problems
  - TLDs honor robustness
    - root, net, com: 13 servers
    - ccTLDs: 5.2 servers in average
  - usually 2-3 are adequate
- Where should they be located?
  - locate multiple servers at different (topological) places
    - avoid a single point of failure
  - operation quality for remote sites is important
    - not to make secondaries "lame"

# Separate caching from authoritative

- Two different functions
  - caching: recursive resolver for local clients
  - authoritative: public database server
  - BIND has historically allowed both in a single server
  
- Trend towards separation
  - separate caching with access control
    - for local clients only
    - minimize security risks (DoS, cache contamination)
  - some newer implementations provide auth-only functionality
    - can be simpler and faster
    - e.g. djbdns, NSD

# BIND configurations for separation

- Caching-only configuration

```
options {  
    // allow query for local clients only  
    allow-query { 192.0.2.0/24;  
                 2001:db8:1234:abcd::/64; };  
};
```

- Authoritative-only configuration

```
options {  
    recursion no; // disable recursion  
};
```

- Still can be served by a single process

- with "views"

# Final remarks

- Advanced topics
- References
- Assignments

# Advanced topics

- Load-balanced operation with "anycasting"
- Dynamic update
- Security extensions
  - transaction signature (TSIG)
  - DNS security extension (DNSSEC)
- Internationalization (IDN)
- Some will be covered in future lectures or the workshop

# References for further study

## □ Protocol standards

- RFC 1034: DNS concepts and facilities
- RFC 1035: DNS specification
- RFC 1912: Operational and Configuration Errors
- RFC 2181: Clarifications to the DNS Specification

## □ Operation guides

### ○ DNS & BIND (O'Reilly)

- ▷ 4th ed. covers BIND9
- ▷ Note: description on IPv6/DNSSEC is old

### ○ DNS & BIND cookbook (O'Reilly)

- ▷ collection of useful configuration samples

### ○ BIND9 ARM (Administrator Reference Manual)

- ▷ [bind-9.x.y/doc/arm/Bv9ARM.html](http://bind-9.x.y/doc/arm/Bv9ARM.html)

# Assignment 1: check on your ccTLD

- Check the followings on your ccTLD servers
  - do they have IPv6 addresses (AAAA RRs)?
  - if yes, are those RRs registered in the root zone?
  - do they support EDNS0?
  - are there any lame servers?
    - ▷ (fix it if any)
  - (optional) check other ccTLD servers and compare the results.
  - Note: be sure to describe how you did that as well as the answers.

# Assignment 2: DNS lookup overhead

## □ Try the following steps

- A. select a host name for testing
- B. set up a caching server without any cache
  - (or go to step D depending on your env.)
- C. ask the server for the selected name. record the round trip time (note that dig prints RTT).
- D. emulate process C with "dig +trace" and compare the two results. if there is a significant difference, how would you explain that?
- E. repeat B-D and compare the results. what is the most significant factor on overall RTT (if any)?
- F. ask again the caching server for the same name before the cache expires and compare all the results. how effective is DNS caching?

# Assignment 3

- Provide feedback on today's lecture
  - comments
  - questions
  - requests on workshop