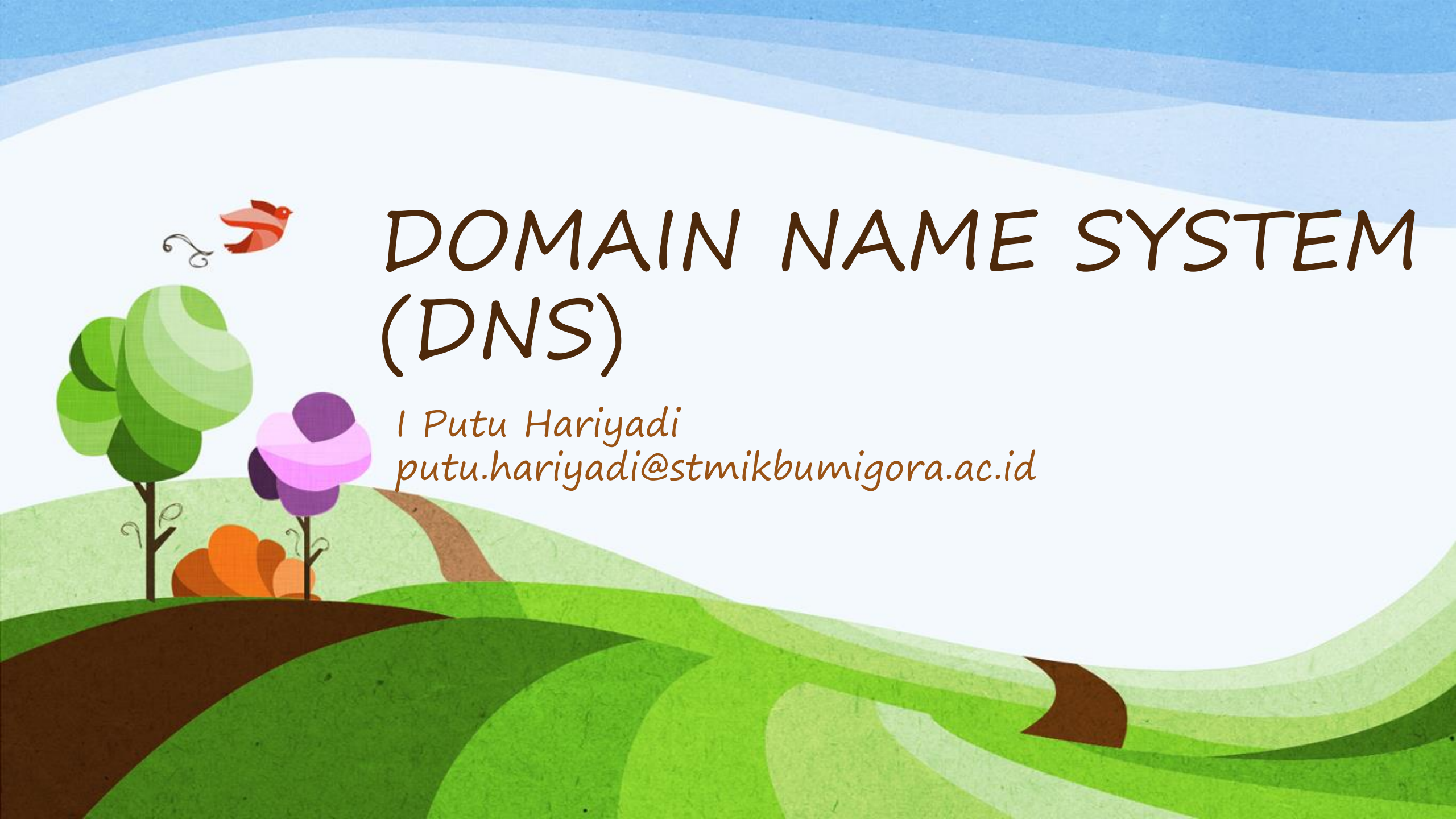


DOMAIN NAME SYSTEM (DNS)

I Putu Hariyadi
putu.hariyadi@stmikbumigora.ac.id

APA ITU DNS?

- Merupakan protocol yang digunakan untuk memetakan atau mentranslasikan nama domain ke alamat IP dan sebaliknya.
- DNS digunakan untuk mentranslasikan atau memetakan nama dari sumber daya ke alamat IP disebut dengan **forward mapping**.
- DNS digunakan untuk mentranslasikan atau memetakan alamat IP ke nama dari sumber daya disebut dengan **reverse mapping**.

SEJARAH DNS (1)

- Tanpa DNS, setiap host yang ingin mengakses sumber daya di jaringan (Internet) seperti halaman web sederhana, sebagai contoh dengan alamat www.thing.com, perlu mengetahui alamat IP-nya.
- Dengan semakin banyaknya host (100 juta host) dan miliaran halaman web, ini adalah tugas yang tidak mungkin – ini juga sangat menakutkan bahkan dengan hanya beberapa host dan sumber daya.
- Pada pertengahan tahun 1970-an dibuat konsep **Name Server** untuk memecahkan permasalahan tersebut sehingga memungkinkan atribut atau properties dari nama sumber daya dipelihara pada lokasi yang dikenal dengan nama "**Name Server**".

SEJARAH DNS (2)

- Dengan kehadiran **Name Server** di jaringan maka host hanya perlu mengetahui alamat IP dari *Name Server* dan nama dari sumber daya yang ingin diakses.
- Sumber daya dapat ditambah, dipindahkan, diubah atau dihapus pada satu lokasi yaitu **Name Server**.
- **Name Server** menjadi sumber daya yang kritis.
- Permasalahan selanjutnya muncul yaitu bagaimana jika Name Server tidak dapat bekerja sehingga host tidak dapat mengakses sumber daya di jaringan? Lebih baik memiliki lebih dari satu Name Server sebagai solusi ketika terjadi kegagalan. Lahirlah konsep **Primary** dan **Secondary Name Server** (bahkan banyak sistem mengijinkan **tertiary** or lebih name server).

SEJARAH DNS (3)

Seiring dengan pertumbuhan jaringan maka memunculkan 3 (tiga) permasalahan baru terkait Name Server yaitu:

1. Pencarian entri di database penamaan menjadi semakin lambat seiring semakin banyaknya penamaan yang tersimpan pada database Name Server sehingga diperlukan cara untuk mengindex atau mengatur penamaan.
2. Jika setiap host mengakses Name Server maka beban menjadi sangat tinggi sehingga diperlukan adanya penyebaran beban ke sejumlah server.
3. Dengan banyaknya nama sumber daya di database maka manajemen menjadi sulit terutama ketika setiap orang ingin melakukan proses pembaharuan ke semua record secara bersamaan sehingga diperlukan cara untuk memisahkan atau mendelegasikan administrasi dari sumber daya penamaan.

KONSEP DAN IMPLEMENTASI DNS

PENGENALAN DNS

DNS diperlukan untuk menyelesaikan permasalahan berupa adanya kebutuhan, seperti berikut:

1. Hirarki penamaan.
2. Penyebaran beban operasional pada Name Server.
3. Pendelegasian administrasi dari Name Server.

DOMAIN DAN DELEGASI (1)

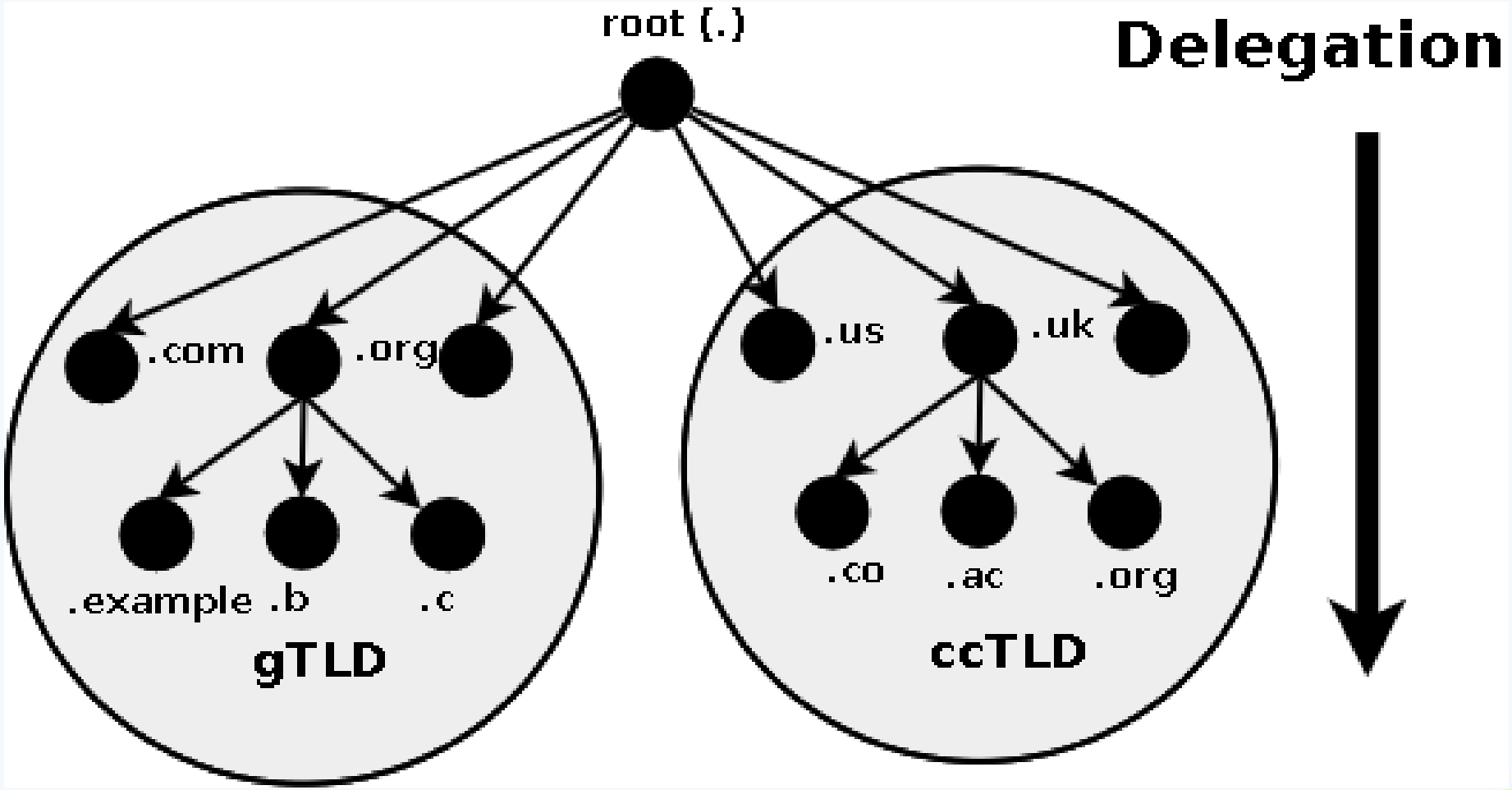
- DNS menggunakan struktur penamaan **tree** (hirarki).
- Bagian atas dari **tree** adalah **root**, diikuti oleh **Top Level Domain (TLD)** kemudian nama domain (**domain name**) dan sejumlah level yang lebih rendah (**lower level**) masing-masing dipisahkan oleh tanda titik (**dot**).
- **Root** dari **tree** dinyatakan menggunakan **silent dot (.)**.
- **Top Level Domain (TLD)** dipecah menjadi 2 (dua) jenis:

1. **Generic Top Level Domain (gTLD)** .com, .edu, .net, .org, .mil dan lain-lain.
2. **Country Code Top Level Domain (ccTLD)** sebagai contoh .us, .ca, .tv, .uk, .id dan lain-lain

DOMAIN DAN DELEGASI (2)

- **Country Code Top Level Domain (ccTLD)** menggunakan **standard two letter sequence** yang didefinisikan oleh **ISO 3166.**
- Sejak 2004, gTLD memiliki sub kategori yaitu **Sponsored TLD (sTLD)** dengan registrasi terbatas seperti **.aero**, **.museum**, **.travel** dan **.jobs** dimana normalnya GTLD memiliki persyaratan pendaftaran terbuka.
- Pada akhirnya sejak 2011 kebijakan TLD tidak dibatasi.

DIAGRAM STRUKTUR DOMAIN DAN DELEGASI



DOMAIN DAN DELEGASI (3)

- Domain Name sebenarnya adalah kombinasi dari nama domain dan TLD dan ditulis dari KIRI ke KANAN dengan level terendah dalam hirarki di sebelah kiri dan level tertinggi di sebelah kanan.

  domain-name.tld # example.com

- Dalam kasus gTLD, seperti .com, .net dll, bagian pengguna dari nama yang didelegasikan - nama yang terdaftar pengguna - adalah Second Level Domain (SLD).
- Ini adalah tingkat kedua dalam hirarki.

DOMAIN DAN DELEGASI (4)

- Bagian pengguna oleh karena itu sering hanya disebut sebagai SLD. Jadi Nama Domain pada contoh di atas bisa didefinisikan ulang terdiri dari:

  sld.tld # example.com

- Istilah Second Level Domain (SLD) jauh lebih sedikit berguna dengan ccTLD dimana bagian yang terdaftar pengguna biasanya adalah Third Level Domain, misalnya:

  example.co.uk

  example.com.br

DOMAIN DAN DELEGASI (5)

- Istilah Second Level Domain (SLD) menyediakan memberikan ketepatan teknis namun dapat membingungkan bila diterapkan pada konsep generik seperti domain pengguna – kecuali ketepatan yang diperlukan.
- Untuk itu Nama Domain atau hanya Domain akan digunakan untuk menggambarkan keseluruhan nama.
- Misalnya, pada panduan ini pemanggilan Nama Domain merujuk ke example.com atau example.co.uk.

OTORITAS DAN DELEGASI (1)

- Konsep Delegasi dan Otoritas terletak pada inti hirarki sistem nama domain.
- Otorita untuk domain root terletak pada Internet Corporation for Assigned Numbers and Names (ICANN).
- Sejak tahun 1998 ICANN, sebuah organisasi nirlaba, telah mengambil tanggung jawab ini dari pemerintah AS.
- GTLD secara otoritatif dikelola oleh ICANN dan diserahkan ke serangkaian pendaftar terakreditasi.
- ccTLDs didelegasikan ke masing-masing negara untuk keperluan administrasi.

OTORITAS DAN DELEGASI (2)

- Diagram struktur dan delegasi sebelumnya menunjukkan bagaimana otoritas apapun pada gilirannya mendelegasikan ke tingkat yang lebih rendah dalam hierarki, dengan kata lain, hal itu mungkin mendelegasikan sesuatu yang bersifat otoritatif.
- Setiap lapisan dalam hirarki dapat mendelegasikan kontrol otoritatif ke tingkat yang lebih rendah berikutnya.
- Dalam kasus ccTLD, negara-negara seperti Kanada (ccTLD .ca) dan AS (ccTLD .us) dan yang lainnya dengan pemerintah federal memutuskan bahwa mereka akan mengelola di tingkat nasional dan mendelegasikan ke setiap provinsi (Kanada) atau negara (AS) dua karakter provinsi / kode negara, misalnya, qc = Quebec, .ny = New York, md = Maryland dll.

OTORITAS DAN DELEGASI (3)

- Dengan demikian mycompany.md.us akan menjadi Nama Domain mycompany yang didelegasikan dari negara bagian MaryLand di AS.
- Ini adalah model delegasi sampai sekitar tahun 2006 ketika kedua negara mengubah kebijakan pendaftaran mereka dan mengadopsi model delegasi yang pada dasarnya datar.
- Jadi, hari ini Anda bisa mendaftarkan mycompany.us atau mycompany.ca (asalkan tersedia).
- Model delegasi lama masih berlaku dan Anda masih memiliki domain seperti quebec.qc.ca serta banyak contoh lain dari model delegasi multi-layer.

OTORITAS DAN DELEGASI (4)

- Negara-negara dengan pemerintah yang lebih terpusat, seperti Inggris, Brasil dan Spanyol dan lainnya, telah memilih segmentasi fungsional dalam model delegasi mereka, misalnya ,co = perusahaan, .ac = akademis dll.
- Jadi mycompany.co.uk adalah Nama Domain dari mycompany, terdaftar sebagai perusahaan dari otoritas registrasi Inggris.
- Delegasi dalam domain apapun mungkin hampir tidak terbatas dan diputuskan oleh otoritas yang didelegasikan.
- Misalnya, AS dan Kanada mendelegasikan kota ke dalam wilayah propinsi/negara bagian sehingga alamat (atau URL) tennisshoes.nb.us adalah kota dari Sepatu Tenis yang terdapat di Negara Bagian Nebraska di Amerika Serikat dan kita bahkan bisa memiliki mycompany.tennisshoes.nb.us.

OTORITAS DAN DELEGASI (5)

- Dengan membaca nama domain dari KANAN ke KIRI Anda dapat melacak delegasinya.
- Unit delegasi ini juga dapat disebut sebagai **zona** dalam dokumentasi standar.

Apa itu www.example.com? (1)

- **www.example.com** dibangun dari **www** dan **example.com**.
- Bagian dari nama domain **example.com** didelegasikan dari **registrar gTLD** yang mana sebelumnya didelegasikan dari ICANN.
- Bagian **www** dipilih oleh pemilik domain karena mereka sekarang orang yang memiliki otoritas untuk nama **example.com**.
- Mereka memiliki SEGALANYA sampai KIRI dari Nama Domain yang didelegasikan.

Apa itu www.example.com? (2)

- Bagian paling kiri, **www** dalam kasus ini, disebut **hostname**.
- Setiap komputer, atau layanan, yang dapat dialamatkan (memiliki URL) melalui Internet atau jaringan internal memiliki bagian nama host, berikut adalah beberapa contoh ilustratif lainnya:

  www.example.com – layanan web perusahaan

  ftp.example.com – server transfer file perusahaan

  pc17.example.com – PC biasa atau host

  accounting.example.com – sebuah sistem akuntansi
- Bagian nama host harus unik dalam Nama Domain, namun bisa jadi apa pun yang pemilik example.com inginkan.

Apa itu www.example.com? (3)

- Contoh lainnya adalah www.us.example.com

  example.com adalah nama domain. **www** bisa jadi mengindikasikan website. **us** disebut dengan **sub domain** yang dialokasikan oleh pemilik example.com.

- Dapat disimpulkan PEMILIK dapat mendelegasikan, DENGAN CARA APAPUN YANG MEREKA INGINKAN, APA PUN ke KIRI dari Nama Domain yang mereka miliki (didelegasikan).

- Pemilik juga BERTANGGUNG JAWAB untuk mengelola delegasi ini yang berarti menjalankan, atau mendelegasikan tugas menjalankan, DNS yang berisi informasi otoritatif (atau record) untuk Nama Domain (atau zona) mereka.

ORGANISASI DAN STRUKTUR DNS (1)

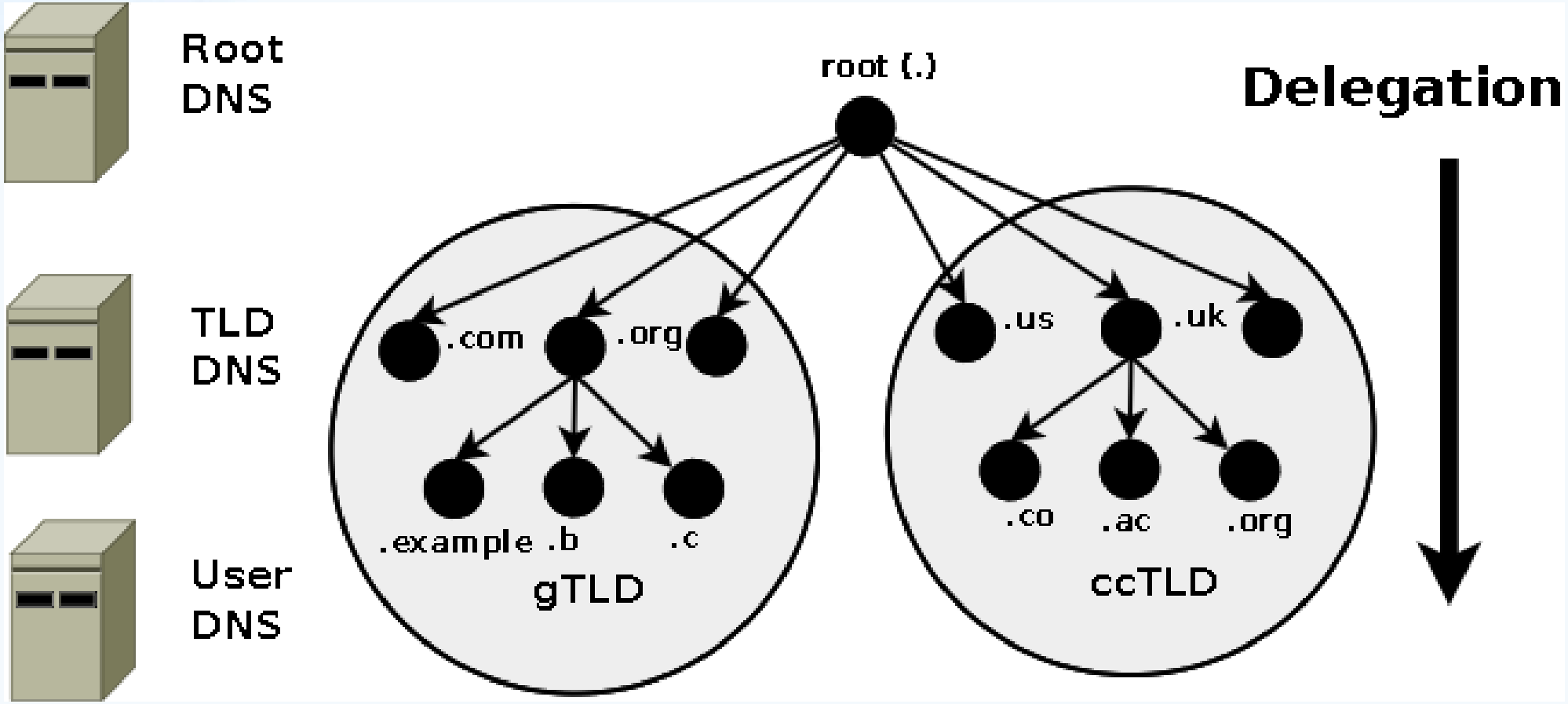


Diagram pemetaan DNS ke delegasi domain

ORGANISASI DAN STRUKTUR DNS (2)

- **Root Server (Root DNS)** adalah tanggung jawab dari **ICANN** namun dioperasikan oleh konsorsium berdasarkan perjanjian delegasi.
- **ICANN** menciptakan **Root Servers Systems Advisory Committee (RSSAC)** untuk memberikan saran dan panduan mengenai pengoperasian dan pengembangan sumber daya kritis ini.
- **IETF** diminta oleh RSSAC untuk mengembangkan standar teknik untuk pengoperasian Root-Servers.
- Permintaan ini menghasilkan publikasi RFC 2870.

ORGANISASI DAN STRUKTUR DNS (3)

- Saat ini (pertengahan 2003) terdapat **13 root-server** di seluruh dunia. Root-Server diketahui oleh setiap server DNS publik di dunia dan merupakan titik awal untuk setiap operasi pencarian nama (atau query).
- Untuk membuat ketahanan tambahan setiap root-server biasanya memiliki banyak instances (salinan) yang tersebar di seluruh dunia.
- Setiap instance memiliki alamat IP yang sama namun data dikirim ke instance terdekat dengan menggunakan proses yang disebut **anycasting**.
- Server TLD (ccTLD dan gTLD) dioperasikan oleh berbagai lembaga dan organisasi (di bawah seperangkat perjanjian yang cukup kompleks) yang disebut **Registry Operator**.

ORGANISASI DAN STRUKTUR DNS (4)

- Otoritas dan tanggungjawab dari DNS Server pengguna (atau nama domain) terletak pada pemilik domain.
- Pada banyak kasus tanggungjawab ini didelegasikan oleh pemiliki domain ke **Internet Service Provider (ISP)**, perusahaan **Web Hosting** atau **Registrar**.
- Banyak perusahaan juga memilih untuk menjalankan Server DNS sendiri dan bahkan mendelegasikan otoritas dan tanggungjawab untuk sub-domain Server DNS untuk memisahkan bagian-bagian dari organisasi.

ORGANISASI DAN STRUKTUR DNS (5)

- Ketika tidak ada DNS yang dapat menjawab (resolve) permintaan (query) untuk nama domain dari client, sebagai contoh example.com maka query ini diteruskan ke root-server.
- Root-server yang akan mengarahkan (refer) permintaan ke TLD DNS Server yang tepat (untuk .com).
- Pada gilirannya TLD DNS Server akan mengarahkannya ke DNS Server Domain pengguna yang sesuai.

KOMPONEN SISTEM DNS (1)

DNS yang didefinisikan pada RFC 1034 meliputi 3 (tiga) bagian yaitu:

1. Data yang menjelaskan domain.
2. Satu atau lebih program Name Server.
3. Program Resolver atau Library.

KOMPONEN SISTEM DNS (2)

- Sebuah server DNS dapat mendukung banyak domain.
- Data dari masing-masing domain menjelaskan sifat umum dari domain dan host atau layanan (services).
- Data ini didefinisikan dalam bentuk tekstual dari Resource Record yang diatur pada file zone.
- Format dari file zone didefinisikan pada RFC 1035 dan didukung oleh perangkat lunak DNS.

KOMPONEN SISTEM DNS (3)

- Program Name Server umumnya melakukan 3 (tiga) hal:

1. Membaca file konfigurasi yang mendefinisikan zone yang menjadi tanggungjawabnya.
2. Bergantung pada fungsi dari Name Server, file konfigurasi dapat menjelaskan beragam kelakuan, sebagai contoh dapat di cache atau tidak.
3. Membalas pertanyaan (query) dari local atau remote host.

- Program Resolver atau Library ditempatkan pada setiap host dan menyediakan cara untuk mentranslasi permintaan pengguna sebagai contoh www.thing.com ke satu atau lebih query ke server DNS menggunakan protocol UDP atau TCP.

ZONE AND ZONE FILES (1)

- Zone merupakan bagian dari ruang nama DNS yang didelegasikan ke server atau administrator lain.
- File zone memuat **Resource Record (RR)** yang menjelaskan domain atau sub domain.
- RR menjelaskan sifat umum (*global properties*) dari zone.
- Contoh RR antara lain SOA Record, NS Records, A Records, CNAME Records, PTR Records, MX Records.
- Format dari file zone adalah standard IETF yang didefiniskan oleh RFC 1035.

CONTOH RESOURCE RECORD (RR) PADA FILE ZONE

```
foo.com.        IN      SOA     ns.joe.com. root.foo.com. (
                                2003080800 ; se = serial number
                                3h         ; ref = refresh
                                15m        ; ret = update retry
                                3w         ; ex = expiry
                                3h         ; min = minimum
                                )
                IN      NS      ns1.foo.com.
                IN      MX  10  mail.anotherdomain.com.
joe             IN      A       192.168.254.3
www             IN      CNAME   joe
```

ZONE AND ZONE FILES (2)

File zone akan terdiri dari jenis data berikut:

1. Data yang mengindikasikan bagian atas dari zone dan properties umum (**Start Of Authoritity (SOA) Record**).
2. Data otoritatif untuk seluruh node atau host di dalam zone biasanya **A (IPv4)** atau **AAAA (IPv6) record**.
3. Data yang menggambarkan informasi umum untuk zone termasuk **Mail Exchange (MX)** dan **Name Server (NS) record**.
4. Pada kasus delegasi sub domain, name server yang bertanggungjawab untuk sub domain tersebut (satu atau lebih **NS record**)

ZONE AND ZONE FILES (3)

5. Pada kasus delegasi sub domain, satu atau lebih **glue record** yang mengijinkan name server untuk menjangkau sub domain, umumnya satu atau lebih **A** atau **AAAA record** untuk *name server* dari sub domain.

DNS QUERIES (1)

- Tugas utama dari DNS Server adalah menjawab query (pertanyaan) dari **local** atau **remote resolver** atau DNS lain yang bertindak sebagai **resolver**.
- Query dapat berupa sesuatu seperti "Apa alamat IP dari **stmikbumigora.ac.id**?".
- DNS Server dapat menerima query terkait domain apa pun.
- DNS Server dapat dikonfigurasi untuk menjadi **authoritative** untuk domain tertentu, **slave** untuk domain lainnya, meneruskan (**forward**) **queries** atau kombinasi lainnya.

DNS QUERIES (2)

- Terdapat 3 (tiga) jenis query yang didefinisikan untuk DNS yaitu:

1. **Recursive Query**
   Jawaban lengkap dari query selalu diberikan. DNS Server tidak perlu mendukung **recursive queries**.

2. **Iterative (Non Recursive) Query**
   Jawaban lengkap BISA jadi diberikan atau rujukan (**referral**) ke DNS lainnya diberikan. Semua DNS Server harus mendukung **iterative queries**.

DNS QUERIES (3)

3. **Inverse Query**

   Digunakan ketika pengguna ingin mengetahui nama domain yang diberikan **resource record**. *Inverse query* tidak didukung dengan baik, sangat jarang dan sudah tidak digunakan lagi menurut RFC 3425.

Proses **Reverse Mapping** (memetakan alamat IP ke nama domain) tidak menggunakan **Inverse Queries** tetapi menggunakan **Recursive** dan **Iterative (Non Recursive) Query** serta menggunakan nama domain khusus yaitu: **IN-ADDR.ARPA**

DNS QUERIES (4)

- Secara historis reverse IPv4 mapping tidak wajib.
- Banyak sistem saat ini menggunakan reverse mapping untuk skema keamanan dan otentikasi sederhana (terutama server email) sehingga penerapan dan pemeliharaan yang tepat sekarang praktis menjadi penting.
- IPv6 awalnya mengamanatkan reverse mapping namun, seperti banyak mandat IPv6 yang asli, kini telah diluncurkan kembali.

RECURSIVE QUERIES (1)

- Merupakan salah satu query dimana server DNS akan sepenuhnya menjawab query (atau memberikan kesalahan).
- Server DNS tidak diharuskan untuk mendukung recursive query dan kedua resolver (atau DNS lain yang bertindak secara rekursif atas nama resolver lainnya) menegosiasikan penggunaan layanan rekursif menggunakan bit (RD) pada query header.

RECURSIVE QUERIES (2)

Terdapat tiga kemungkinan tanggapan terhadap query recursive yaitu:

1. Jawaban atas permintaan disertai dengan CNAME record (alias) yang mungkin berguna. Respon akan menunjukkan apakah data tersebut authoritative atau di-cache.
2. Kesalahan yang menunjukkan domain atau host tidak ada (NXDOMAIN). Tanggapan ini mungkin juga berisi CNAME record yang menunjuk ke host yang tidak ada.

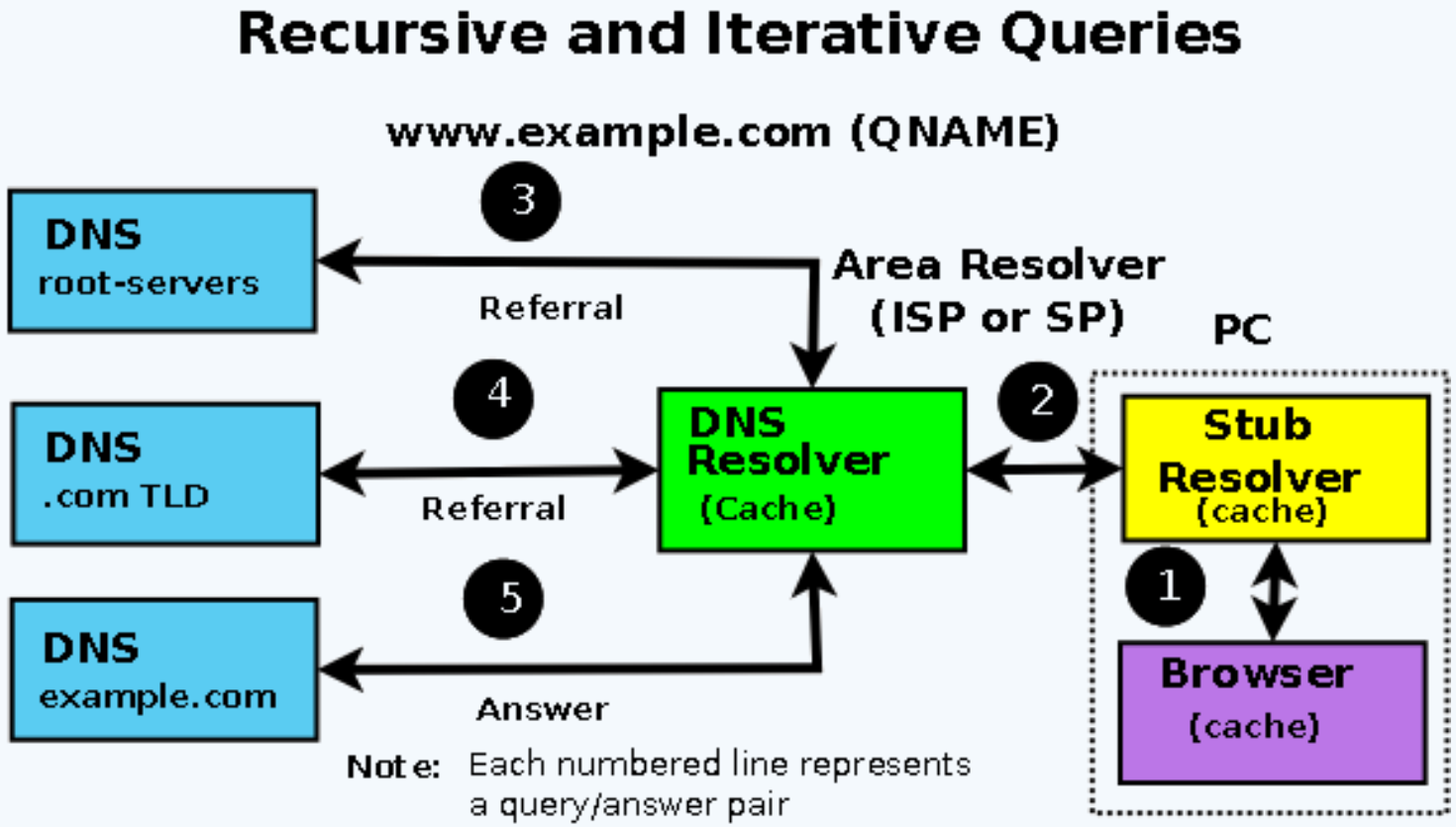
RECURSIVE QUERIES (3)

3. Indikasi kesalahan sementara, misalnya, tidak dapat mengakses DNS lain karena kesalahan jaringan dan lain-lain.

Dalam sebuah recursive query, Resolver DNS akan, atas nama klien (stub-resolver), mengejar jejak sistem DNS di seluruh alam semesta untuk mendapatkan jawaban sebenarnya atas pertanyaan tersebut.

Perjalanan sebuah query sederhana seperti "apa alamat IP dari www.example.com?" ke DNS Resolver yang mendukung recursive query namun tidak otoritatif untuk example.com ditunjukkan pada diagram recursive query processing berikut:

DIAGRAM RECURSIVE QUERY PROCESSING



RECURSIVE QUERIES (4)

1. Pengguna memasukkan **www.example.com** pada address bar dari browser. Browser mengeluarkan standard function library call (1) ke local stub-resolver.

2. Stub-resolver mengirimkan sebuah query (2) "apa alamat IP dari www.example.com?" ke resolver DNS yang dikonfigurasi secara lokal (alias recursive name server). Ini adalah permintaan DNS standar yang meminta layanan rekursif (RD (Recursion Desired) = 1).

3. DNS Resolver mencari alamat www.example.com di tabel lokalnya (cache-nya) dan tidak menemukannya. (Jika ditemukan, segera dikembalikan ke Stub-resolver dalam pesan jawaban dan transaksi akan selesai).

RECURSIVE QUERIES (5)

4. DNS Resolver mengirim query (3) ke root-server untuk IP dari www.example.com. Setiap DNS Resolver dikonfigurasi dengan file yang memberitahu nama dan alamat IP dari root server. Root-server, TLD Server dan name server yang dikonfigurasi oleh pengguna dengan benartidak mendukung recursive query sehingga resolver akan, biasanya, tdak mengatur Recursion Desired (RD=0). Ini adalah iterative query.

5. Root-server tidak mengetahui tentang example.com, apalagi bagian www, tetapi ia mengetahui tentang level berikutnya dari hirarki. Pada kasus ini adalah .com sehingga ia akan membalas dengan rujukan (3) yang mengacu ke TLD server untuk .com.

RECURSIVE QUERIES (6)

6. DNS Server mengirim query baru (4), "Apa alamat IP dari www.example.com?" ke salah satu dari server TLD .com menggunakan iterative query.
7. Serve TLD mengetahui tentang example.com, tetapi tidak mengetahui tentang www. Karena tidak dapat memberikan jawaban lengkap terkait query maka ia membalas (4) dengan rujukan ke name server untuk example.com
8. DNS resolver mengirim query lainnya (5), "Apa alamat IP dari www.example.com?" ke salah satu dari name server untuk example.com. Sekali lagi ini biasanya menggunakan Iterative query.

RECURSIVE QUERIES (7)

9. File zone **example.com** mendefinisikan **A (IPv4 address) record** sehingga *authoritative server* untuk **example.com** mengembalikan (5) **A record** untuk www.example.com. Ini adalah jawaban lengkap dari pertanyaan.

10. DNS Resolver mengirim balasan (jawaban) www.example.com=x.x.x.x ke client stub resolver (2) dan menyimpan informasi ini pada cache.

11. Stub Resolver menyimpan informasi www.example.com=x.x.x.x pada cache-nya dan membalas original standard function call (1) dengan www.example.com=x.x.x.x. Sejak 2003 kebanyakan stub resolver telah menjadi caching stub resolver.

RECURSIVE QUERIES (8)

12. Browser menerima balasan terhadap standard function call, menempatkan informasinya pada cache dan memulai sesi HTTP ke alamat x.x.x.x. Transaksi DNS selesai. Cukup sederhana, tidak banyak yang bisa salah.

Singkatnya, stub resolver menuntut layanan rekursif dari DNS Resolver. DNS Resolver menyediakan layanan rekursif namun biasanya menggunakan iterative query untuk mencapainya.

ITERATIVE (NON-RECURSIVE) QUERIES (1)

- Merupakan salah satu query dimana server DNS dapat memberikan jawaban atau jawaban sebagian (rujukan) pada query (atau memberikan pesan kesalahan).
- Semua server DNS harus mendukung query non-rekursif (Iteratif).
- Iteratif query secara teknis hanyalah queri DNS normal yang tidak meminta layanan rekursif.

ITERATIVE (NON-RECURSIVE) QUERIES (2)

Terdapat empat kemungkinan tanggapan terhadap query non-recursive yaitu:

1. Jawaban atas permintaan disertai dengan CNAME record (alias) yang mungkin berguna. Dalam Iterative Query ini hanya akan terjadi jika data yang diminta sudah tersedia di cache). Respon akan menunjukkan apakah data tersebut authoritative atau di-cache.

2. Kesalahan yang menunjukkan domain atau host tidak ada (NXDOMAIN). Tanggapan ini mungkin juga berisi CNAME record yang menunjuk ke host yang tidak ada.

ITERATIVE (NON-RECURSIVE) QUERIES (3)

3. Indikasi kesalahan sementara, misalnya, tidak dapat mengakses DNS lain karena kesalahan jaringan dan lain-lain.

4. Rujukan (Referral): Jika data yang diminta tidak tersedia di cache maka nama dan addess IP dari satu atau beberapa name server yang lebih dekat dengan nama domain yang diminta (dalam semua kasus ini adalah tingkat terendah berikutnya di hirarki DNS) akan dikembalikan. Rujukan ini mungkin, atau mungkin juga bukan name server authoritave untuk domain target.

ITERATIVE (NON-RECURSIVE) QUERIES (4)

- Pada diagram Recursive Query Processing, transaksi transaksi (3), (4) dan (5) biasanya semua Iterative queries. Bahkan jika server DNS meminta Recursion (RD = 1) itu akan ditolak dan rujukan normal (atau jawaban) dikembalikan.
- Mengapa menggunakan Iterative queries? Mereka jauh lebih cepat. Server DNS yang menerima query dan jika sudah memiliki jawabannya di cache-nya maka akan mengirimkannya. Sebaliknya jika tidak maka akan mengirimkan rujukan.
- Tidak main-main. Iterative query memberikan pemohon kontrol yang lebih besar.
- Rujukan (referral) biasanya berisi daftar name server untuk tingkat berikutnya dalam hierarki DNS.

ITERATIVE (NON-RECURSIVE) QUERIES (5)

- Pemohon mungkin memiliki informasi tambahan tentang satu atau lebih dari name server ini dalam cache-nya (termasuk yang tercepat) dari mana ia dapat membuat keputusan yang lebih baik tentang name server mana yang akan digunakan.
- Iterative query juga sangat berguna dalam situasi diagnostik.

INVERSE QUERIES

- Secara historis, Inverse query memetakan resource record ke domain.
- Contoh Inverse query seperti "apa nama domain untuk record MX ini?".
- Inverse query bersifat opsional dan diizinkan untuk server DNS mengembalikan respons "**Not Implemented**".
- Inverse query tidak digunakan untuk menemukan nama host dari alamat IP yang diberikan.
- Proses ini disebut **Reverse Mapping (Look-up)** menggunakan query rekursif dan Iteratif (non-rekursif) dengan nama domain khusus **IN-ADDR.ARPA**.

ZONE UPDATES (1)

- Desain awal dari DNS memungkinkan perubahan disebarkan menggunakan Zone Transfer (AXFR) namun dunia Internet lebih sederhana dan lebih tenang pada masa itu (1987).
- Keinginan untuk mempercepat proses penyebaran zone update sambil meminimalkan sumber daya yang digunakan telah menghasilkan sejumlah perubahan pada aspek desain DNS dan implementasi dari sederhana namun efektif seperti **Incremental Zone Transfer (IXFR)** dan pesan **NOTIFY** hingga konsep **Dynamic Updates** yang memiliki konsekuensi keamanan yang signifikan jika tidak diterapkan dengan baik.

ZONE UPDATES (2)

- Meskipun transfer zona pada umumnya penting untuk pengoperasian sistem DNS, mereka juga merupakan sumber ancaman. Slave Name Server dapat diracuni jika menerima update zona dari sumber yang jahat.
- Paling minimal selama konfigurasi perlu dipastikan slave hanya akan menerima transfer dari sumber yang diketahui.

FULL ZONE UPDATE (AXFR)

- Spesifikasi asli dari DNS pada RFC 1034 dan 1035 membayangkan bahwa Slave atau Secondary name server akan melakukan "poll" domain atau zone master.
- Waktu untuk melakukan "polling" ditentukan oleh nilai **refresh** dari **SOA Resource Record** pada domain.
- Proses **polling** dilakukan oleh **Slave** dengan mengirim query ke **Master** untuk meminta **SOA Resource Record (RR)** terkini.

FULL ZONE UPDATE (AXFR)

- Jika **serial number** dari **RR** lebih tinggi dari yang digunakan oleh **Slave** saat ini maka **Authoritative Transfer (AXFR)** akan diminta.
- Diperlukan perubahan serial number pada SOA secara disiplin setiap kali dilakukan perubahan pada zone record apapun.
- Zone transfer menggunakan TCP port 53 dimana operasi query DNS normalnya menggunakan UDP pada port 53.

INCREMENTAL ZONE UPDATE (IXFR)

- Mentransfer file zone yang sangat besar dapat memakan waktu lama dan menghabiskan bandwidth dan sumber daya lainnya.
- Ini sangat boros jika hanya sebuah RR tunggal yang telah diubah.
- RFC 1995 memperkenalkan Incremental Zone Transfers (IXFR) yang sesuai namanya memungkinkan Slave dan Master mentransfer hanya record yang telah berubah.
- Prosesnya bekerja seperti AXFR. Slave mengirimkan sebuah query untuk domain SOA RR setiap refresh interval. Jika serial number dari record SOA lebih tinggi daripada yang saat ini dipelihara oleh Slave maka ia akan meminta zone transfer dan mengindikasikan apakah ia dapat menerima Incremental Transfer (IXFR) atau tidak. Jika kedua Master dan slave mendukung fitur tersebut maka terjadi Incremental Transfer (IXFR), sebaliknya terjadi Full Transfer (AXFR). Incremental Zone Transfer Zona menggunakan TCP pada port 53, sedangkan operasi query DNS normal menggunakan UDP pada port 53.

INCREMENTAL ZONE UPDATE (IXFR)

- Mode default untuk BIND saat bertindak sebagai Slave adalah meminta IXFR kecuali jika dikonfigurasi untuk tidak menggunakan parameter **request-ixfr** di server atau *options clause* dari file *named.conf*.
- Mode default untuk BIND saat bertindak sebagai Master adalah menggunakan IXFR hanya bila zonanya dinamis. Penggunaan IXFR dikendalikan menggunakan parameter **provide-ixfr** di server atau *options clause* dari file *named.conf*.

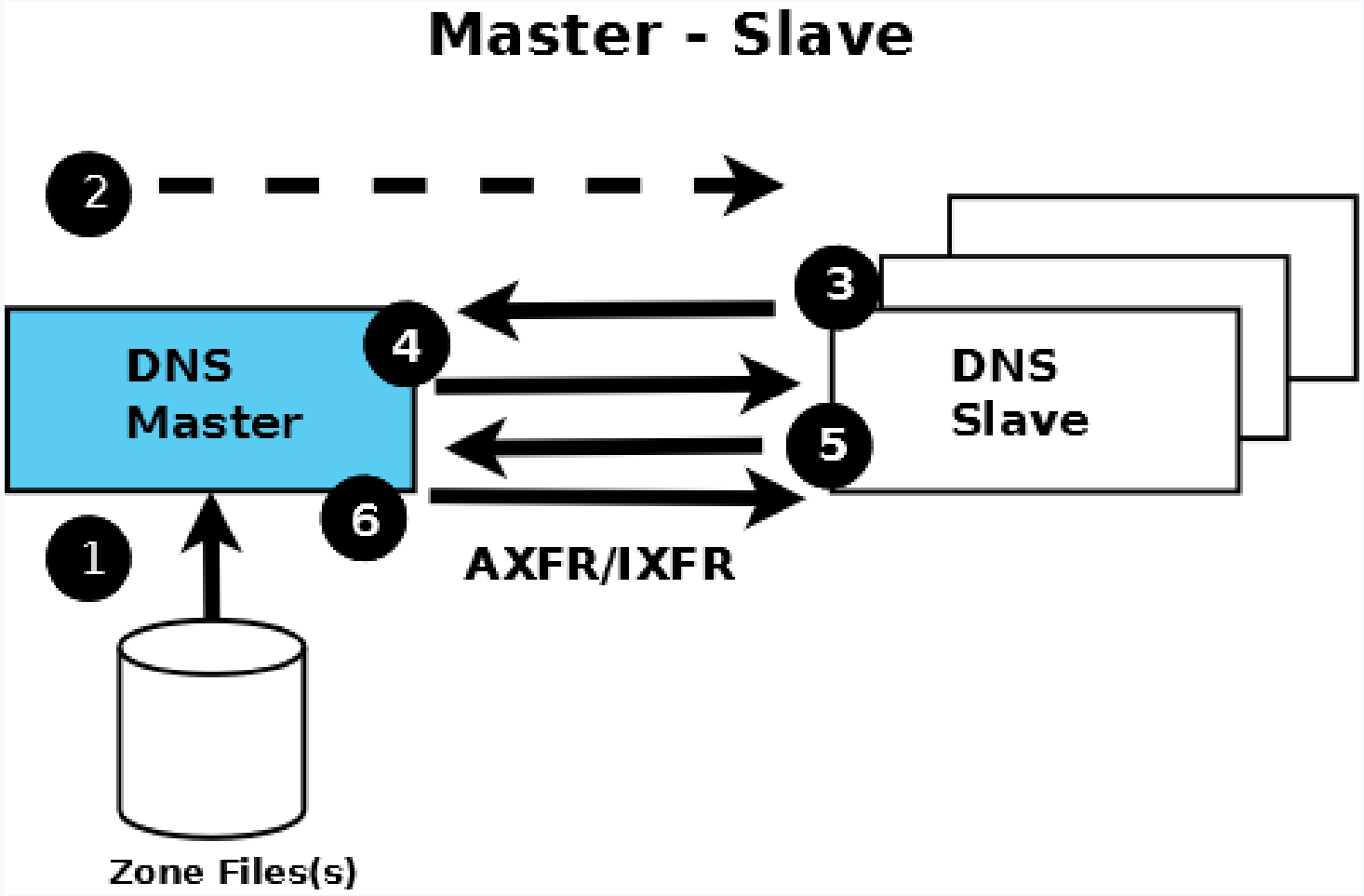
NOTIFY

- RFC 1912 merekomendasikan interval REFRESH sampai 12 jam pada REFRESH interval dari SOA Resource Record.
- Hal ini berarti pada kasus terburuk, perubahan Master Name Server dapat tidak terlihat pada Slave Name Server selama 12 jam.
- Pada lingkungan yang dinamis ini bisa jadi tidak dapat diterima.
- RFC 1996 memperkenalkan skema dimana Master akan mengirim pesan NOTIFY ke Slave Name Server bahwa perubahan dapat terjadi pada record domain.

NOTIFY

- Saat Slave menerima NOTIFY akan meminta SOA Record Resource terbaru dan jika serial number dari SOA RR lebih besar dari nilai saat ini maka ia akan melakukan zone transfer menggunakan Full Zone Transfer (AXFR) atau Incremental Zone Transfer (IXFR).
- Perilaku NOTIFY di BIND dikendalikan oleh parameter notify, also-notify dan notify-source di zone atau options statement dari file named.conf.

DIAGRAM INTERAKSI MASTER-SLAVE DAN ZONE TRANSFER



DESKRIPSI PROSES ZONE TRANSFER

- Waktu yang dibutuhkan untuk menyebarkan perubahan zone ke seluruh Internet ditentukan oleh dua faktor yaitu:

a) Waktu yang dibutuhkan untuk memperbaharui semua Domain Name Server ketika terjadi perubahan zone apapun. Ini ditentukan oleh metode yang digunakan untuk menginisialisasi zone transfer ke seluruh Slave Name Server dimana dapat berupa passive (Slave secara berkala melakukan poll ke Master) atau active (Master akan mengirim NOTIFY ke slave yang dikonfigurasi).

b) Nilai TTL saat ini (sebelum perubahan) ketika zone record apapun diubah akan menentukan kapan Resolvers akan menyegarkan (refresh) cache dengan menginterograsi Autoritative Name Server.

DESKRIPSI PROSES ZONE TRANSFER

1. Jika master diatur untuk mendukung pesan NOTIFY maka ketika status dari file zone Master (1) berubah, ia akan mengirim pesan NOTIFY (2) ke masing-masing Slave yang dikonfigurasi.

   Pesan NOTIFY tidak harus menunjukkan bahwa file zone telah berubah, sebagai contoh jika Master atau zone dimuat ulang maka pesan NOTIFY dipicu meskipun tidak ada perubahan yang terjadi. Saat Slave menerima pesan NOTIFY, maka Slave akan mengikuti prosedur yang didefinisikan pada langkah 3.

DESKRIPSI PROSES TRANSFER ZONE

2. Terlepas dari apakah Master telah dikonfigurasi untuk mendukung pesan NOTIFY atau tidak, maka Slave akan selalu menggunakan proses pasif atau "polling" yang dijelaskan pada langkah ini. (Sementara ini tampak berlebihan dalam kasus di mana Master telah dikonfigurasi untuk menggunakan NOTIFY, meskipun memberi perlindungan terhadap pesan NOTIFY yang hilang karena kesalahan konfigurasi atau serangan berbahaya).

   Ketika server Slave terisi maka ia akan membaca file zone yang tersimpan saat ini (melihat statement file) atau segera memulai zone transfer jika tidak ada file zone yang tersimpan. Kemudian memulai timer dengan menggunakan nilai refresh pada zona SOA RR. Saat timer ini berakhir maka Slave mengikuti prosedur yang didefinisikan pada langkah 3.

DESKRIPSI PROSES TRANSFER ZONE

3. Jika refresh timer dari Slave telah berakhir atau ia menerima pesan NOTIFY maka Slave akan segera mengeluarkan query untuk zone Master SOA RR (3).

4. Ketika jawaban tiba (4) maka Slave membandingkan **serial number** dari **SOA RR** miliknya dengan jawaban dari Master **SOA RR**. Jika nilai **serial number** dari Master SOA RR lebih besar dari serial number di salinan Slave SOA maka zone transfer (5) dimulai oleh Slave. Jika Slave gagal membaca Master SOA RR (atau gagal untuk memulai zone transfer) maka ia akan mencoba lagi setelah *retry time* yang didefinisikan dalam zone SOA RR namun akan terus menjawab secara otoritatif untuk Domain (atau zone). Prosedur *retry* akan diulang (setiap interval *retry*) sampai berhasil (dalam hal ini proses berlanjut pada langkah 5) atau sampai *expiry timer* dari zone SOA RR tercapai, titik di mana Slave akan berhenti menjawab pertanyaan untuk Domain.

DESKRIPSI PROSES TRANSFER ZONE

5. Slave selalu menginisialisasi (5) operasi zone transfer menggunakan AXFR atau IXFR melalui TCP Port 53. Ini dapat diatur menggunakan *statement* **transfer-source**.
6. Master akan mentransfer file zone yang diminta (6) ke slave. Setelah selesai, Slave akan me-*reset timer refresh* dan *expiry*.

DYNAMIC UPDATE

- Metode klasik untuk memperbaharui zone Resource Record adalah dengan mengubah secara manual file zone dan menghentikan serta menjalankan *name server* untuk menyebarkan perubahan.
- Ketika volume perubahan mencapai level tertentu maka operasi ini tidak dapat diterima khususnya pada organisasi yang menangani file zone dengan jumlah besar, seperti *Service Provider*. BIND dapat memakan waktu yang lama untuk me-restart ketika memiliki zone statement yang sangat besar.

DYNAMIC UPDATE

- Hal yang dicari adalah DNS dapat menyediakan metode untuk secara dinamis mengubah record DNS sementara tetap melayani permintaan.
- Terdapat 2 (dua) pendekatan untuk memecahkan permasalahan ini yaitu:

1. Mengijinkan pembaharuan pada zone record ketika berjalan (*run-time*) dari sumber eksternal atau aplikasi.
2. Langsung memberikan DNS dari database yang dapat diperbaharui secara dinamis.

DYNAMIC UPDATE

- RFC 2136 mengambil pendekatan pertama dan mendefinisikan proses dimana zone record dapat diperbaharui dari sumber eksternal.
- Limitasi utama dari spesifikasi ini adalah domain baru tidak dapat ditambahkan secara dinamis. Semua record lainnya dengan zone yang telah ada dapat ditambahkan, diubah, atau dihapus.

PENDEKATAN ALTERNATIF DYNAMIC DNS

- Limitasi dari pendekatan **standard Dynamic DNS (RFC 2136)** adalah domain baru tidak dapat dibuat secara dinamis.
- BIND-DLZ mengambil pendekatan yang jauh lebih radikal dengan menggunakan patch BIND yang serius sehingga memungkinkan penggantian semua file zone dengan satu file zone yang mendefinisikan entri database. Dukungan database mencakup sebagian besar database utama seperti MySQL, PostgreSQL, BDB, dan LDAP sehingga memungkinkan penambahan domain baru serta perubahan domain yang sudah ada sebelumnya tanpa perlu menghentikan dan menjalankan BIND.

PENDEKATAN ALTERNATIF DYNAMIC DNS

- *Authoritative Only Name Server* menggunakan PowerDNS mengambil pendekatan yang sama dengan basis kode sendiri (non-BIND) dengan merujuk semua query ke database sehingga mengijinkan semua domain baru ditambahkan secara dinamis.

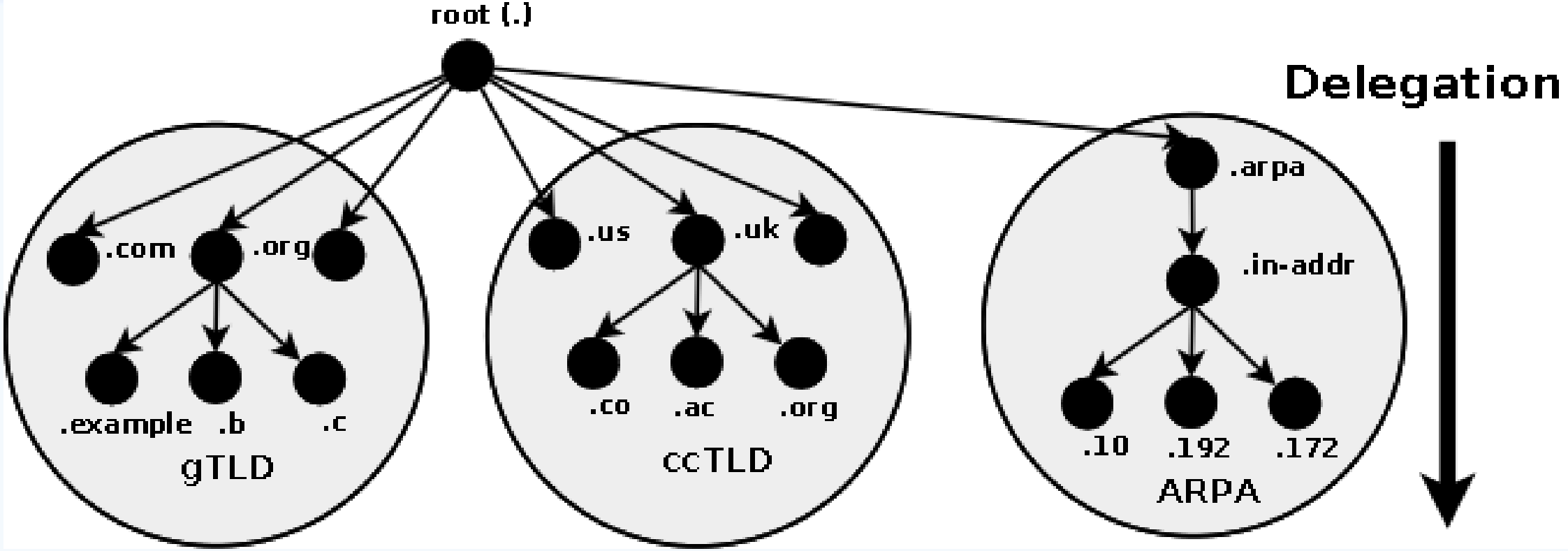
DNS REVERSE MAPPING (1)

- Permintaan DNS normal dapat berupa "apa alamat IP dari host=www di domain = mydomain.com?". Ada kalanya ketika kita ingin bisa mengetahui nama host yang alamat IP = x.x.x.x (atau x: x: x: x: x untuk IPv6).
- Terkadang hal ini diperlukan untuk tujuan diagnostik, saat ini lebih sering digunakan untuk tujuan keamanan yaitu melacak hacker atau spammer.
- Kebanyakan sistem email modern menggunakan pemetaan terbalik untuk memberikan otentikasi sederhana, sensor awal, menggunakan proses pencarian ganda - IP ke nama dan nama ke IP.

DNS REVERSE MAPPING (2)

- IPv4 reverse mapping tidak wajib, seperti yang ditunjukkan oleh contoh mail, sangat penting bagi host yang mengirim email, menggunakan **Mail Transfer Agent (MTA)** atau **Mail User Agent (MUA)**.
- IPv6 reverse-mapping pada awalnya bersifat wajib, namun saat ini sudah tidak lagi menjadi persyaratan wajib.
- Untuk melakukan Reverse Mapping menggunakan query rekursif dan Iteratif (non-rekursif) yang normal, perancang DNS mendefinisikan Nama Domain khusus (dicadangkan) **IN-ADDR.ARPA** untuk alamat IPv4 dan **IP6.ARPA** untuk alamat IPv6.

DIAGRAM IN-ADDR.ARPA REVERSE MAPPING



DNS REVERSE MAPPING (3)

- Normalnya nama domain ditulis dari KIRI ke KANAN. Namun struktur hirarki ditulis dari KANAN ke KIRI.
- Alamat IPv4 ditulis sebagai:

  192.168.23.17
- Alamat IPv4 ini mendefinisikan host (17) yang kebetulan berada dalam kisaran alamat Kelas C (192.168.23.x).
- Dalam kasus ini bagian yang paling penting (simpul tertinggi dalam hirarki alamat) ada di KIRI (192) bukan KANAN.
- Solusinya adalah membalik urutan alamat dan tempatkan hasilnya di bawah domain khusus IN-ADDR.ARPA.
- Bagian terakhir dari Alamat IPv4 (17) adalah alamat host.

DNS REVERSE MAPPING (4)

Hasilnya adalah:

- Alamat IP = 192.168.23.17
- Basis Kelas C = 192.168.23; menghilangkan alamat host = 17
- Basis Kelas C terbalik = 23.168.192
- Ditambahkan ke IN-ADDR.ARPA domain = 23.168.192.IN-ADDR.ARPA

CONTOH REVERSE MAPPED ZONE MENGGUNAKAN PTR RECORD

```
$TTL 2d  ; 172800 seconds
$ORIGIN 23.168.192.IN-ADDR.ARPA.
@               IN      SOA   ns1.example.com. hostmaster.example.com. (
                                  2003080800 ; serial number
                                  3h         ; refresh
                                  15m        ; update retry
                                  3w         ; expiry
                                  3h         ; nx = nxdomain ttl
                                  )
                IN      NS      ns1.example.com.
                IN      NS      ns2.example.com.
1               IN      PTR     www.example.com. ; qualified name
2               IN      PTR     joe.example.com.
.....
17              IN      PTR     bill.example.com.
.....
74              IN      PTR     fred.example.com.
....
```

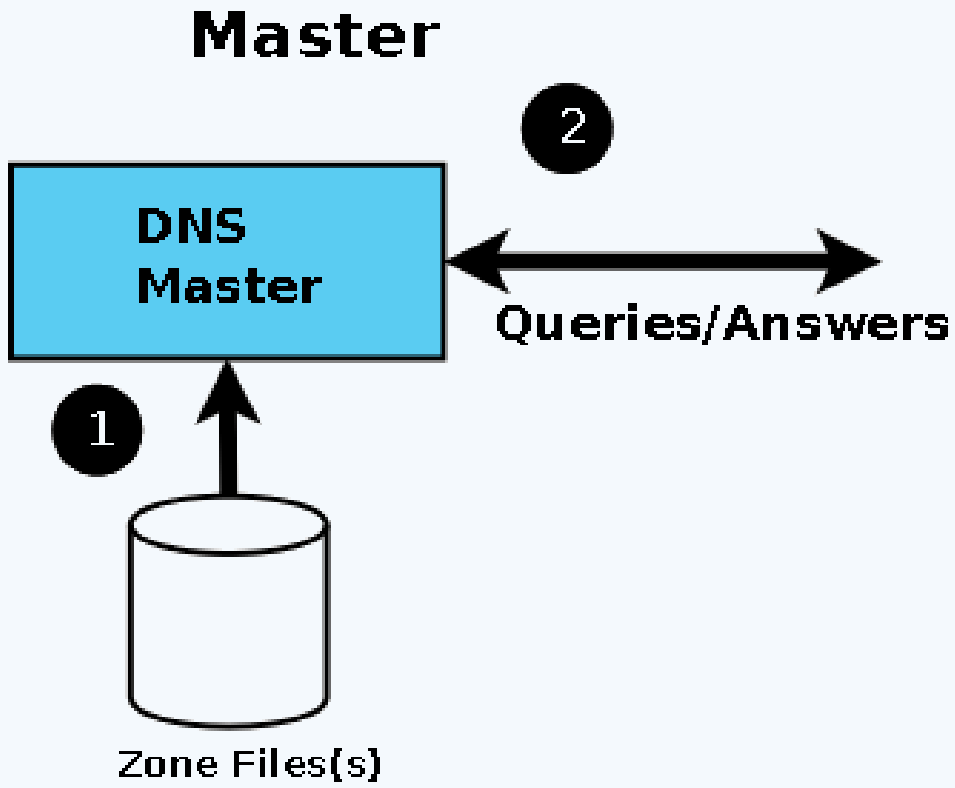
Jenis Konfigurasi DNS

1. Master (Primary) DNS Server
2. Slave (Secondary) DNS Server
3. But Slaves can also be Masters
4. Caching (Hint) DNS Server
5. Forwarding (Proxy, Client, Remote) DNS Server
6. Stealth (DMZ atau Hidden Master) DNS Server
7. Authoritative Only DNS Server
8. Split Horizon DNS Server

MASTER (PRIMARY) NAME SERVER

- Mendefinisikan satu atau lebih file zone dimana DNS ini adalah Authoritative ("type master").
- Zone telah didelegasikan melalui NS (Name Server) Resource Record ke DNS ini.
- Istilah "master" diperkenalkan pada BIND 8.x yang menggantikan istilah "primary".

DIAGRAM DNS MASTER



MASTER (PRIMARY) NAME SERVER

- Ketika DNS Master menerima query untuk zone yang authoritative maka akan direspon sebagai "Authoritative". Bit AA akan diatur sebagai respon dari query.
- Jika server DNS menerima query untuk zone dimana bukan master atau slave maka akan bertindak sesuai dengan konfigurasi yaitu:

1. Jika perilaku caching diijinkan dan recursive query diijinkan maka server akan menjawab secara lengkap permintaan atau mengembalikan pesan kesalahan (error).

MASTER (PRIMARY) NAME SERVER

2. Jika perilaku caching diijinkan dan iterative (non-recursive) query diijinkan maka server akan merespon dengan jawaban lengkap (jika telah terdapat pada cache karena permintaan lainnya), rujukan atau pesan kesalahan (*error*).
3. Jika perilaku caching tidak diijinkan (Authoritative Only DNS Server) maka server akan mengembalikan rujukan (*referral*) atau pesan kesalahan (*error*).

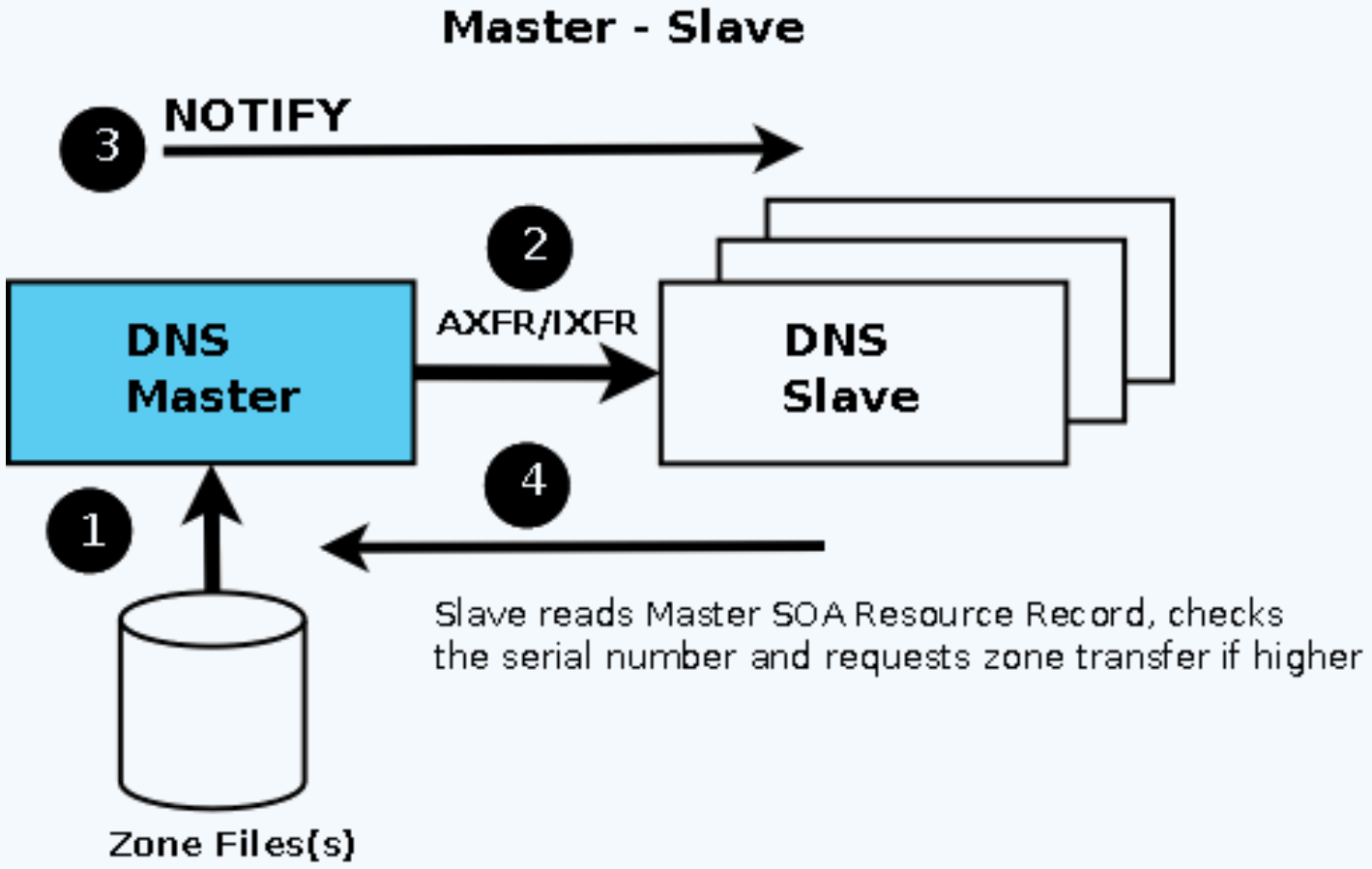
MASTER (PRIMARY) NAME SERVER

- Master DNS Server dapat melakukan notifikasi (NOTIFY) terkait perubahan zone untuk mendefinisikan (biasanya Slave) server dimana ini merupakan perilaku *default*.
- Pesan NOTIFY memastikan perubahan zone secara cepat disebarkan ke slave (interrupt driven) daripada mempercayakan pada slave server yang secara berkala melakukan *polling* terhadap perubahan.
- Zone master dapat disembunyikan (hanya satu atau lebih dari satu slave yang mengetahui keberadaannya). Tidak terdapat persyaratan konfigurasi agar server master muncul pada NS RR dari domain. Persyaratan yang dibutuhkan hanya dua atau lebih name server mendukung zone. Kedua server dapat berupa kombinasi **master-slave**, **slave-slave** atau bahkan **master-master**.

SLAVE NAME SERVERS

- Memperoleh zone data menggunakan operasi zone transfer (biasanya dari zone master) dan akan meresponnya sebagai authoritative untuk zone tersebut dimana didefinisikan sebagai slave dengan konfigurasi zone yang valid.
- Tidak memungkinkan untuk menentukan dari hasil query apakah berasal dari zone master atau slave.
- Istilah **slave** diperkenalkan pada BIND 8.X dan menggantikan istilah **secondary**.

DIAGRAM DNS SLAVE SERVER



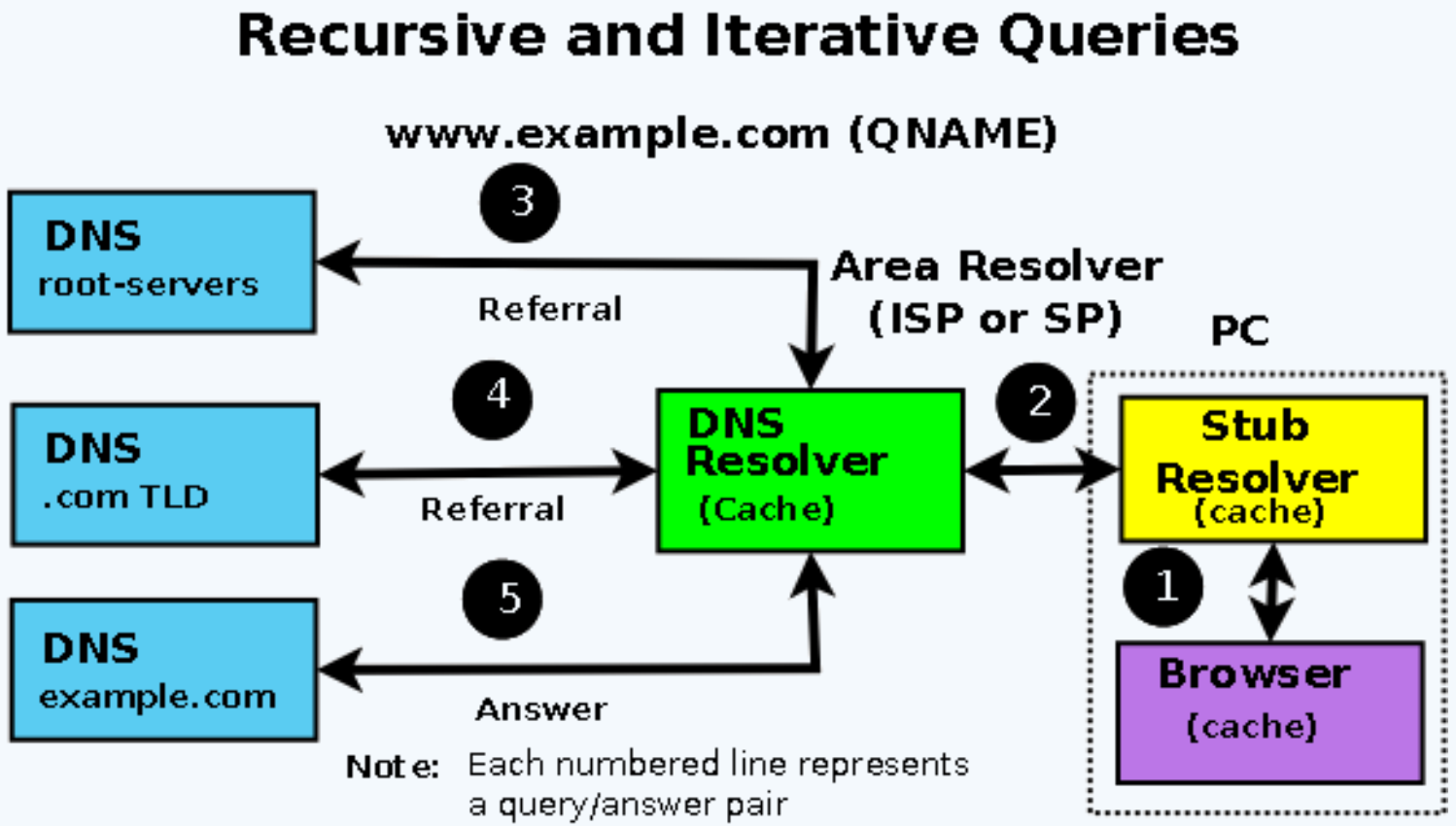
SLAVES CAN ALSO BE MASTERS

- Slave server dapat juga sebagai master.
- Slave server dapat didefinisikan secara sederhana yaitu memperoleh zone data melalui zone transfer, dimana master memperoleh zone data dari file sistem local. Sumber dari zone transfer adalah slave lainnya sebagai master.

CACHING NAME SERVERS (RESOLVER)

- Memperoleh informasi dari server lainnya (Zone Master) sebagai respon dari host query dan menyimpan (cache) datanya secara local.
- Pada permintaan kedua atau lanjutan untuk data serupa maka Caching Server (Resolver) akan merespon dengan data yang tersimpan di local (cache) sampai nilai **Time-To-Live (TTL)** dari jawaban berakhir masa berlakunya *(expires)*. Saat itu Server akan memperbaharui data dari *zone master*.
- Ketika *caching server (resolver)* memperoleh data secara langsung dari *zone master* maka respon adalah "**authoritative**".
- Sebaliknya apabila data diperoleh dari cache maka respon adalah "**non-authoritative**".

DNS RESOLVER (RECURSIVE SERVER)



CACHING NAME SERVERS (RESOLVER)

- Konfigurasi yang umum dari DNS server caching adalah:

1. DNS server bertindak sebagai master atau slave untuk satu atau lebih *zone* (*domains*) dan bertindak sebagai *cache* untuk semua permintaan lainnya. Fungsi umum dari server DNS.

2. *A caching only local server.* Umumnya digunakan untuk meminimalkan akses eksternal atau untuk mengkompensasi jalur eksternal yang lambat. Dikenal dengan nama **proxy server** atau **forwarding server**.

CACHING NAME SERVERS (RESOLVER)

Respon dari *query* adalah **Authoritative** jika memenuhi tiga kondisi berikut:

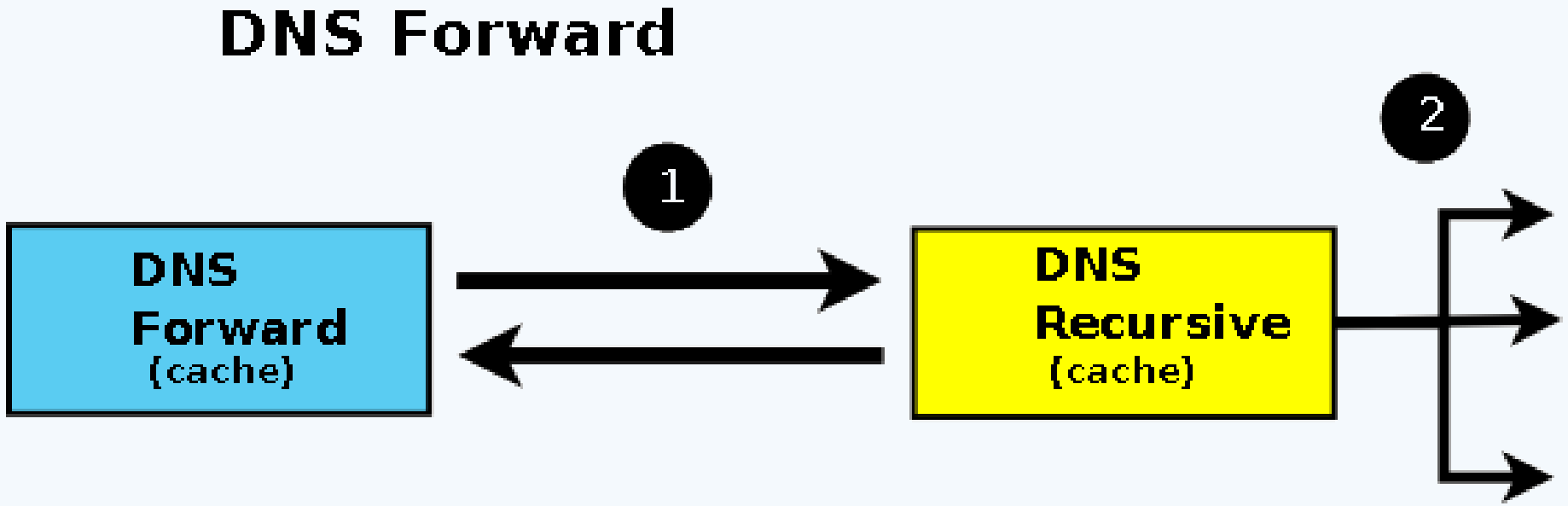
1. Respon diperoleh dari *zone master*.
2. Respon diperoleh dari zone slave dengan data zone yang tidak kadaluarsa.
3. Respon diperoleh dari *caching name server* secara langsung baik dari *zone master* maupun *slave*. Apabila respon diperoleh dari cache maka itu **non-authoritative**.

FORWARDING NAME SERVERS

- Forwarding disebut juga Proxy, Client, Remote server merupakan server yang meneruskan permintaan ke DNS lainnya dan melakukan cache hasilnya.
- Digunakan ketika situasi seperti:

1. Akses ke jaringan eksternal lambat atau mahal.
2. Meringankan beban administrasi lokal dengan memberikan satu titik dimana perubahan pada name server remote dapat dikelola daripada harus memperbaharui semua host.
3. Sanitizing traffic.
4. Sebagai bagian konfigurasi Split Server untuk perimeter defence.

DNS FORWARDING SERVER

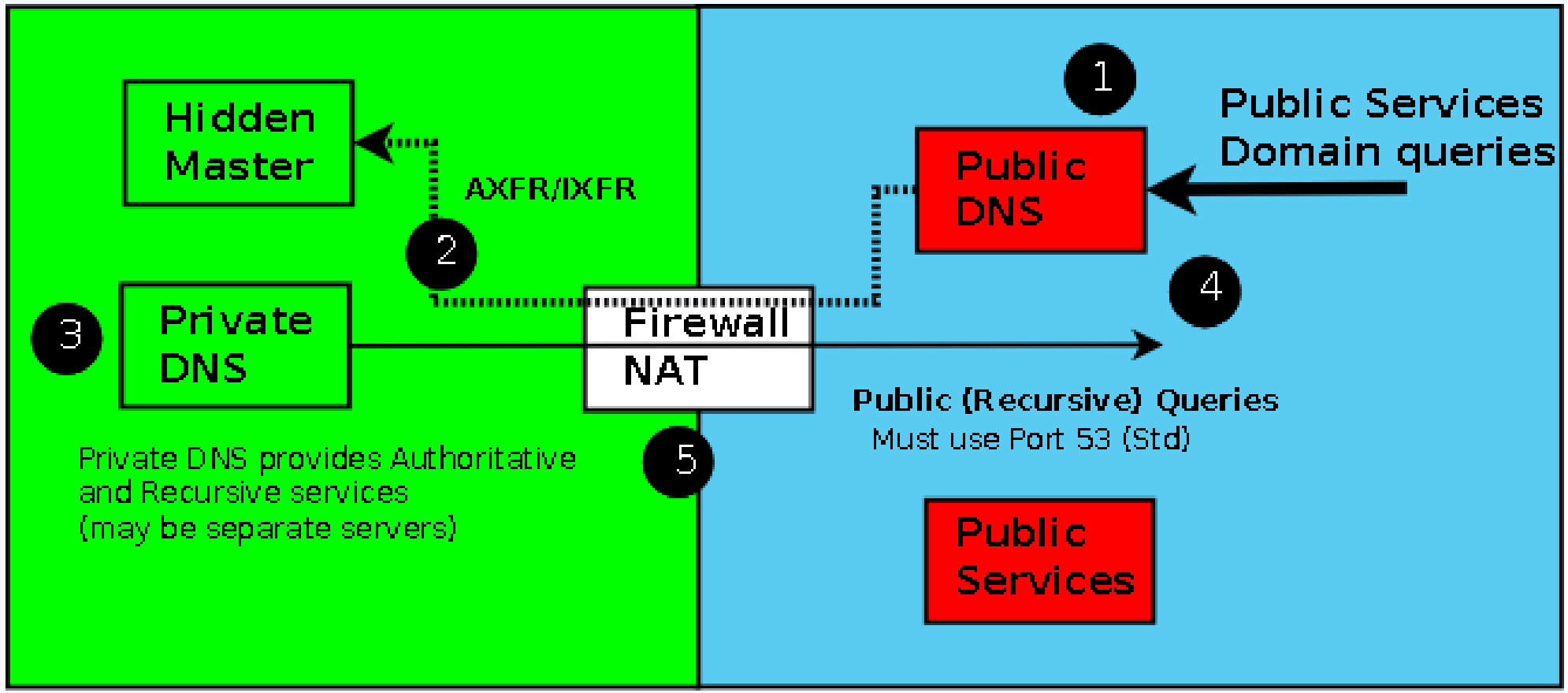


STEALTH (DMZ OR HIDDEN MASTER) DNS

- Didefinisikan sebagai **name server** yang tidak muncul pada NS Records yang terlihat oleh publik untuk domain tersebut.
- *Stealth server* memiliki karakteristik sebagai berikut:

1. Organisasi memerlukan DNS publik untuk mengaktifkan akses ke layanan public seperti web, ftp dan lainnya.
2. Organisasi tidak menginginkan dunia melihat host internal baik dengan interogasi (*query* atau *zone transfer*) maupun jika layanan DNS disusupi.

STEALTH SERVER TOPOLOGY



Authoritative Only Server

- Istilah **Authoritave Only** biasanya digunakan untuk menggambarkan 2 (dua) konsep yaitu:

1. Server memberikan Authoritative Response karena merupakan *zone master* atau *slave* dari satu atau lebih *domain*.
2. Server **tidak** melakukan Cache.

- Terdapat 2 (dua) konfigurasi dimana server **Authoritative Only** biasanya digunakan yaitu:

1. Sebagai Public atau Eksternal DNS pada *Stealth (DMZ* atau *Hidden Master)* DNS yang digunakan untuk memberikan *perimeter security*.
2. *High Performance DNS Servers*.

Split Horizon DNS Server

- Istilah *Split Horizon* biasanya digunakan untuk menggambarkan server DNS yang akan memberikan respon berbeda (Alamat IP) berdasarkan alamat sumber, atau karakteristik lainnya dari query.
- Digunakan pada situasi seperti:

a) Pemetaan Geografis.

b) Konsistensi Penamaan.

c) Load Balancing.

ADA PERTANYAAN?

TERIMAKASIH

REFERENSI

- Zytrax, 2017, DNS for Rocket Scientiests, http://zytrax.com/books/dns/, diakses pada 27 Oktober 2017
- Think Like A Computer, DNS Zones Explained, http://www.think-like-a-computer.com/2011/06/11/dns-zones-explained/, diakses pada 13 Nopember 2017