

21. DNS система за именуване.

Процес на резолвинг на
имената.

Защо ни трябва

Чрез **IP адресите** се осъществява адресирането на дейтаграмите, които носят в себе си данните.

Неудобното е, че те са **числа** и трудно се запомнят.

Затова се въвежда система за именуване – **DNS**.

Domain Name System

Domain Name System (DNS) е йерархична разпределена база от данни.

Тя съхранява информация за съответствието между Internet хост имена и IP адреси и обратно, информация за маршрутизиране на ел. поща и др. данни, използвани от Internet приложения.

Клиентите търсят информация в DNS, извиквайки *resolver library*, която изпраща заявки до един от сървърите за имена (*name servers*) и интерпретира отговорите.

BIND софтуерът съдържа сървър за имена *named*, и две библиотеки - *resolver libraries*: *liblwres* и *libbind*.

ISC BIND

BIND (Berkeley Internet Name Domain) е реализация на DNS протоколите и осигурява отворена система за редистрибуция на **основните компоненти** на Domain Name System:

- Domain Name System server (named);
- Domain Name System resolver library;
- средства за верифициране на операциите на DNS server.

Домейни и имена на домейни

Данните, съхранени в DNS са *domain names*, организирани в дървовидна структура.

Всеки възел в дървото се нарича *domain* и му се дава *етикет*.

Името на домейна във възела е поредица от етикетите, показващи пътя от възела до корена (*root*).

В писмена форма се представя като низ от етикети, от дясно наляво, разделени с точки.

Домейни и имена на домейни

Домейните представляват области от имена. Домейните са от първо, второ и трето ниво.

(Ако не се брои root.)

Няма пречки да има домейни от четвърто ниво, но те почти не се използват.

Основният домейн е така нареченият **root** домейн. Той няма име и е един единствен. Представя се с точка.

Домейни и поддомейни

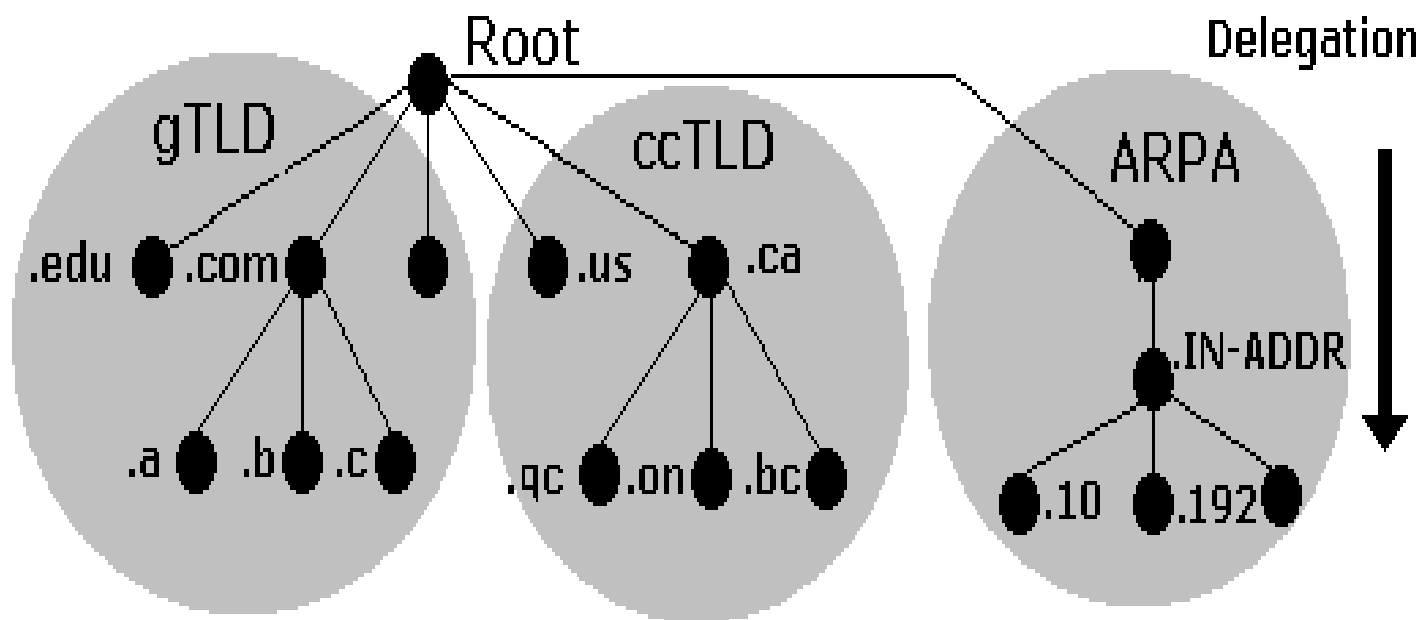
Под него се нареждат домейните от първо ниво, **top-level domain (TLD)**.

Управлението на TLDs е делегирано на различни организации от страна на **ICANN**, която менижира **IANA**, и е отговорна за **DNS root зоната**.

Най-често използвани TLDs са:

generic top-level domains (gTLD) – отворени за регистрация за всеки по света, например: **com, net, org, biz** и др.

DNS йерархия



Домейни и поддомейни

В началото всички те са в САЩ, но после в тях влизат още много имена на обекти извън САЩ, те нарастват твърде много.

Затова се въвежда **друга** голяма група от домейни на **първо ниво**, свързани с **географското** разположение по държави – uk, de, bg и др. Това са **country-code top-level domains (ccTLD)**, показващи принадлежност към държава. Състоят се само от две букви. В повечето случаи съвпадат с кода на страната по **ISO 3166**.

infrastructure top-level domain: Има само една TLD - **Address and Routing Parameter Area (ARPA)**. Управлява се от IANA и има отношение към **обратния резолвинг**.

Цялостното име, което включва домейните и обекта се нарича **URL (uniform resource locator)**. Пример за URL е

`http://www.fmi.uni-sofia.bg`

Първите не-латински web адреси

6 май, 2010 –
исторически за
ICANN

Egypt: مصر

Saudi Arabia:
السعودية

United Arab
Emirates: امارات

Египет, Саудитска
Арабия и ОАЕ:

Първите ccTLDs с
нелатински букви

Ще ги последват
Китай, Тайланд и
др.

URL

В това URL **bg** е името на домейна от **първо** ниво, **uni-sofia** е името на **поддомейна** на **bg** от **второ** ниво, **fmi** е името на домейна от **трето** ниво

www е web-сървър от домейна **fmi**, **http** е името на протокола по който клиента се свързва към съответния обект.

Колкото са точките в едно URL, толкова са нивата на домейните без да се брои root.

В URL **точката** на root **се пропуска** (подразбира се).

Resolving

DNS е йерархична именна система с три компонента – именно пространство (как се изграждат имената), resolver-и и именни сървъри (name servers).

Resolver-те са абонатите в Internet, които знаят URL и искат да получат съответния IP адрес.

Процесът на преобразуване се нарича resolving. Той се извършва от DNS протокола.

DNS протокол

DNS основно използва User Datagram Protocol (**UDP**) на **порт 53** за обслужване на заявки.

DNS заявките се състоят от една единствена UDP заявка от клиента, последвана от един единствен UDP отговор от сървъра.

DNS протокол

Transmission Control Protocol (TCP) се използва, когато в отговора се съдържат повече от 512 bytes или при трансфер на зони.

Някои операционни системи като HP-UX използват TCP за всички заявки.

Зони

За по-лесно администриране пространството с имената е разделено на области, наречени **зони** (*zones*)

Всяка зона **започва от възел** и се простира надолу до “**листата**” (*leaf nodes*) или до възли, където стартират **други зони**.

Данните за всяка зона се съхраняват в **сървър за имена** (*name server*), който отговаря на **запитвания** (*queries*) в рамките на зоната, използвайки *DNS* *протокол*.

Данните, които са обвързани с всяко име на домейн, се съхраняват под формата на **ресурсни записи**, *resource records* (*RRs*).

Зони

От особена важност е да се разбере **разликата** между **зона** и **домейн**, за да се вникне в същността на сървъра за имена.

Зона е **точката на делегиране** на DNS дървото.

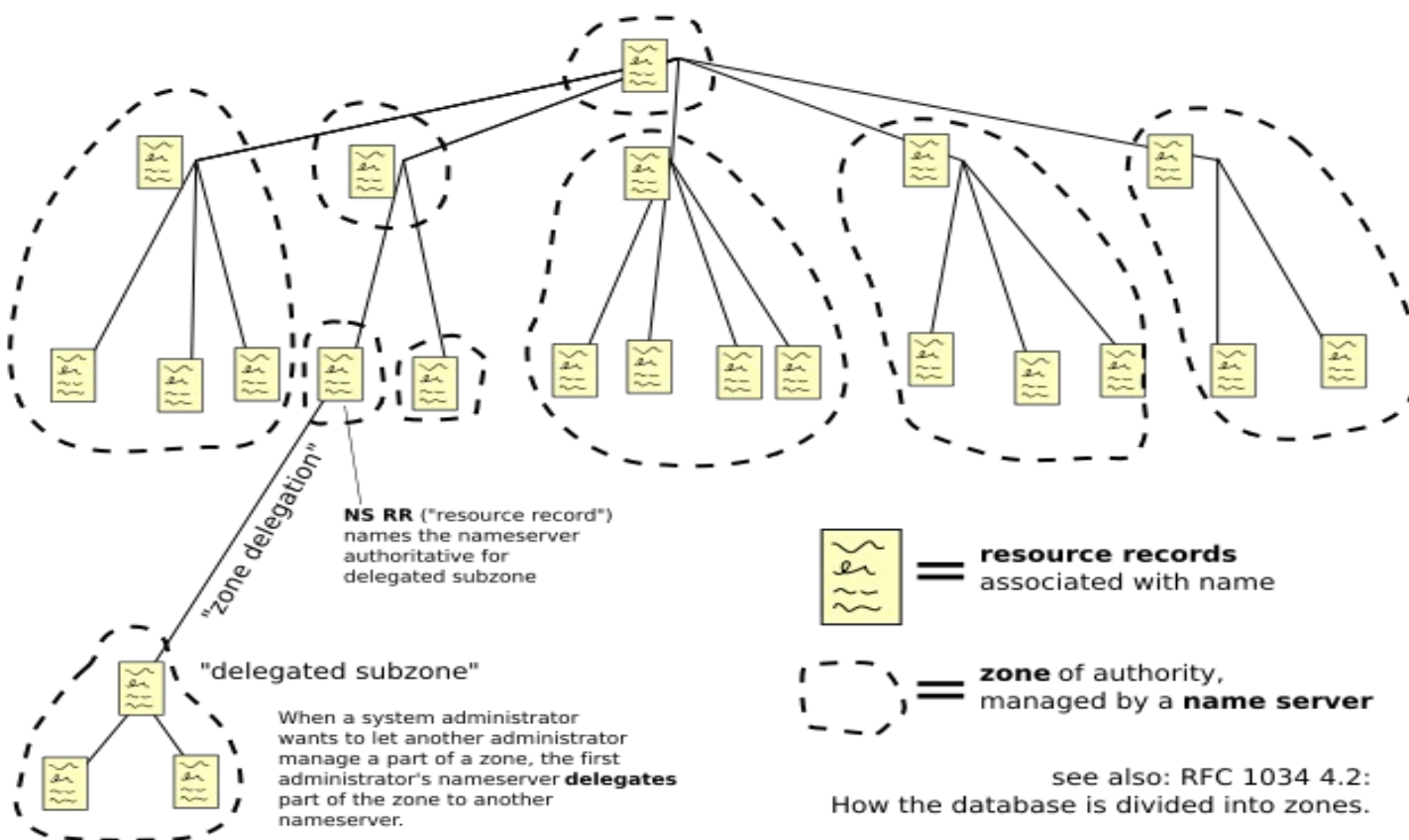
Зоната се състои от тези последователни **части от дървото** на домейните, за които **сървърът за имена има пълна информация** и върху която **има власт**.

Състои се от всички имена на домейни, от дадена точка надолу по дървото с изключение на тези, които са делегирани на други зони.

Точката на делегиране се маркира с един или повече записа: **NS records**, в родителската зона, които трябва да съвпадат с еквивалентни NS записи в корена на делегираната зона.

Зони

Domain Name Space



Зони

Напр., да вземем домейна `example.com`, който включва имена като `host.aaa.example.com` и `host.bbb.example.com`.

`example.com` зоната включва делегирания за зоните `aaa.example.com` и `bbb.example.com`.

Една зона може да съответства точно на един единствен домейн, но може и да включва само част от домейна.

Като останалата част от него да бъде делегирана на други сървъри за имена.

Всяко име в DNS дървото е *domain*, даже ако е *terminal*, т.е няма *subdomains* (поддомейни). Всеки поддомейн е домейн и всеки домейн с изключение на root (коренния) е също поддомейн.

Терминологията не е интуитивна, за по-добро разбиране прочетете RFCs 1033, 1034 и 1035.

master и slave зони

Макар че BIND се нарича "domain name server", той се занимава предимно със зони.

Декларациите **master** и **slave** във файла **named.conf** определят зони а не домейни.

Ако питате някой друг сайт дали иска да бъде *slave* сървър на вашия *domain*, вие всъщност **молите за slave услуга** за някакъв набор от зони.

Файл named.conf

```
###// uni-sofia.bg primary name server boot file
```

```
...
```

```
options {
```

```
    directory "/var/named";
```

```
    pid-file "/var/run/named/named.pid";
```

```
    listen-on port 53 { 62.44.96.1 ; 62.44.96.7 ;};
```

```
    listen-on port 5353 { 62.44.96.1 ; };
```

```
    listen-on-v6 {
```

```
        any;
```

```
    };
```

```
...
```

```
// zone "." IN {  
//     type hint;  
//     file "named.ca";  
//};  
  
...  
  
zone "fmi.uni-sofia.bg" {  
    type slave;  
    file "slaves/fmi.uni-sofia.bg";  
    masters { 62.44.101.1; 62.44.96.7; };  
};
```

```
zone "theo.uni-sofia.bg" {  
    type slave;  
    file "slaves/theo.uni-sofia.bg";  
    masters { 62.44.106.1; 62.44.96.7; };  
};
```

...

```
zone "slav.uni-sofia.bg" {  
    type master;  
    file "slav.uni-sofia.bg";  
};
```

named.conf B ns.theo

```
zone "." IN {  
    type hint;  
    file "named.ca";  
};  
zone "theo.uni-sofia.bg" IN {  
    type master;  
    file "theo.uni-sofia.bg";  
};  
zone "0.0.127.in-addr.arpa" IN {  
    type master;  
    file "named.local";  
};  
zone "106.44.62.in-addr.arpa" IN {  
    type master;  
    file "106.44.62.in-addr.arpa";  
};
```

Видове зони

Master Сървърът чете данните за зоната директно от локалния диск (т.е от **zone file**) и е овластен да дава отговори за тази зона.

Hint В тази зона се дефинират **root-servers**.

Slave Зона **slave** е реплика на **master** зона и получава данни за тази зона чрез зонов трансфер. **slave** ще даде овластен отговор за зоната, само ако има **валидни** (не timed out) данни за зоната.

Редът **masters** определя IP адрес/и на master сървър/и, с които **slave** контактува, за да **refresh** или **update** копие на зоната.

Authoritative Name Servers

- Всяка зона се обслужва най-малко от един **овластен** сървър за имена (*authoritative name server*), който държи всички данни за зоната.
- За по-висока надеждност се препоръчва зоната да има два или повече такива сървъри.
- В отговорите на *authoritative servers*, в пакета с отговора, е вдигнат бит "**authoritative answer**" (**AA**). Така по-лесно се **диагностицират** (debugging) DNS конфигурациите с инструменти като **dig**.

Primary Master

authoritative server, където се поддържа
главното (*master*) копие на данните за *зоната*.

Нарича се *primary master* сървър или просто
primary.

Той зарежда съдържанието на зоната от
локален файл, редактиран ръчно или
генериран от някакъв друг локален файл.

Този файл се нарича *зонов* - *zone file* или
master file.

Slave Servers

Другите authoritative servers, *slave* сървъри (известни още като *secondary*) зареждат съдържанието на зоната от друг сървър чрез процес на **репликация** - *zone transfer*.

Обикновено данните се прехвърлят директно от primary master, но е **ВЪЗМОЖНО** и от **друг slave**.

Т.е, *slave server* може да действа като *master* за подчинен *slave server*.

Caching Name Servers

resolver библиотеки, които присъстват в повечето операционни системи, са *stub resolvers*, т.е. те не са способни да изпълняват пълния процес на DNS резолюция, “говорейки” директно с authoritative servers.

Те разчитат на локален сървър за имена, който да изпълнява резолюцията вместо тях.

Такъв сървър се нарича “**recursive**” (**рекурсивен**) сървър за имена, защото изпълнява *рекурсивни търсения* за сметка на локалните клиенти.

Caching (*recursive*) Servers

За да се подобри производителността, рекурсивните сървъри кешират резултатите от търсенията, които са изпълнили.

Процесите на рекурсия и кеширане са взаимно свързани, на термините *recursive server* и *caching server* често се гледа като на синоними.

Перодът от време, за който един запис се държи в кеша, се контролира от Time To Live (TTL) полето в него.

Caching Servers. Forwarding.

Кеширащият сървър за имена не е необходимо да изпълнява сам пълното рекурсивно търсене.

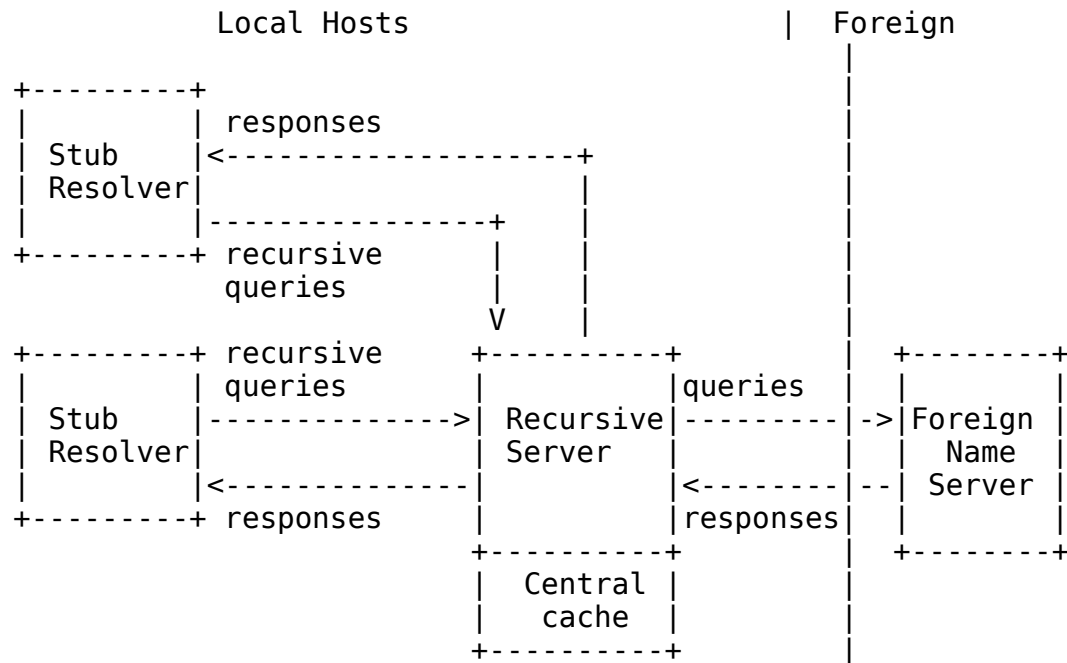
Вместо това той **препраща** (*forward*) някои или всички заявки, които не може да удовлетвори, от своя кеш **към кеша на друг сървър** за имена, който се определя като ***forwarder***.

Многофункционални сървъри

Сървърът за имена BIND може **едновременно** да бъде **и master** за някои зони, **и slave** за други зони, **и кеширащ** (рекурсивен) сървър за определен брой локални клиенти.

Все пак, функциите на овластени (**authoritative**) услуги за имена и такива на **caching/recursive** са **логически разделени**.

Затова е по-изгодно **да работят на различни машини**. Така ще се **повиши надеждността и сигурността**.



Пример на зонов файл

```
$ORIGIN .
$TTL 3600      ; 1 hour
theo.uni-sofia.bg    IN SOA  ns.theo.uni-sofia.bg. root.ns.theo.uni-sofia.bg. (
                        2009030901 ; serial
                        3600      ; refresh (1 hour)
                        900       ; retry (15 minutes)
                        3600000    ; expire (5 weeks 6 days 16 hours)
                        3600      ; minimum TTL (1 hour)
                        )
NS      ns.theo.uni-sofia.bg.
NS      ns.uni-sofia.bg.
NS      ady.uni-sofia.bg.
MX      5 ns.theo.uni-sofia.bg.
MX      10 ns.uni-sofia.bg.
MX      20 ady.uni-sofia.bg.
$ORIGIN theo.uni-sofia.bg.
a02      A      62.44.106.13
PPPoE-Server  A      62.44.106.2
assitenti  A      62.44.106.76
```

Ресурсни записи

SOA определя кой е първичният сървър и как се обработват данните към него.

NS съдържа информация кои DNS сървъри са отговорни за този домейн.

MX указва име на хост, готов да приема електронна поща в рамките на домейн.

Адресните записи съдържат съответствие между име и IP-адрес. Имат следния формат:

<hostname> A <IP address>

Ресурсни записи

В DNS е възможно създаването на прякори, т.е. няколко имена да отговарят на един и същ IP адрес. Това става с помощта на **CNAME-записите**, които имат следния формат:

mail	CNAME	tiger
proxy	CNAME	tiger
tiger	A	62.44.118.1

Ресурсни записи в IPv6

...		
fmi-gw	A	62.44.127.15
	AAAA	2a01:288:8000::15
...		
ivkm-gw	A	62.44.127.51
	AAAA	2a01:288:8000::51
...		
rec-gw	A	62.44.127.61
	AAAA	2a01:288:8000::61

Root сървъри за имена

Кореновият сървър за имена (**root nameserver**) е DNS сървър, който отговаря на запитвания относно имената в **коренния домейн** и отправя заявките към конкретни **top-level domain (TLD)**, т.е към техните сървъри за имена.

Всички имена в Internet завършват с точка . - напр., "**www.wikipedia.org.**" Но съвременният DNS софтуер не се нуждае от нея, когато се опитва да транслира домейн име в IP адрес.

Празният низ след крайната точка се нарича коренов домейн (**root domain**), а всички останали (т.е. .com, .org, .net, и т.н.) се съдържат вътре в коренния (root).

Root сървъри за имена

Когато компютър в Internet **иска да** открие съответствие (**resolve**) за домейн име, започва от **дясно на ляво**, запитвайки всеки **name server** поред относно елемента от ляво.

root nameservers (отговарящи за домейна **.**) знаят кои сървъри са отговорни за top-level домейните.

Всеки такъв домейн (напр. **.bg**) има свой набор от сървъри, които от своя страна делегират към nameserver-те, отговарящи за отделните имена на домейни (като **uni-sofia.bg**), които пък отговарят на запитванията за IP адреси на поддомейни или хостове (напр. **Poshta.fmi**).

Root сървъри за имена

Информацията не се променя често, затова се кешира, така че **DNS търсенията** към **root nameservers** са относително **редки**.

Но в Internet има доста некоректно конфигурирани системи, които генерират трафик към root servers.

Напр., **заявки с източник адрес 0.0.0.0** (т.е. където и да е, навсякъде) отиват натам.

В момента има **13 root name servers**, като имената им са с формат **буква.root-servers.net** (буква е от А до М)

Root сървъри за имена

A.ROOT-SERVERS.NET.

B.ROOT-SERVERS.NET.

...

M.ROOT-SERVERS.NET.

По-подробна информация:

<http://www.internic.net/zones/named.root>

http://en.wikipedia.org/wiki/Root_nameserver

<http://www.root-servers.org/>

Root сървъри за имена

В момента С, F, I, J и K сървърите се намират на много места, на различни континенти, като за целта се използват **anycast** анонси, с което се осигурява разпределена услуга.

Това **предпазва** и от DoS и DDoS атаки.

named.ca

;This file holds the **information on root name servers**
needed to initialize cache of Internet domain name
servers

```
...  
.          3600000 IN NS  A.ROOT-SERVERS.NET.  
A.ROOT-SERVERS.NET.  3600000  A   198.41.0.4  
.          3600000  NS  B.ROOT-SERVERS.NET.  
B.ROOT-SERVERS.NET.  3600000  A   128.9.0.107  
.          3600000  NS  C.ROOT-SERVERS.NET.  
C.ROOT-SERVERS.NET.  3600000  A   192.33.4.12
```

Регистриране на име

Регистрирането на име не е автоматично, а става чрез специална заявка към **регистратор** за съответния домейн или фирма, на която са делегирани съответни права за регистрация.

За домейна **.bg** регистратор е **register.bg**.

Резолвинг на имена

За да се използва системата на URL-имената в клиента (resolver) трябва да има **агент**, който да може **да работи с URL** - началото на resolving процеса.

Освен това в клиента трябва да има и **малък кеш**, в който да се съхранява **информация за вече заявени** и resolve-нати адреси за този клиент.

Също така, клиентът трябва да разполага с адрес на DNS сървър, който отговаря за съответната област.

Резолвинг на имена

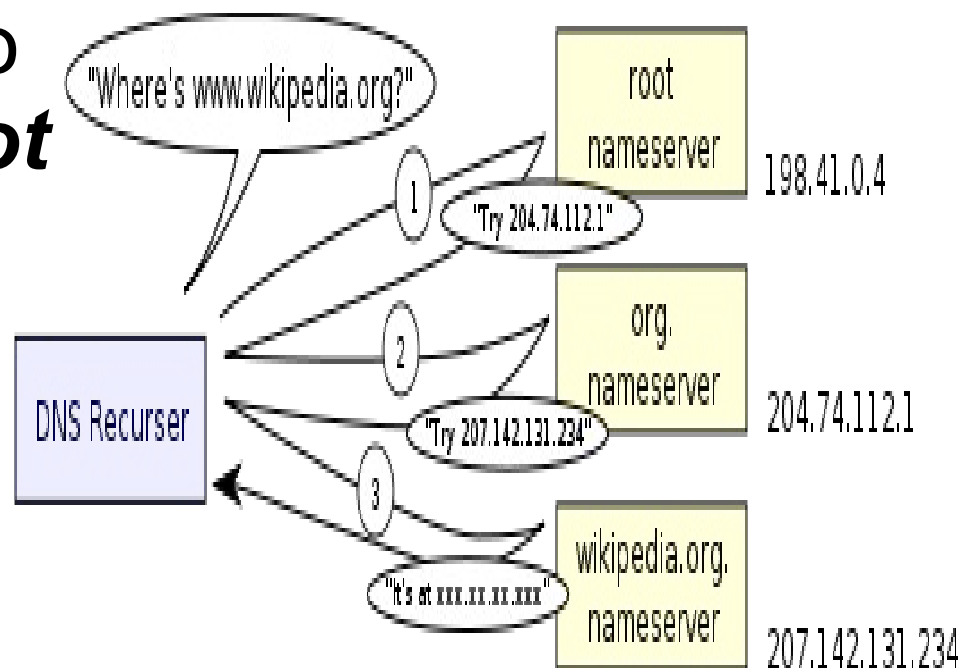
Когато към агента се подаде URL за resolve-ане той първо проверява **дали отговора** не стои **в кеша**.

Ако не, той изпраща заявка до DNS сървър.

DNS сървърът може да формира **три типа заявки** – рекурсивна, итеративна или инверсна.

Резолвинг на имена

- Една тежка процедура, която товари много **root** сървърите



Рекурсивна заявка

При **рекурсивна заявка** DNS сървърът има прилежащ към него **друг сървър за имена**.

Този сървър също може да има **кеш**, който евентуално да съдържа отговора.

Сървърът може да съдържа отговора в **своите зонални файлове**.

Ако и двата случая не са налице, но има конфигуриран друг сървър за имена, той ще изпрати заявката към него и т.н.

В един момент някой сървър по описаната верига може да направи **рекурсивната заявка в итеративна**.

Итеративна заявка

При итеративната заявка сървър е в свободното Internet пространство.

Той започва да раздробява съответното URL и постъпково, съгласно структурата на URL започва resolve-то.

Първо се изпраща заявка към root-сървъра, като се иска адреса на сървъра, който отговаря за TLD.

След това се праща заявка към сървъра от първо ниво за адреса на сървъра, който отговаря за домейна от второ ниво, участващ в URL-то и т.н.

Връщане на отговор

Например, URL www.fmi.uni-sofia.bg

След като една рекурсивна заявка е превърната в итеративна и итеративната заявка е изпълнена,

полученият отговор се връща обратно по веригата на рекурсивната заявка и се стига обратно до resolve-ра, който слага получения отговор в кеша си.

Инверсни заявки

Инверсните заявки служат за обратен resolve – по IP адрес да се получи URL.

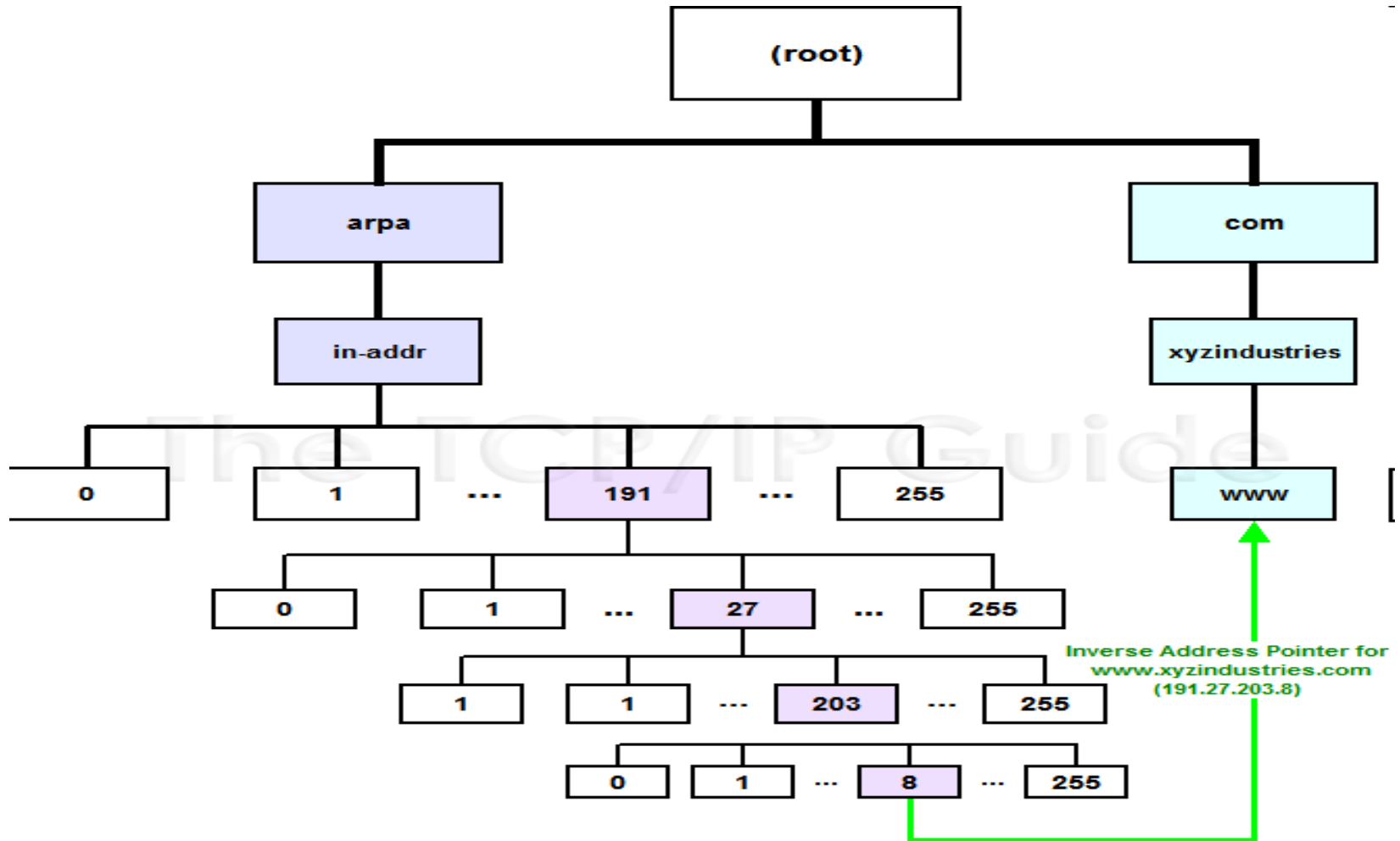
В сървърите за имена има специални записи, предназначени за инверсни заявки: домейна *in-addr.arpa* и (Pointer) PTR записите.

Йерархията на имената тук е спазена с помощта на специалния домейн “IN-ADDR.ARPA”, разположен в резервирания .ARPA TLD (Address and Routing Parameter Area)

“IN-ADDR” означава “INternet ADDRess”.

За IPv6 reverse lookup домейнът е *ip6.arpa*

IN-ADDR.ARPA Reverse Name Resolution Hierarchy



in-addr.arpa имена

in-addr.arpa имената се записват в ред, **обратен на записа на IP** адресите – от младши към старши или отляво надясно.

Например, машина с IP адрес **10.1.2.3** ще има in-addr.arpa име **3.2.1.10.in-addr.arpa**.

Това име ще има PTR ресурсен запис:

\$ORIGIN	2.1.10.in-addr.arpa
3 PTR	foo.example.com.

\$ORIGIN се поставят само за пояснение, но не са задължителни.

Reverse B named.conf

```
zone "0.0.127.in-addr.arpa" {  
    type master;  
    file "local/0.0.127.in-addr.arpa";  
};  
  
zone "96.44.62.in-addr.arpa" {  
    type master;  
    file "96.44.62.in-addr.arpa.signed";  
};  
  
...  
  
zone "118.44.62.in-addr.arpa" {  
    type slave;  
    file "slaves/118.44.62.in-addr.arpa";  
    masters { 62.44.118.1; 62.44.96.7; };  
};
```

named.conf във ns.flaw

```
zone "localhost" IN {  
    type master;  
    file "localhost.zone";  
    allow-update { none; };  
  
...  
  
zone "0.0.127.in-addr.arpa" IN {  
    type master;  
    file "named.local";  
    allow-update { none; };  
};  
  
...  
  
zone "118.44.62.in-addr.arpa" IN {  
    type master;  
    file "118.44.62.in-addr.arpa";  
};
```

named.local

TTL 86400

```
@      IN      SOA      localhost. root.localhost. (
                                1997022701 ; Serial
                                28800      ; Refresh
                                14400      ; Retry
                                3600000    ; Expire
                                86400      ; Minimum
                                )
```

```
      IN      NS       localhost.
```

```
1      IN      PTR     localhost.
```

localhost.zone:

\$TTL 86400

\$ORIGIN localhost.

```
@              1D IN SOA      @ root (
                                42          ; serial (d. adams)
                                3H          ; refresh
                                15M         ; retry
                                1W          ; expiry
                                1D )        ; minimum
```

```
1D IN NS       @
```

```
1D IN A        127.0.0.1
```

ns.flaw.uni-sofia.bg reverse file

```
;ns.flaw.uni-sofia.bg reverse file
$TTL 84600 ; 10 hours
@ IN SOA ns.flaw.uni-sofia.bg. root.flaw.uni-sofia.bg. (
    2009031901 ; Serial
    3600 ; Refresh
    900 ; Retry
    3600000 ; Expire
    3600 ) ; Minimum
;
IN NS ns.flaw.uni-sofia.bg.
IN NS ns.uni-sofia.bg.
IN NS ady.uni-sofia.bg.
;
IN MX 5 ns.flaw.ucc.uni-sofia.bg.
IN MX 15 ady.uni-sofia.bg.
IN MX 20 ns.uni-sofia.bg.
;
1 IN PTR ns.flaw.uni-sofia.bg.
1 IN PTR ns.law.uni-sofia.bg.

129 IN PTR priam.flaw.uni-sofia.bg.
```


Classless reverse DNS

```
zone "64.96.44.62.in-addr.arpa" {  
    type slave;  
    masters { 62.44.96.80 ; };  
    file "slaves/64.96.44.62.in-addr.arpa";  
};
```

В миналото Internet регистраторите и ISPs алокираха **октет-базирани IP адресни блокове** от по 256 (Class C) или по-големи - класове B и A.

С въвеждането на CIDR се алокират по-малки адресни блокове. RFC 2317 решава този проблем чрез **делегиране на права за администриране**:

64.96.44.62.in-addr.arpa IN SOA ns2-it.fmi.uni-sofia.bg.
misho.fmi.uni-sofia.bg. (

2010040802 ; serial

28800 ; refresh (8 hours)

7200 ; retry (2 hours)

604800 ; expire (1 week)

86400 ; minimum (1 day)

)

NS ns.uni-sofia.bg.

NS ady.uni-sofia.bg.

IPv6 reverse

Обратният DNS резолвинг за IPv6 адреси използва домейна **ip6.arpa**.

IPv6 адресите се представят като последователност от **nibbles** (полуоктети – 16-ни цифри) в обратен ред (както при Ipv4).

Например, домейн за IPv6 address **2001:db8::567:89ab**:

b.a.9.8.7.6.5.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.
8.b.d.0.1.0.0.2.ip6.arpa.

3.0.0.0.0.0.8.8.8.2.0.1.0.a.2.ip6.arpa

\$ORIGIN .

\$TTL 86000 ; 23 hours 53 minutes 20 seconds

3.0.0.0.0.0.8.8.8.2.0.1.0.a.2.ip6.arpa IN SOA openfmi.net. root.fmi.uni-sofia.bg.

(

1 ; serial

3600 ; refresh (1 hour)

3600 ; retry (1 hour)

3600000 ; expire (5 weeks 6 days 16 hours)

86400 ; minimum (1 day)

)

NS ns.uni-sofia.bg.

NS ady.uni-sofia.bg.

NS openfmi.net.

\$ORIGIN 3.0.0.0.0.0.8.8.8.2.0.1.0.a.2.ip6.arpa.

1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0 PTR border.fmi.uni-sofia.bg.

Диагностични и администраторски инструменти

Dig - domain information groper:

`dig @server domain query-type query-class`

(вж. `#man dig`)

e.g. `dig @localhost uni-sofia.bg` или

`dig @ns.uni-sofia.bg spectrum.net`

`root@ns ~]# host portal.uni-sofia.bg`

`portal.uni-sofia.bg has address 62.44.96.22`

`[root@ns ~]# host 62.44.96.22`

`22.96.44.62.in-addr.arpa domain name
pointer portal.uni-sofia.bg.`

rndc

С помощта на програмата **remote name daemon control (rndc)** администраторът контролира работата на **name** сървъра.

След всяка промяна в **zone** и/или **reverse** файл се изпълнява **rndc reload** (e.g.):

```
[root@ns named]# vim uni-sofia.bg
```

```
[root@ns named]#rndc reload uni-sofia.bg
```

или

```
[root@ns named]# vim 96.44.62.in-addr.arpa
```

```
[root@ns named]#rndc reload 96.44.62.in-addr.arpa
```

DNSSEC

За борба срещу DNS измами като **cache poisoning**:
електронно подписване на ресурсните записи в зоните с
помощта на криптография с публичен ключ (RSA и
DSA базирана);
урежда съхраняването на извършените върху
ресурсните записи електронни подписи под формата
на **допълнителни ресурсни записи** в същата зона;
дава възможност за **проверка на извършените върху**
ресурсните записи електронни подписи от страна на
рекурсивни/кеширащи сървъри за имена, с цел
проверка на автентичността на записите.

DSA - Digital Signature Algorithm

*RSA - Ronald L. **R**ivest, Adi **S**hamir und Leonard **A**dleman*