Computer Networks
Jacobs University Bremen
Jovan Shandro

Course: CO20-320301
May 5, 2020

Homework 6

Problem 6.1

Solution:

a) Concerning the lookup process, first a query is sent to the DNS resolver which makes it first see if the name that is being asked for has been cached or not. In the first case, the address is directly returned whereas in the latter, the full name is split and the resolver initiates the querying label by label from right to left. Each of the queries returns name servers that keep getting more and more specific until the exact match is found with the full domain name.
To resolve the domain name **grader.eecs.jacobs-university.de** into an IPv6 address we run the command `dig grader.eecs.jacobs-university.de AAAA`. You can find the output in the dig.txt file.
Now, concerning the output of the dig command we can see that the EDNS specification was used. Then we see the output has a question and answer section. In the question section we see what was asked, which in out case is the IP address of grader's domain name. Under the Answer section we see the results that were found for what we asked, and in our case we see that it found out that **grader.eecs.jacobs-university.de** contains a CNAME record that points towards **cantaloupe.eecs.jacobs-university.de** and then it shows the result of the search, and it is the IP address of `2001:638:709:3000::29`.

b) A SRV record has the form:
_service._proto.name. TTL class SRV priority weight port target
In which
`service`: the symbolic name of the desired service.
`proto`: the transport protocol of the desired service; this is usually either TCP or UDP.
`name`: the domain name for which this record is valid, ending in a dot.
`TTL`: standard DNS time to live field.
`class`: standard DNS class field (this is always IN).
`priority`: the priority of the target host, lower value means more preferred.
`weight`: A relative weight for records with the same priority, higher value means higher chance of getting picked.
`port`: the TCP or UDP port on which the service is to be found.
`target`: the canonical hostname of the machine providing the service, ending in a dot.

An example would be *_sip._tcp.example.com. 86400 IN SRV 0 5 5060 sipserver.example.com.* As for the difference between priority and weight, a client first tries to contact the target host with the lowest priority and in case there are 2 entries with same priority then the one with higher weight is selected. It is defined in RFC 2782.

c) In favor:
SRV resource record would allow websites to not be served only in port 80, but on any port which might provide flexibility in case for some reason port 80 might be blocked.
It would be able to redirect traffic to different domains under different ports, and even if a host might fail, another host can be tried.
Would made client side balancing possible.

Cons:
Being able to serve HTTP on any port would mean that the traffic might even be able to pass firewalls which creates a huge security vulnerability.
It will require more than just one query to perform the lookup.

d) Traditional DNS messages have a size upper bound of 512 bytes, which is not suitable for the current rate at which the internet is growing. That is where EDNS0 is introfuced, in which E refers to extension mechanisms, so it allows for messages that pass the previous upper bound of 512 bytes. It is defined in RFC6891. As for the last question, the CLASS field refers to the requestor's UPD payload size, whereas the TTL field is used to store extended RCODE and flags.

e) The results I got were for google.com:

IPv4 DNS Address	A	AAAA
1.1.1.1	216.58.208.46	2a00:1450:4001:815::200e
8.8.8.8	172.217.22.78	2a00:1450:4001:81c::200e
9.9.9.9	172.217.16.142	2a00:1450:4001:806::200e

So we can easily observe that they all return different answers.

Problem 6.2

Solution:

a) Multicast DNS (mDNS) protocol resolves hostnames to IP addresses within small networks that do not include a local name server. It uses multicast UDP packets and it is defined in RFC6762. It differs from the regular DNS protocol as it sends requests to the multicast address 224.0.0.251 in iPv4 or ff02::fb in IPv6. There is a delay of a random number up to 500ms to prevent collisions. Then the first response is picked up and it resolves the address. Another difference is that it can also use continuous querying until no response is further needed.

b) DNS-SD is defined in RFC6763 and it gives clients the ability to discover a named list of service instances using DNS queries. THe way how it works, it uses DNS PTR's which are pointers to some other part of the namespace. A client fires a DNS PTR query and will receive a set of names (if any) which will be the SRV/TXT pairs.

References:

https://en.wikipedia.org/wiki/SRV_record
https://en.wikipedia.org/wiki/Multicast_DNS
http://www.dns-sd.org/