DNS Wildcards

(Not so simple as they seem)

Carsten Strotmann, Alan Clegg and the ISC Team

Welcome

Welcome our Webinar of DNS wildcards

In this Webinar

- What are DNS Wildcards
- Examples of DNS Wildcard expansion
- DNS Wildcards and DNSSEC
- Important DNS Wildcard rules

DNS wildcards

About DNS wildcards

- The DNS protocol allows the definition of *wildcard* DNS records to synthesize DNS answers from authoritative servers
  - DNS wildcards are domain names (owner names) with the leftmost label being a single asterisk `*`
  - There are no other wildcard characters in DNS other than `*`
- DNS wildcards have their own rules that are different from other systems with wildcards (regular expressions, Unix shell filename globbing, DOS/Windows filename wildcards …)

RFC Standards

- DNS wildcards have been originally defined in the “original” DNS RFC 1034, and have been updated in RFC 6672 (DNAME Redirection in the DNS) and RFC 4592
  - RFC 4592 “The Role of Wildcards in the Domain Name System” is a must read to understand all nuances of DNS wildcards

Wildcard examples

- DNS wildcard examples (from different zones):

```
*.example.com.            3600 IN A  192.0.2.80
*.company.example.com.     600 IN MX 10 mail.example.net.
*.example.               86400 IN TXT "a wildcard"
```

Non-Wildcard examples

- not wildcard records (just "normal" domain names):

```
*test.example.de.   400 IN A   192.0.2.11
test.*.example.com. 600 IN TXT "not a wildcard"
**.example.com.     600 IN TXT "also not a wildcard"
```

DNS Zone with Wildcards

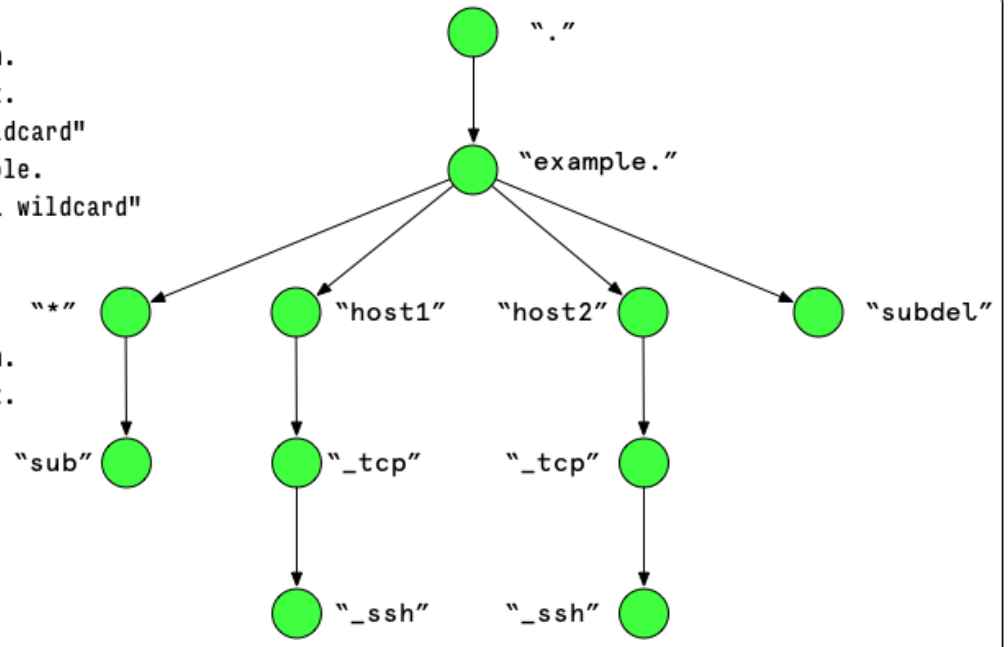
- Example zone with wildcards (from RFC 4592):

```
$ORIGIN example.
example.                 3600 IN  SOA   <SOA RDATA>
example.                 3600 IN  NS    ns.example.com.
example.                 3600 IN  NS    ns.example.net.
*.example.               3600 IN  TXT   "this is a wildcard"
*.example.               3600 IN  MX    10 host1.example.
sub.*.example.           3600 IN  TXT   "this is not a wildcard"
host1.example.           3600 IN  A     192.0.2.1
_ssh._tcp.host1.example. 3600 IN  SRV   <SRV RDATA>
_ssh._tcp.host2.example. 3600 IN  SRV   <SRV RDATA>
subdel.example.          3600 IN  NS    ns.example.com.
subdel.example.          3600 IN  NS    ns.example.net.
```

DNS Wildcard Query examples

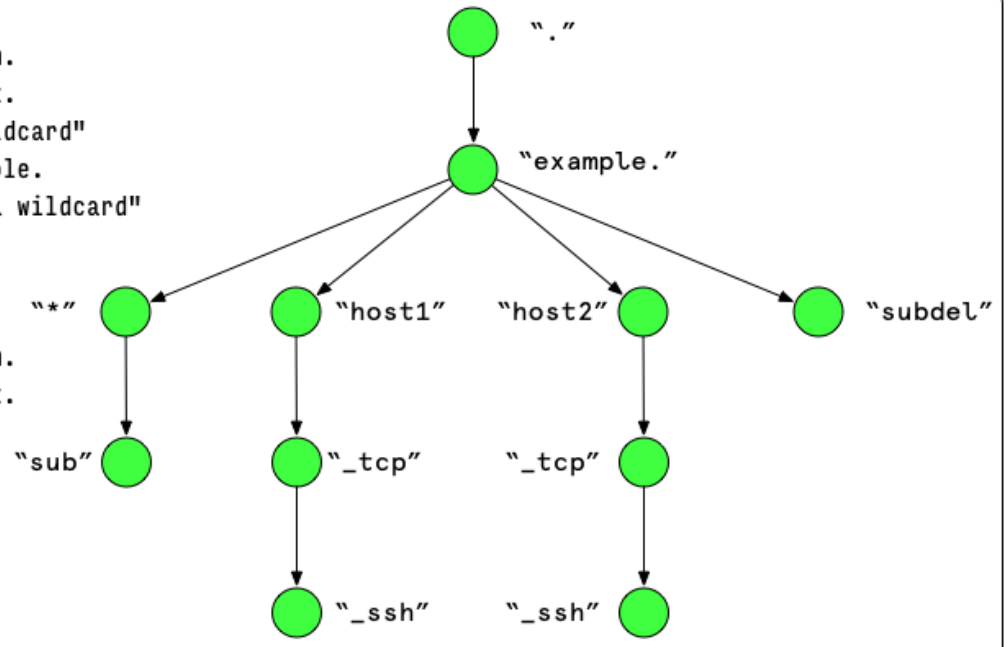
Our example zone

```
example.                    3600 IN  SOA   <SOA RDATA>
example.                    3600     NS    ns.example.com.
example.                    3600     NS    ns.example.net.
*.example.                  3600     TXT   "this is a wildcard"
*.example.                  3600     MX    10 host1.example.
sub.*.example.              3600     TXT   "this is not a wildcard"
host1.example.              3600     A     192.0.2.1
_ssh._tcp.host1.example.    3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.    3600     SRV   <SRV RDATA>
subdel.example.             3600     NS    ns.example.com.
subdel.example.             3600     NS    ns.example.net.
```



Question 1

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

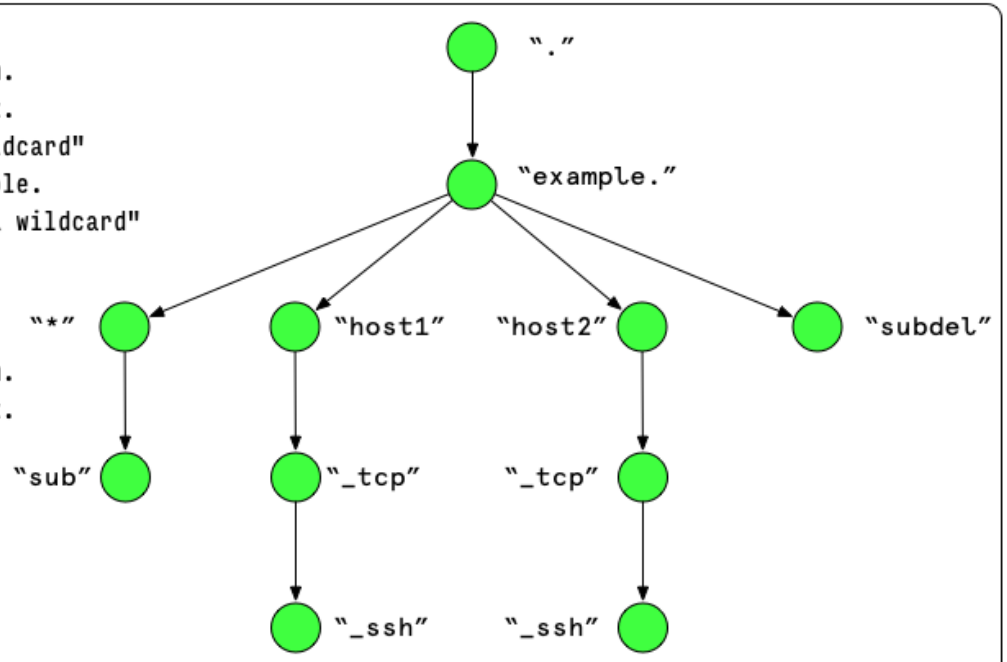


Query: host3.example. ? IN MX ?

Answer:?

Answer 1

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

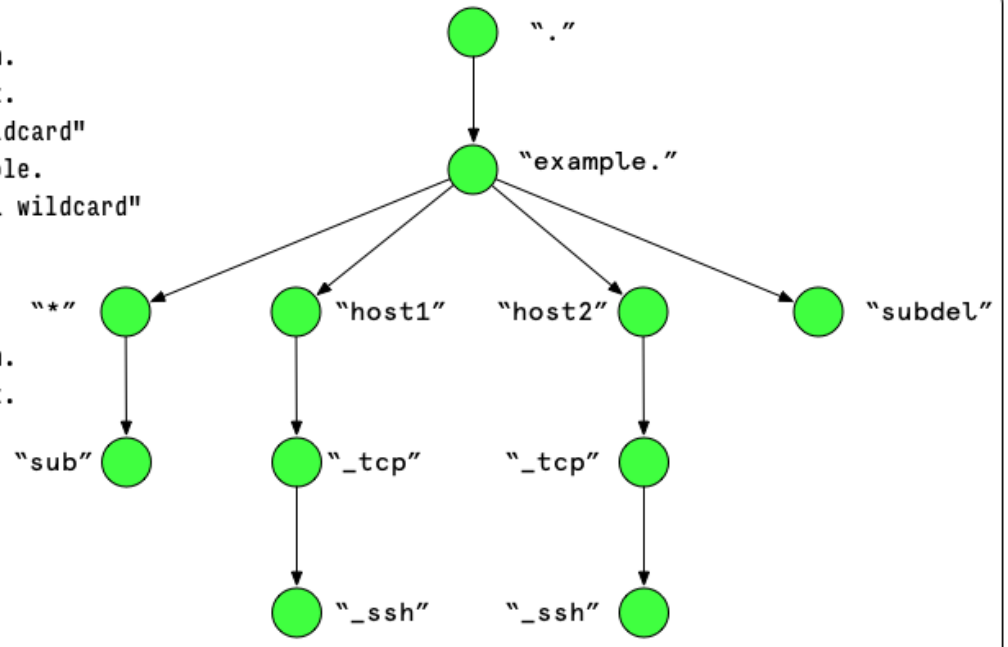


Query: host3.example. ? IN MX ?

Answer: host3.example. 3600 IN MX 10 host1.example.

Question 2

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

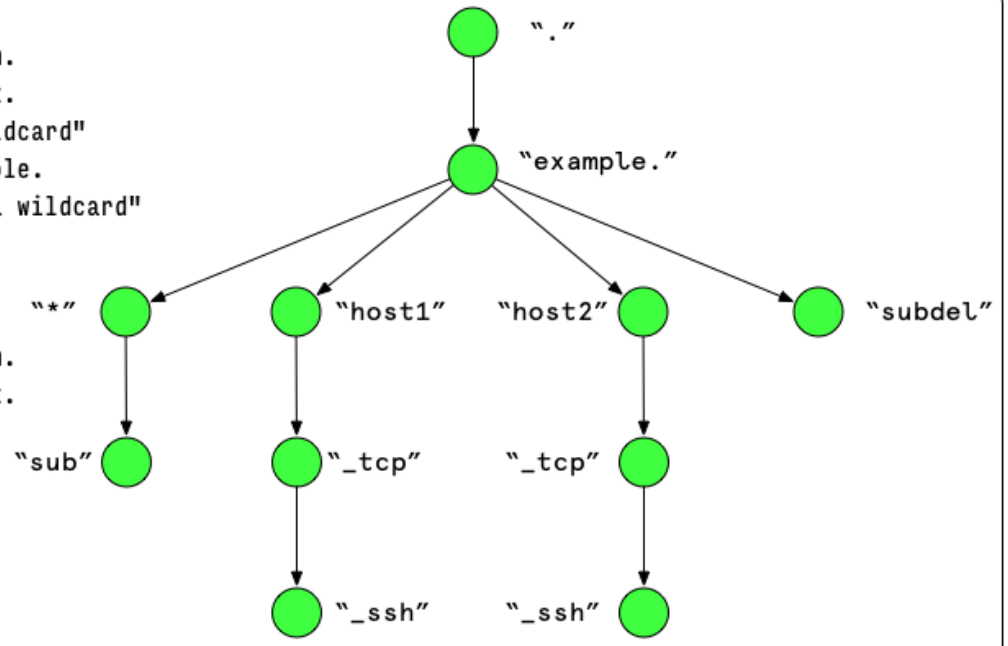


Query: host3.example. ? IN A ?

Answer:

Answer 2

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

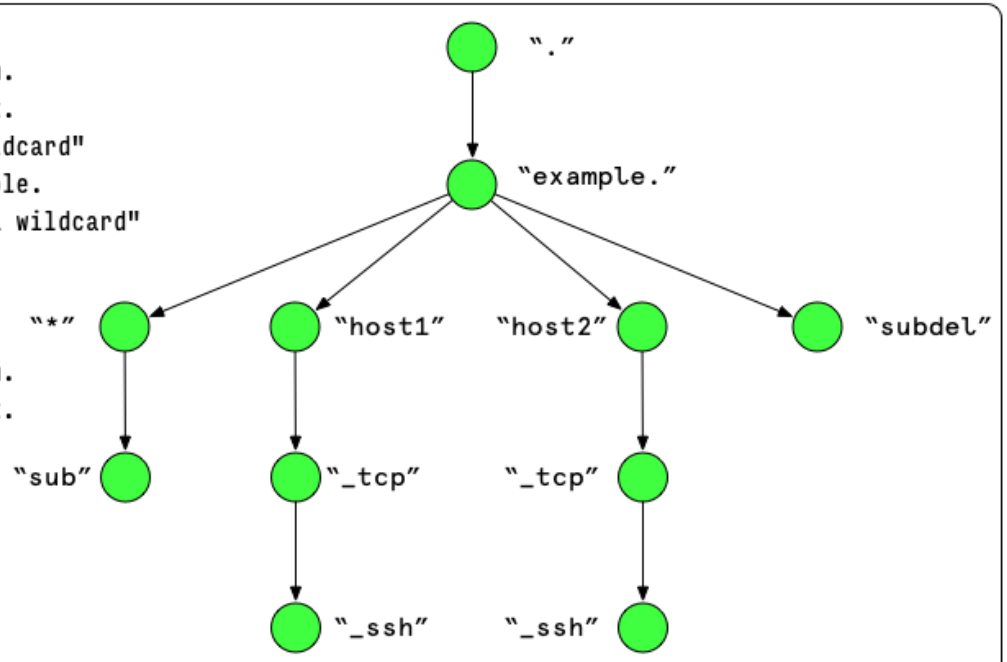


Query: host3.example. ? IN A ?

Answer: NOERROR / NODATA (Answer = 0)

Question 3

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

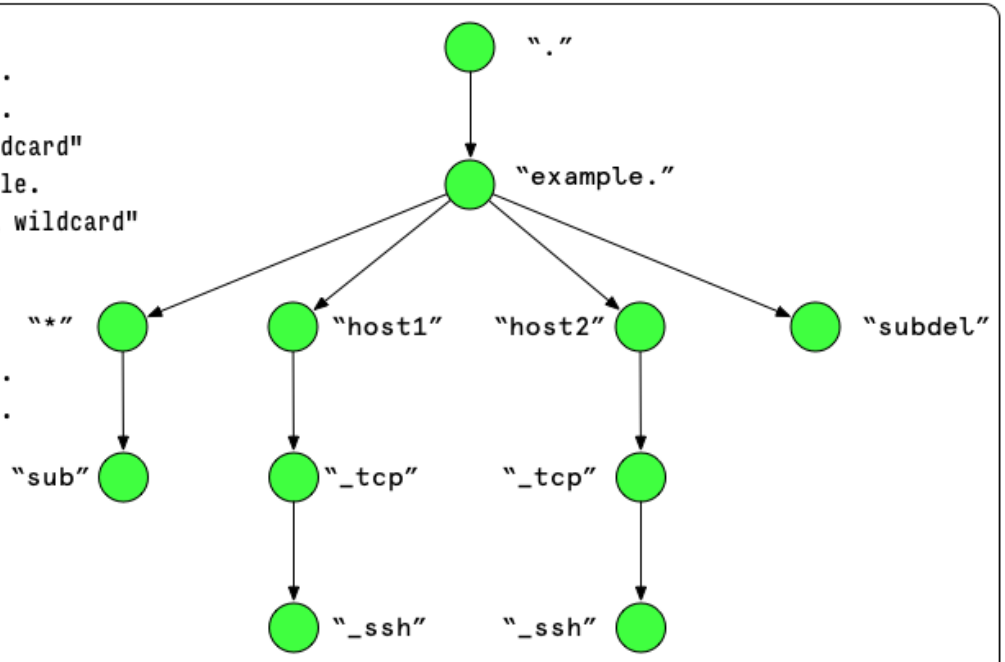


Query: foo.bar.example. ? IN TXT ?

Answer:

Answer 3

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

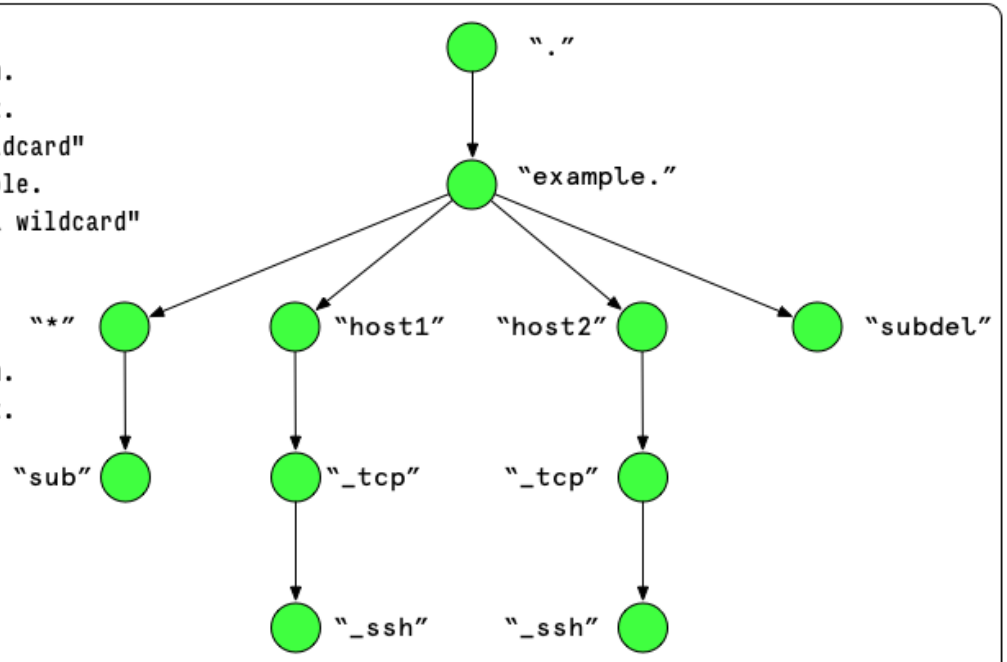


Query: foo.bar.example. ? IN TXT ?

Answer: foo.bar.example. 3600 IN TXT “this is a wildcard”

Question 4

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

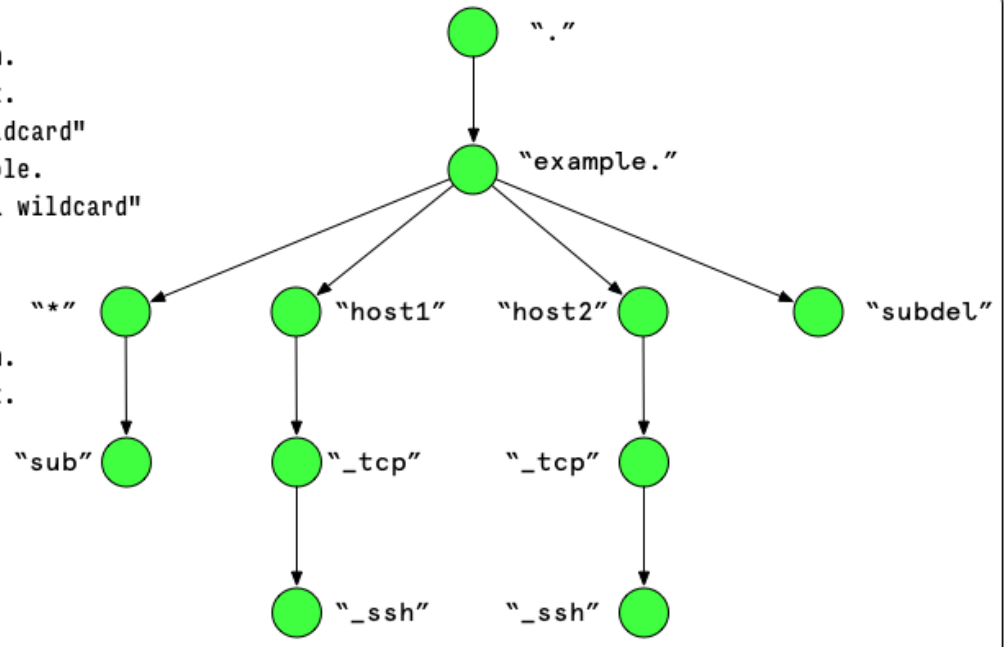


Query: host1.example. ? IN MX ?

Answer:

Answer 4

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

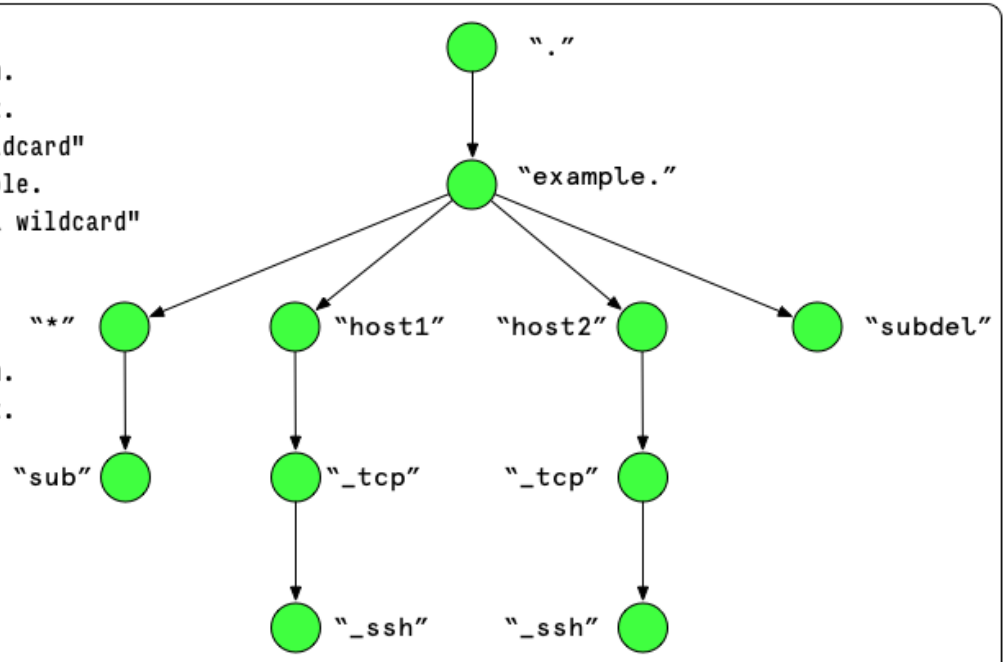


Query: host1.example. ? IN MX ?

Answer: NOERROR / NODATA (Answer = 0)

Question 5

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

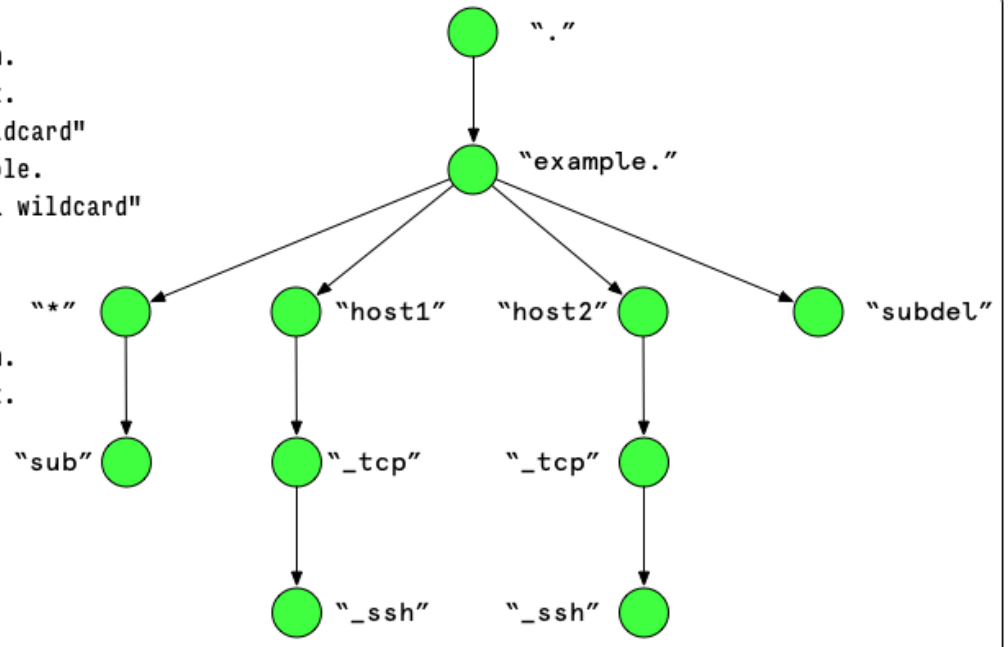


Query: sub.*.example. ? IN MX ?

Answer:

Answer 5

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

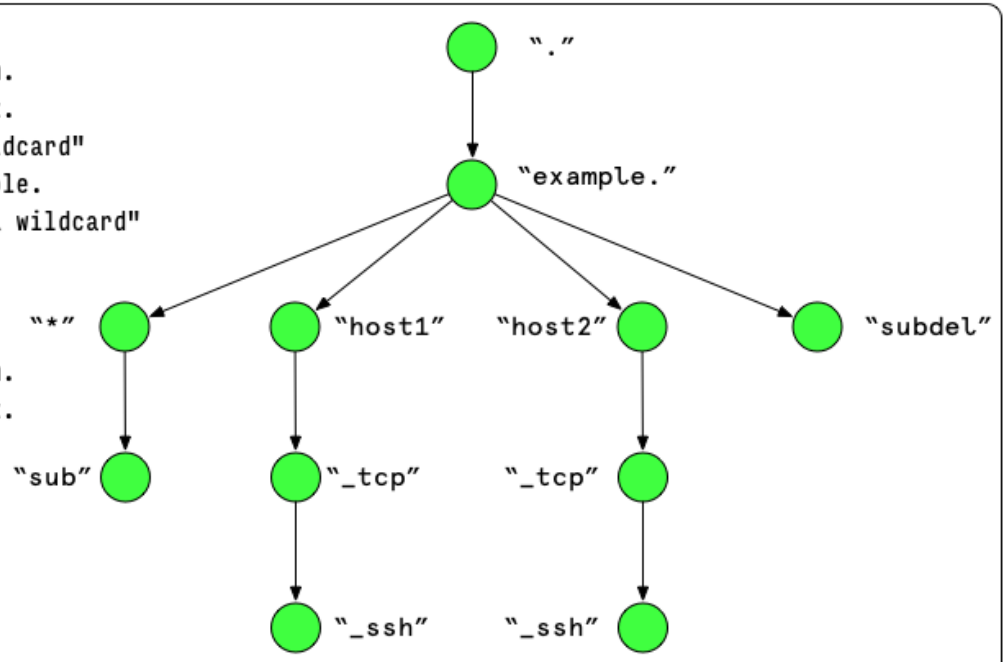


Query: sub.*.example. ? IN MX ?

Answer: NOERROR / NODATA (Answer = 0)

Question 6

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

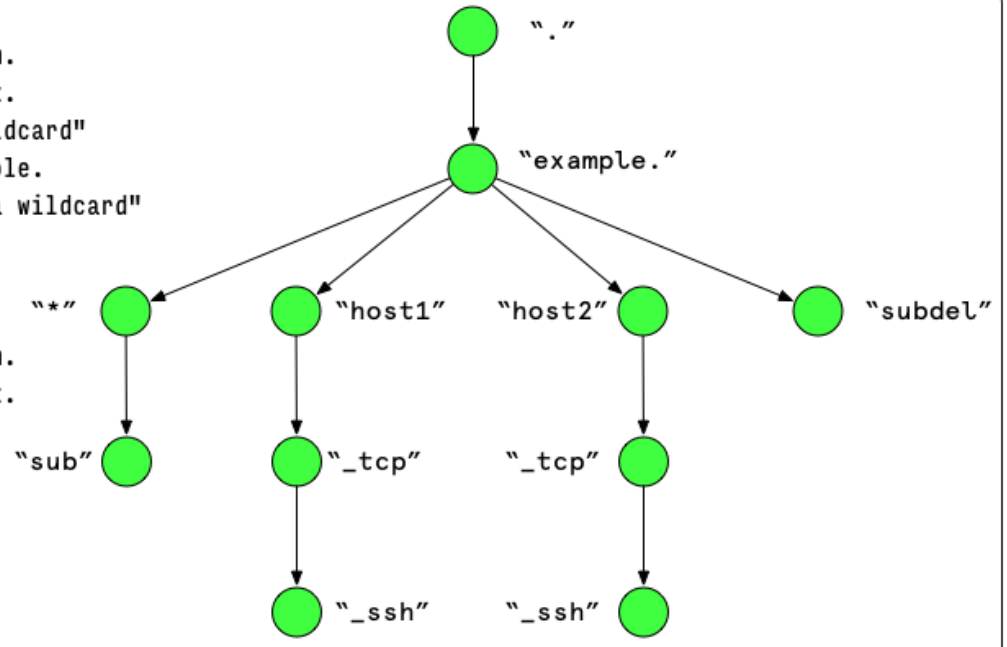


Query: _telnet._tcp.host1.example. ? IN SRV ?

Answer:

Answer 6

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

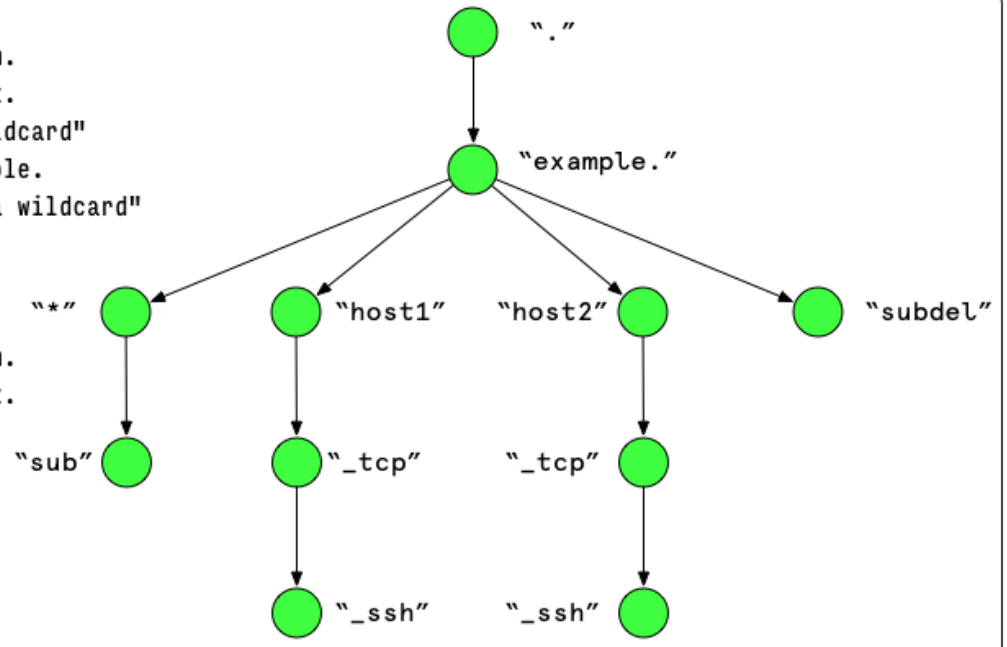


Query: _telnet._tcp.host1.example. ? IN SRV ?

Answer: NOERROR / NODATA (Answer = 0)

Question 7

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

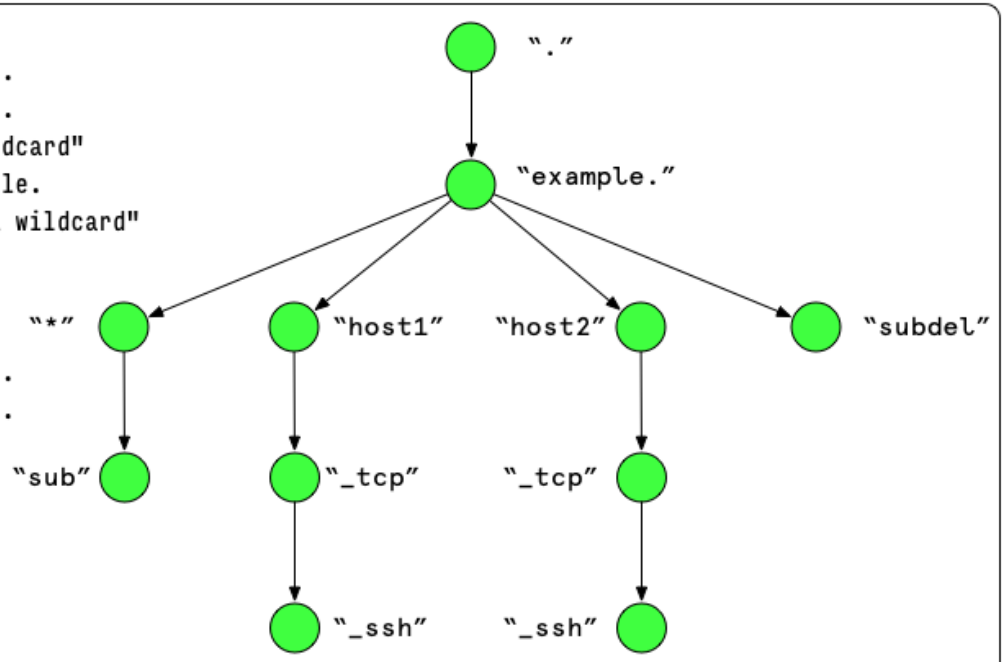


Query: host1.example. ? IN A ?

Answer:

Answer 7

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

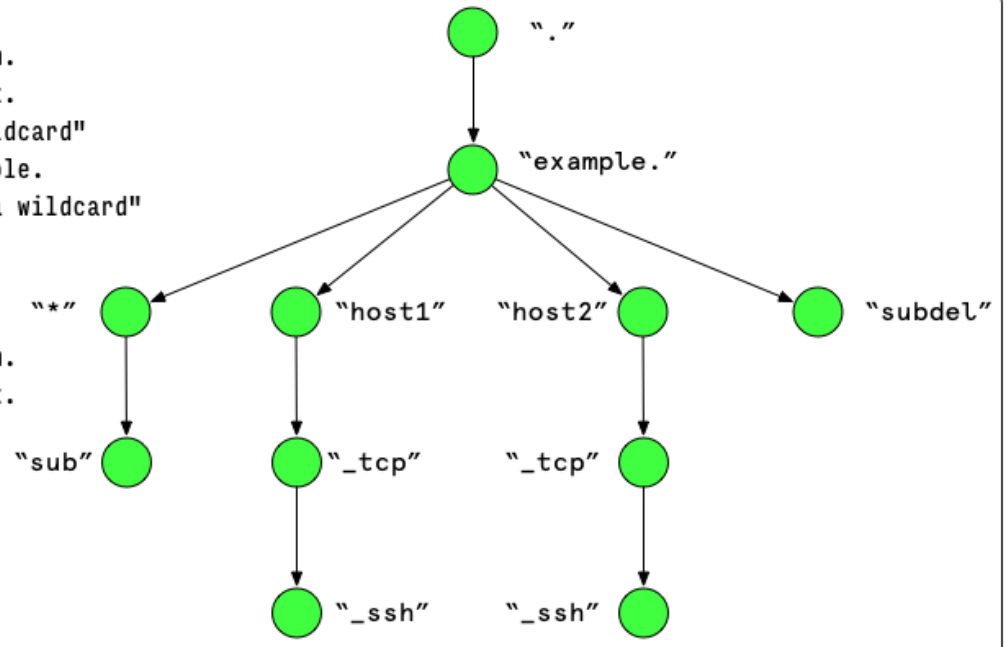



```
Query: host1.example. ? IN A ?
```

```
Answer: host1.example. 3600 IN A 192.0.2.1
```

Question 8

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

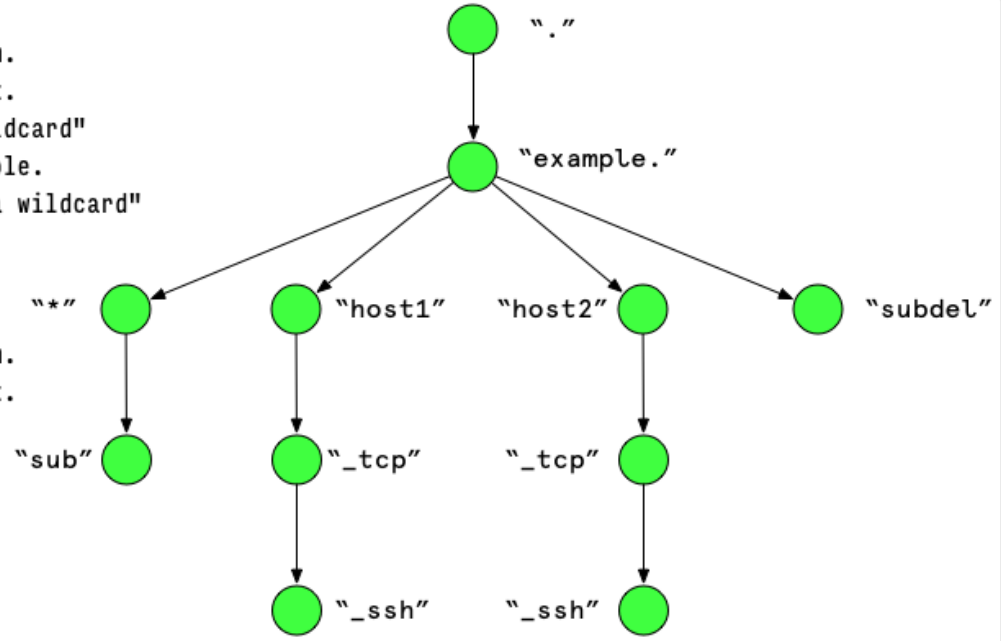


Query: host.subdel.example. ? IN A ?

Answer:

Answer 8

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

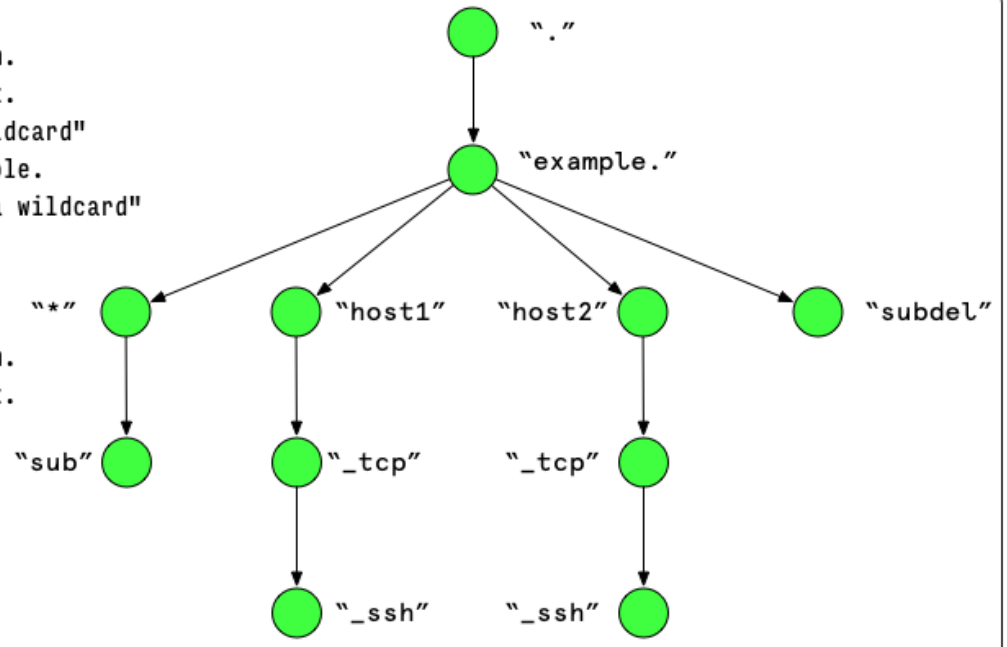



```
Query: host.subdel.example. ? IN A ?
```

```
Answer: referral to subdel.example
subdel.example.          3600     NS    ns.example.com.
subdel.example.          3600     NS    ns.example.net.
```

Question 9

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

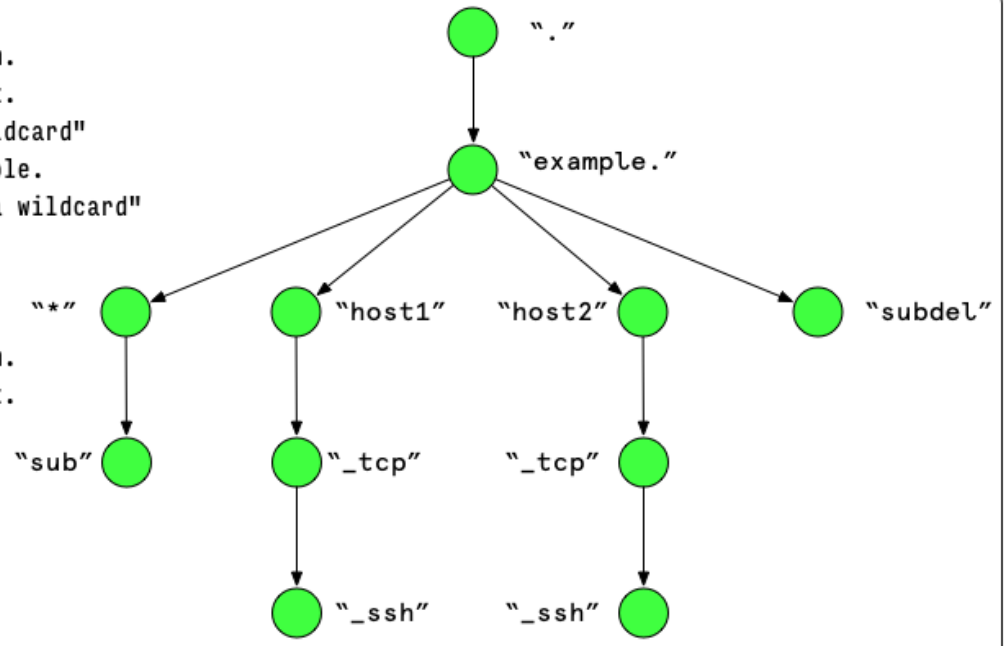


Query: ghost.*.example. ? IN MX ?

Answer:

Answer 9

```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```

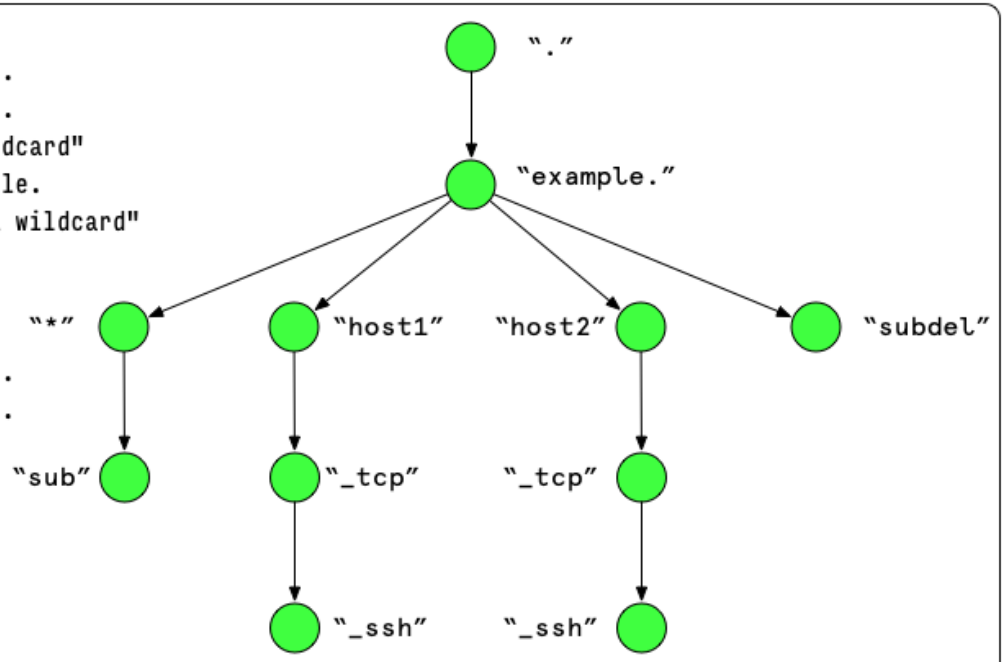


Query: ghost.*.example. ? IN MX ?

Answer: NOERROR / NODATA (Answer = 0)

Question 10

```
example.                    3600 IN  SOA   <SOA RDATA>
example.                    3600     NS    ns.example.com.
example.                    3600     NS    ns.example.net.
*.example.                  3600     TXT   "this is a wildcard"
*.example.                  3600     MX    10 host1.example.
sub.*.example.              3600     TXT   "this is not a wildcard"
host1.example.              3600     A     192.0.2.1
_ssh._tcp.host1.example.    3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.    3600     SRV   <SRV RDATA>
subdel.example.             3600     NS    ns.example.com.
subdel.example.             3600     NS    ns.example.net.
```

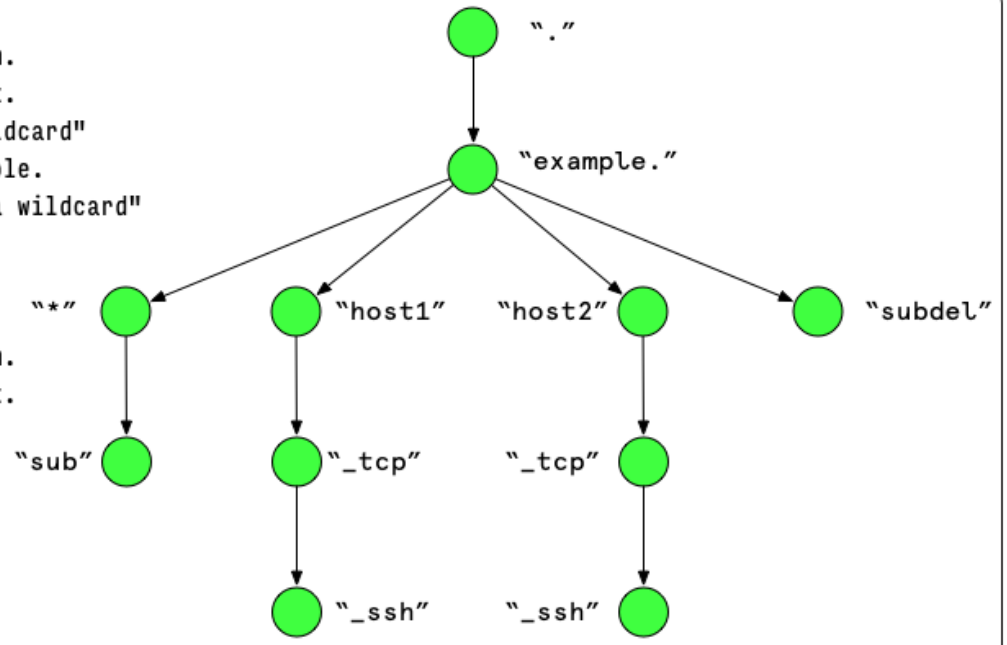


Query: *.example. ? IN TXT ?

Answer:

Answer 10

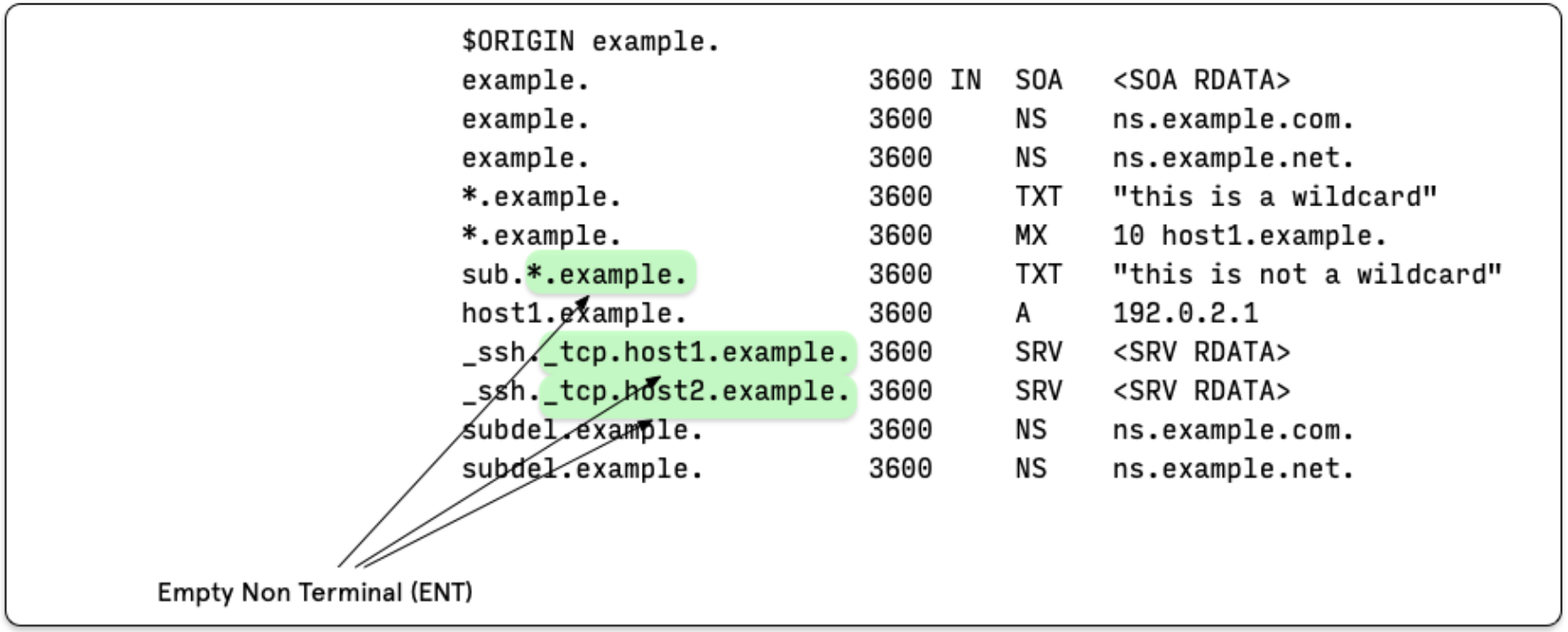
```
example.                  3600 IN  SOA   <SOA RDATA>
example.                  3600     NS    ns.example.com.
example.                  3600     NS    ns.example.net.
*.example.                3600     TXT   "this is a wildcard"
*.example.                3600     MX    10 host1.example.
sub.*.example.            3600     TXT   "this is not a wildcard"
host1.example.            3600     A     192.0.2.1
_ssh._tcp.host1.example.  3600     SRV   <SRV RDATA>
_ssh._tcp.host2.example.  3600     SRV   <SRV RDATA>
subdel.example.           3600     NS    ns.example.com.
subdel.example.           3600     NS    ns.example.net.
```



Query: *.example. ? IN TXT ?

Answer: *.example. 3600 TXT "this is a wildcard"

Empty Non Terminal



Closest Encloser and the Source of Synthesis

- The closest encloser is the node in the zone's tree of existing domain names that has the most labels matching the query name (consecutively, counting from the root label downward). Each match is a "label match" and the order of the labels is the same.
- The closest encloser is, by definition, an existing name in the zone. The closest encloser might be an empty non-terminal or even be a wildcard domain name itself. In no circumstances is the closest encloser to be used to synthesize records for the current query.

Closest Encloser and the Source of Synthesis

- The source of synthesis is defined in the context of a query process as that wildcard domain name immediately descending from the closest encloser, provided that this wildcard domain name exists. "Immediately descending" means that the source of synthesis has a name of the form:

```
*.<closest encloser>.
```

- A source of synthesis does not guarantee having a RRSet to use for synthesis. The source of synthesis could be an empty non-terminal.

Closest Encloser and the Source of Synthesis

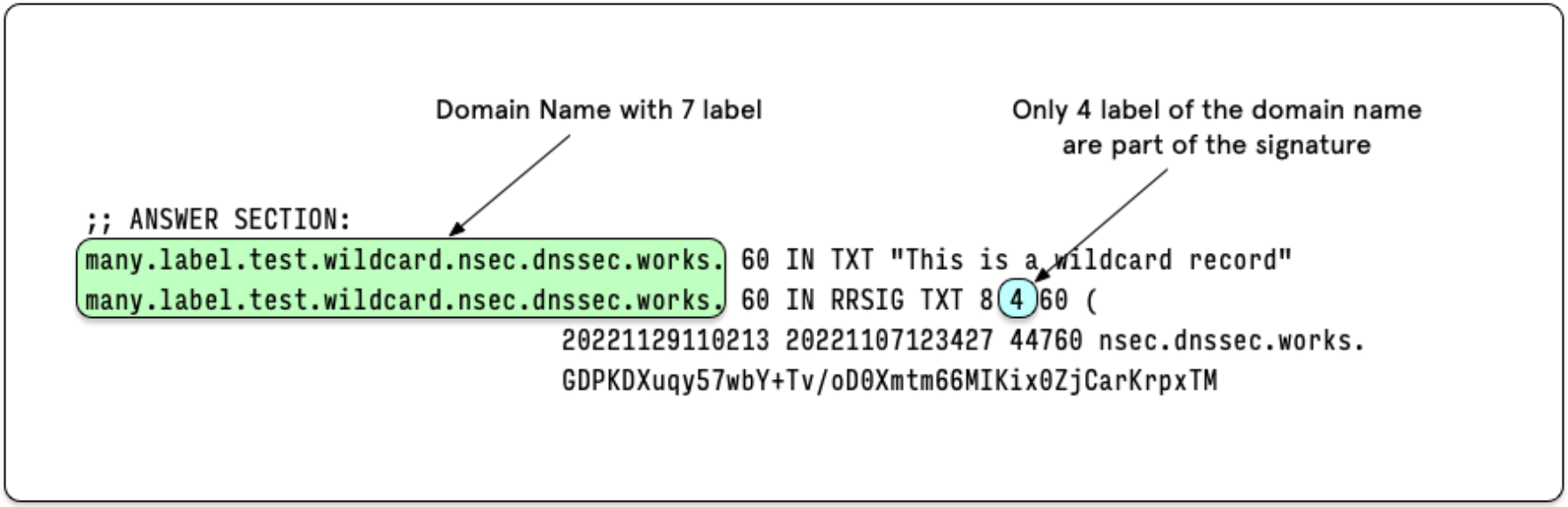
Domain Name in Query	closest encloser	source of synthesis
host3.example.	example.	*.example.
_telnet._tcp.host1.example.	_tcp.host1.example.	no source
_dns._udp.host2.example.	host2.example.	no source
_telnet._tcp.host3.example.	example.	*.example.
_chat._udp.host3.example.	example.	*.example.
foobar.*.example.	*.example.	no source

DNS Wildcards and DNSSEC

Label Count in RRSIG records

- DNSSEC signatures records (RRSIG) have a field for the number of labels in the domain name
  - This field defines how many labels from right to left in the domain name are significant and part of the name secured by the signature
  - If the number of the label count in the RRSIG record lower than the number of labels in the signature record domain name, the domain name have been synthesized from a wildcard

Label Count in RRSIG records



NSEC Wildcard Answer Response

- A DNSSEC signed positive answer from a wildcard DNS records **MUST** include the NSEC record in the *Authority Section* that proves the wildcard exists
- RFC 4035 "Protocol Modifications for the DNS Security Extensions" - 3.1.3. Including NSEC RRs in a Response

NSEC Wildcard Answer Response

```
% dig test.wildcard.nsec.dnssec.works in txt +dnssec +multi @9.9.9.9

; <<>> DiG 9.18.8 <<>> test.wildcard.nsec.dnssec.works in txt +dnssec +multi @9.9.9.9
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20006
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 512
;; QUESTION SECTION:
;test.wildcard.nsec.dnssec.works. IN TXT

;; ANSWER SECTION:
test.wildcard.nsec.dnssec.works. 60 IN TXT "This is a wildcard record"
test.wildcard.nsec.dnssec.works. 60 IN RRSIG TXT 8 4 60 (
                                 20221129110213 20221107123427 44760 nsec.dnssec.works.
                                 <signature-blob> )

;; AUTHORITY SECTION:
*.wildcard.nsec.dnssec.works. 3600 IN NSEC www.nsec.dnssec.works. TXT RRSIG NSEC
*.wildcard.nsec.dnssec.works. 3600 IN RRSIG NSEC 8 4 3600 (
                                 20221129110213 20221107123427 44760 nsec.dnssec.works.
                                 <signature-blog> )
;; Query time: 4077 msec
;; SERVER: 9.9.9.9#53(9.9.9.9) (UDP)
;; WHEN: Mon Nov 07 14:38:10 CET 2022
;; MSG SIZE  rcvd: 625
```

NSEC NXDOMAIN/NODATA Response

- The wildcard record will be treated *as-is* in NSEC *denial-of-existence* proofs
  - The wildcard record is **not** expanded in the NSEC data

NSEC NXDOMAIN/NODATA Response

```
% dig nsec.dnssec.works in txt +dnssec +multi

; <<>> DiG 9.18.8 <<>> nsec.dnssec.works in txt +dnssec +multi
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11205
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 4, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
;; QUESTION SECTION:
;nsec.dnssec.works.     IN TXT

;; AUTHORITY SECTION:
nsec.dnssec.works.      2658 IN NSEC *.wildcard.nsec.dnssec.works. A NS SOA AAAA RRSIG NSEC DNSKEY TYPE
nsec.dnssec.works.      2658 IN RRSIG NSEC 8 3 3600 (
                             20221129110213 20221107123427 44760 nsec.dnssec.works.
                             <signature-blob> )
```

NSEC Wildcard No Data Responses

- If a DNS request matches a wildcard name, but the querytype of the request is not found in the wildcard (*NODATA* response), the NSEC record proving the record types of the wildcard record is returned
  - In this case, the wildcard record also appears *as-is* (non-expanded) in the NSEC record inside the *Authority Section*

NSEC Wildcard No Data Responses

```
% dig 'zzz.wildcard.nsec.dnssec.works' in aaaa +dnssec +multi

; <<>> DiG 9.18.8 <<>> zzz.wildcard.nsec.dnssec.works in aaaa +dnssec +multi
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41693
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 4, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
;; QUESTION SECTION:
;zzz.wildcard.nsec.dnssec.works.        IN AAAA

;; AUTHORITY SECTION:
*.wildcard.nsec.dnssec.works. 2183 IN NSEC www.nsec.dnssec.works. TXT RRSIG NSEC
*.wildcard.nsec.dnssec.works. 2183 IN RRSIG NSEC 8 4 3600 (
                                20221129110213 20221107123427 44760 nsec.dnssec.works.
                                <signature-blob> )
```

NSEC3 Wildcard Answer Responses

- As with NSEC records, the NSEC3 record will appear in the *Authority Section* of positive answers
- RFC 5155 "7.2.6. Wildcard Answer Responses"

NSEC3 Wildcard Answer Responses

```
% dig 'zzz.wildcard.nsec3.dnssec.works' in TXT +dnssec +multi

; <<>> DiG 9.18.8 <<>> zzz.wildcard.nsec3.dnssec.works in TXT +dnssec +multi
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 3322
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
;; QUESTION SECTION:
;zzz.wildcard.nsec3.dnssec.works. IN TXT

;; ANSWER SECTION:
zzz.wildcard.nsec3.dnssec.works. 60 IN TXT "This is a NSEC3 wildcard record"
zzz.wildcard.nsec3.dnssec.works. 60 IN RRSIG TXT 8 4 60 (
                                 20221206013302 20221107123446 29910 nsec3.dnssec.works.
                                 <signature-blob> )

;; AUTHORITY SECTION:
H20BIEPJ7S0QQO7GPN4VFT13RP06N7QH.nsec3.dnssec.works. 3561 IN NSEC3 1 0 100 5CA1AB1E (
                                 56D1798FPEVCUN13SSNO3BFRC6QKQRSC
                                 A NS SOA AAAA RRSIG DNSKEY NSEC3PARAM
                                 TYPE65534 )
H20BIEPJ7S0QQO7GPN4VFT13RP06N7QH.nsec3.dnssec.works. 3561 IN RRSIG NSEC3 8 4 3600 (
                                 20221203215922 20221103215035 29910 nsec3.dnssec.works.
                                 <signature-blob> (

;; Query time: 38 msec
;; SERVER: 172.22.1.8#53(172.22.1.8) (UDP)
;; WHEN: Tue Nov 08 12:41:01 CET 2022
;; MSG SIZE  rcvd: 706
```

NSEC3 Wildcard Negative Responses

- Negative responses from zones signed with NSEC3 do require more information
  - This can result in large answers, that might trigger fragmentation or TCP transport
  - From RFC 5155 "NSEC3" https://www.rfc-editor.org/rfc/rfc5155#section-7.2.5

```
7.2.5.  Wildcard No Data Responses

   If there is a wildcard match for QNAME, but QTYPE is not present at
   that name, the response MUST include a closest encloser proof for
   QNAME and MUST include the NSEC3 RR that matches the wildcard.  This
   combination proves both that QNAME itself does not exist and that a
   wildcard that matches QNAME does exist.  Note that the closest
   encloser to QNAME MUST be the immediate ancestor of the wildcard RR
   (if this is not the case, then something has gone wrong).
```

NSEC3 Wildcard Negative Responses

```
% dig 'zzz.wildcard.nsec3.dnssec.works' in aaaa +dnssec +multi
;; Truncated, retrying in TCP mode.

; <<>> DiG 9.18.8 <<>> zzz.wildcard.nsec3.dnssec.works in aaaa +dnssec +multi
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 25574
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 8, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
;; QUESTION SECTION:
;zzz.wildcard.nsec3.dnssec.works. IN AAAA

;; AUTHORITY SECTION:
C425NI9NK7KNHSRK71A58UHFTIE2TS8G.nsec3.dnssec.works. 3148 IN NSEC3 1 0 100 5CA1AB1E (
                                EJ5B3VVKGMVI1GPV60K2JBOGG3UJK69O )
C425NI9NK7KNHSRK71A58UHFTIE2TS8G.nsec3.dnssec.works. 3148 IN RRSIG NSEC3 8 4 3600 (
                                20221206013302 20221107123446 29910 nsec3.dnssec.works.
                                <signature-blob> )
H20BIEPJ7S0QQO7GPN4VFT13RP06N7QH.nsec3.dnssec.works. 3148 IN NSEC3 1 0 100 5CA1AB1E (
                                56D1798FPEVCUN13SSNO3BFRC6QKQRSC
                                A NS SOA AAAA RRSIG DNSKEY NSEC3PARAM
                                TYPE65534 )
H20BIEPJ7S0QQO7GPN4VFT13RP06N7QH.nsec3.dnssec.works. 3148 IN RRSIG NSEC3 8 4 3600 (
                                20221203215922 20221103215035 29910 nsec3.dnssec.works.
                                <signature-blob> )
5FJNKNM4L9KPRROD95RD2RB8H3PIJ2BK.nsec3.dnssec.works. 3148 IN NSEC3 1 0 100 5CA1AB1E (
                                678TABK89US4LOIL828DIS70UQG89TO1
                                TXT RRSIG )
5FJNKNM4L9KPRROD95RD2RB8H3PIJ2BK.nsec3.dnssec.works. 3148 IN RRSIG NSEC3 8 4 3600 (
                                20221206013302 20221107123446 29910 nsec3.dnssec.works.
                                <signature-blob> )
nsec3.dnssec.works.     3148 IN SOA ns2.myinfrastructure.org. hostmaster.strotmann.de. (
                                472        ; serial
                                86400      ; refresh (1 day)
                                7200       ; retry (2 hours)
                                3542400    ; expire (5 weeks 6 days)
                                3600       ; minimum (1 hour)
                                )
nsec3.dnssec.works.     3148 IN RRSIG SOA 8 3 3600 (
                                20221207133446 20221107123446 29910 nsec3.dnssec.works.
                                <signature-blob> )

;; Query time: 0 msec
;; SERVER: 172.22.1.8#53(172.22.1.8) (TCP)
;; WHEN: Tue Nov 08 12:47:54 CET 2022
;; MSG SIZE  rcvd: 1387
```

Other important DNS Wildcard rules

- DNSSEC does not allow wildcard NS records
- Wildcard `DNAME` records are not allowed (represents a threat to the coherency of the DNS)
- In `SRV` record, the full name is the owner-name. There cannot be a wildcard on the domain part of an `SRV` record (same for `TLSA`, `DKIM` …)
- Wildcard `DS` (*DNSSEC delegation signer*) records are ignored
- If a source of synthesis is an empty non-terminal, the answer will be `NOERROR` / `NODATA` (Answer = 0)

BIND 9 "check-wildcard"

- BIND 9 will check for the wildcard symbol `*` in non-leaf label in domain names such as `leaf.*.example.com.`
  - This use of a wildcard symbol is usually a mistake
  - BIND 9 will report a warning when the `*` appears in a non-leaf label

```
# named-checkzone example.com primary/example.com-zone
primary/example.com-zone:14:
  warning: ownername 'non.wildcard.*.record.example.com' contains an non-terminal wildcard
zone example.com/IN: loaded serial 2012082201
```

Wildcards - should we use it?

Wildcards - should we use it?

- Be careful with DNSSEC, NSEC3 and DNS wildcard records - monitor the answer sizes of queries from the wildcard records
- The Implementation of DNS wildcards inside DNS software is complex and error prone
  - With more features (RRL, RPZ, Serve-Stale etc) being build into the DNS, there is a higher risk that DNS wildcards will break
  - Often there are simpler solutions to wildcard records, such as `$GENERATE`
- Consider DNS Wildcards carefully - don't use them if there is a better alternative

Wildcard Alternatives

- Use `$GENERATE` in BIND 9 zonefiles to create larger numbers of uniform DNS resource records
- Use a DNS server that can dynamically create (and possibly sign) DNS responses (programatically)
  - BIND 9 with DLZ (Dynamical Loadable Zones) or DynDB https://bind9.readthedocs.io/en/v9_18_8/chapter6.html
  - PowerDNS Lua2 Backend https://doc.powerdns.com/authoritative/backends/lua2.html
  - Knot SynthRecords https://www.knot-dns.cz/docs/3.2/singlehtml/index.html#synthrecord-automatic-forward-reverse-records

Literature and Links

- BIND 9 Configuration option `check-wildcard`
  https://bind9.readthedocs.io/en/v9_18_8/reference.html?highlight=wildcard#boolean-options
- RFC 1034 https://www.rfc-editor.org/rfc/rfc1034
- RFC 6672 https://www.rfc-editor.org/rfc/rfc6672
- RFC 4592 https://www.rfc-editor.org/rfc/rfc4592
- RFC 4035 https://www.rfc-editor.org/rfc/rfc4035
- RFC 5155 https://www.rfc-editor.org/rfc/rfc5155

Upcoming ISC Webinars

- 22nd Nov 2022 - SVCB/HTTPS Records
- 15th Dec 2022 - Memory management in BIND 9.16/9.18

Questions and Answers