Stream: Internet Engineering Task Force (IETF)
RFC: 9210
BCP: 235
Updates: 1123, 1536
Category: Best Current Practice
Published: March 2022
ISSN: 2070-1721
Authors: J. Kristoff D. Wessels
DataPlane.org *Verisign*

RFC 9210
DNS Transport over TCP - Operational Requirements

Abstract

This document updates RFCs 1123 and 1536. This document requires the operational practice of permitting DNS messages to be carried over TCP on the Internet as a Best Current Practice. This operational requirement is aligned with the implementation requirements in RFC 7766. The use of TCP includes both DNS over unencrypted TCP as well as over an encrypted TLS session. The document also considers the consequences of this form of DNS communication and the potential operational issues that can arise when this Best Current Practice is not upheld.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

DNS messages are delivered using UDP or TCP communications. While most DNS transactions are carried over UDP, some operators have been led to believe that any DNS-over-TCP traffic is unwanted or unnecessary for general DNS operation. When DNS over TCP has been restricted, a variety of communication failures and debugging challenges often arise. As DNS and new naming system features have evolved, TCP as a transport has become increasingly important for the correct and safe operation of an Internet DNS. Reflecting modern usage, the DNS standards declare that support for TCP is a required part of the DNS implementation specifications [RFC7766]. This document is the equivalent of formal requirements for the operational community, encouraging system administrators, network engineers, and security staff to ensure DNS-over-TCP communications support is on par with DNS-over-UDP communications. It updates [RFC1123], Section 6.1.3.2 to clarify that all DNS resolvers and recursive servers **MUST** support and service both TCP and UDP queries and also updates [RFC1536] to remove the misconception that TCP is only useful for zone transfers.

1.1. Requirements Language

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. History of DNS over TCP

The curious state of disagreement between operational best practices and guidance for DNS transport protocols derives from conflicting messages operators have received from other operators, implementors, and even the IETF. Sometimes these mixed signals have been explicit; on other occasions, conflicting messages have been implicit. This section presents an interpretation of the storied and conflicting history that led to this document. This section is included for informational purposes only.

2.1. Uneven Transport Usage and Preference

In the original suite of DNS specifications, [RFC1034] and [RFC1035] clearly specify that DNS messages could be carried in either UDP or TCP, but they also state that there is a preference for UDP as the best transport for queries in the general case. As stated in [RFC1035]:

> While virtual circuits can be used for any DNS activity, datagrams are preferred for queries due to their lower overhead and better performance.

Another early, important, and influential document, [RFC1123], marks the preference for a transport protocol more explicitly:

> DNS resolvers and recursive servers **MUST** support UDP, and **SHOULD** support TCP, for sending (non-zone-transfer) queries.

and it further stipulates that:

> A name server **MAY** limit the resources it devotes to TCP queries, but it **SHOULD NOT** refuse to service a TCP query just because it would have succeeded with UDP.

Culminating in [RFC1536], DNS over TCP came to be associated primarily with the zone transfer mechanism, while most DNS queries and responses were seen as the dominion of UDP.

2.2. Waiting for Large Messages and Reliability

In the original specifications, the maximum DNS-over-UDP message size was enshrined at 512 bytes. However, even while [RFC1123] prefers UDP for non-zone transfer queries, it foresaw that DNS over TCP would become more popular in the future to overcome this limitation:

> [...] it is also clear that some new DNS record types defined in the future will contain information exceeding the 512 byte limit that applies to UDP, and hence will require TCP.

At least two new, widely anticipated developments were set to elevate the need for DNS-over-TCP transactions. The first was dynamic updates defined in [RFC2136], and the second was the set of extensions collectively known as "DNSSEC", whose operational considerations were originally given in [RFC2541]. The former suggests that

> ...requestors who require an accurate response code must use TCP.

while the latter warns that

> ... larger keys increase the size of the KEY and SIG RRs. This increases the chance of DNS UDP packet overflow and the possible necessity for using higher overhead TCP in responses.

Yet, defying some expectations, DNS over TCP remained little used in real traffic across the Internet in the late 1990s. Dynamic updates saw little deployment between autonomous networks. Around the time DNSSEC was first defined, another new feature helped solidify UDP transport dominance for message transactions.

2.3. EDNS(0)

In 1999, the IETF published the Extension Mechanisms for DNS (EDNS(0)) in [RFC2671] (which was obsoleted by [RFC6891] in 2013). That document standardized a way for communicating DNS nodes to perform rudimentary capabilities negotiation. One such capability written into the base specification and present in every EDNS(0)-compatible message is the value of the maximum UDP payload size the sender can support. This unsigned 16-bit field specifies, in bytes, the maximum (possibly fragmented) DNS message size a node is capable of receiving over UDP. In practice, typical values are a subset of the 512- to 4096-byte range. EDNS(0) became widely deployed over the next several years, and numerous surveys (see [CASTRO2010] and [NETALYZR]) have shown that many systems support larger UDP MTUs with EDNS(0).

The natural effect of EDNS(0) deployment meant DNS messages larger than 512 bytes would be less reliant on TCP than they might otherwise have been. While a non-negligible population of DNS systems lacked EDNS(0) or fell back to TCP when necessary, DNS clients still strongly prefer UDP to TCP. For example, as of 2014, DNS-over-TCP transactions remained a very small fraction of overall DNS traffic received by root name servers [VERISIGN].

2.4. Fragmentation and Truncation

Although EDNS(0) provides a way for endpoints to signal support for DNS messages exceeding 512 bytes, the realities of a diverse and inconsistently deployed Internet may result in some large messages being unable to reach their destination. Any IP datagram whose size exceeds the MTU of a link it transits will be fragmented and then reassembled by the receiving host. Unfortunately, it is not uncommon for middleboxes and firewalls to block IP fragments. If one or more fragments do not arrive, the application does not receive the message, and the request times out.

For IPv4-connected hosts, the MTU is often an Ethernet payload size of 1500 bytes. This means that the largest unfragmented UDP DNS message that can be sent over IPv4 is likely 1472 bytes, although tunnel encapsulation may reduce that maximum message size in some cases.

For IPv6, the situation is a little more complicated. First, IPv6 headers are 40 bytes (versus 20 without options in IPv4). Second, approximately one-third of DNS recursive resolvers use the minimum MTU of 1280 bytes [APNIC]. Third, fragmentation in IPv6 can only be done by the host originating the datagram. The need to fragment is conveyed in an ICMPv6 "Packet Too Big" message. The originating host indicates a fragmented datagram with IPv6 extension headers. Unfortunately, it is quite common for both ICMPv6 and IPv6 extension headers to be blocked by middleboxes. According to [HUSTON], some 35% of IPv6-capable recursive resolvers were unable to receive a fragmented IPv6 packet. When the originating host receives a signal that fragmentation is required, it is expected to populate its path MTU cache for that destination. The application will then retry the query after a timeout since the host does not generally retain copies of messages sent over UDP for potential retransmission.

The practical consequence of all this is that DNS requestors must be prepared to retry queries with different EDNS(0) maximum message size values. Administrators of [BIND] are likely to be familiar with seeing "success resolving ... after reducing the advertised EDNS(0) UDP packet size to 512 octets" messages in their system logs.

Often, reducing the EDNS(0) UDP packet size leads to a successful response. That is, the necessary data fits within the smaller message size. However, when the data does not fit, the server sets the truncated flag in its response, indicating the client should retry over TCP to receive the whole response. This is undesirable from the client's point of view because it adds more latency and is potentially undesirable from the server's point of view due to the increased resource requirements of TCP.

Note that a receiver is unable to differentiate between packets lost due to congestion and packets (fragments) intentionally dropped by firewalls or middleboxes. Over network paths with non-trivial amounts of packet loss, larger, fragmented DNS responses are more likely to never arrive and time out compared to smaller, unfragmented responses. Clients might be misled into retrying queries with different EDNS(0) UDP packet size values for the wrong reason.

The issues around fragmentation, truncation, and TCP are driving certain implementation and policy decisions in the DNS. Notably, Cloudflare implemented what it calls "DNSSEC black lies" [CLOUDFLARE] and uses Elliptic Curve Digital Signature Algorithms (ECDSAs) such that their signed responses fit easily in 512 bytes. The Key Signing Key (KSK) Rollover Design Team [DESIGNTEAM] spent a lot of time thinking and worrying about response sizes. There is growing sentiment in the DNSSEC community that RSA key sizes beyond 2048 bits are impractical and that critical infrastructure zones should transition to elliptic curve algorithms to keep response sizes manageable [ECDSA].

More recently, renewed security concerns about fragmented DNS messages (see [AVOID_FRAGS] and [FRAG_POISON]) are leading implementors to consider smaller responses and lower default EDNS(0) UDP payload size values for both queriers and responders [FLAGDAY2020].

2.5. "Only Zone Transfers Use TCP"

Today, the majority of the DNS community expects, or at least has a desire, to see DNS-over-TCP transactions occur without interference [FLAGDAY2020]. However, there has also been a long-held belief by some operators, particularly for security-related reasons, that DNS-over-TCP services should be purposely limited or not provided at all [CHES94] [DJBDNS]. A popular meme is that DNS over TCP is only ever used for zone transfers and is generally unnecessary otherwise, with filtering all DNS-over-TCP traffic even described as a best practice.

The position on restricting DNS over TCP had some justification given that historical implementations of DNS name servers provided very little in the way of TCP connection management (for example, see Section 6.1.2 of [RFC7766] for more details). However, modern standards and implementations are nearing parity with the more sophisticated TCP management techniques employed by, for example, HTTP(S) servers and load balancers.

2.6. Reuse, Pipelining, and Out-of-Order Processing

The idea that a TCP connection can support multiple transactions goes back as far as [RFC0883], which states: "Multiple messages may be sent over a virtual circuit." Although [RFC1035], which updates the former, omits this particular detail, it has been generally accepted that a TCP connection can be used for more than one query and response.

[RFC5966] clarifies that servers are not required to preserve the order of queries and responses over any transport. [RFC7766], which updates the former, further encourages query pipelining over TCP to achieve performance on par with UDP. A server that sends out-of-order responses to pipelined queries avoids head-of-line blocking when the response for a later query is ready before the response to an earlier query.

However, TCP can potentially suffer from a different head-of-line blocking problem due to packet loss. Since TCP itself enforces ordering, a single lost segment delays delivery of data in any following segments until the lost segment is retransmitted and successfully received.

3. DNS-over-TCP Requirements

An average increase in DNS message size (e.g., due to DNSSEC), the continued development of new DNS features (Appendix A), and a denial-of-service mitigation technique (Section 8) all show that DNS-over-TCP transactions are as important to the correct and safe operation of the Internet

DNS as ever, if not more so. Furthermore, there has been research that argues connection-oriented DNS transactions may provide security and privacy advantages over UDP transport [TDNS]. In fact, the standard for DNS over TLS [RFC7858] is just this sort of specification. Therefore, this document makes explicit that it is undesirable for network operators to artificially inhibit DNS-over-TCP transport.

Section 6.1.3.2 of [RFC1123] is updated: All DNS resolvers and servers **MUST** support and service both UDP and TCP queries.

- DNS servers (including forwarders) **MUST** support and service TCP for receiving queries so that clients can reliably receive responses that are larger than what either side considers too large for UDP.
- DNS clients **MUST** support TCP for sending queries so that they can retry truncated UDP responses as necessary.

Furthermore, the requirement in Section 6.1.3.2 of [RFC1123] around limiting the resources a server devotes to queries is hereby updated:

OLD:

> A name server **MAY** limit the resources it devotes to TCP queries, but it **SHOULD NOT** refuse to service a TCP query just because it would have succeeded with UDP.

NEW:

> A name server **MAY** limit the resources it devotes to queries, but it **MUST NOT** refuse to service a query just because it would have succeeded with another transport protocol.

Lastly, Section 1 of [RFC1536] is updated to eliminate the misconception that TCP is only useful for zone transfers:

OLD:

> DNS implements the classic request-response scheme of client-server interaction. UDP is, therefore, the chosen protocol for communication though TCP is used for zone transfers.

NEW:

> DNS implements the classic request-response scheme of client-server interaction.

The filtering of DNS over TCP is harmful in the general case. DNS resolver and server operators **MUST** support and provide DNS service over both UDP and TCP transports. Likewise, network operators **MUST** allow DNS service over both UDP and TCP transports. It is acknowledged that DNS-over-TCP service can pose operational challenges that are not present when running DNS over UDP alone, and vice versa. However, the potential damage incurred by prohibiting DNS-over-TCP service is more detrimental to the continued utility and success of the DNS than when its usage is allowed.

4. Network and System Considerations

This section describes measures that systems and applications can take to optimize performance over TCP and to protect themselves from TCP-based resource exhaustion and attacks.

4.1. Connection Establishment and Admission

Resolvers and other DNS clients should be aware that some servers might not be reachable over TCP. For this reason, clients **MAY** track and limit the number of TCP connections and connection attempts to a single server. Reachability problems can be caused by network elements close to the server, close to the client, or anywhere along the path between them. Mobile clients that cache connection failures **MAY** do so on a per-network basis or **MAY** clear such a cache upon change of network.

Additionally, DNS clients **MAY** enforce a short timeout on unestablished connections rather than rely on the host operating system's TCP connection timeout, which is often around 60-120 seconds (i.e., due to an initial retransmission timeout of 1 second, the exponential back-off rules of [RFC6298], and a limit of six retries as is the default in Linux).

The SYN flooding attack is a denial-of-service method affecting hosts that run TCP server processes [RFC4987]. This attack can be very effective if not mitigated. One of the most effective mitigation techniques is SYN cookies, described in Section 3.6 of [RFC4987], which allows the server to avoid allocating any state until the successful completion of the three-way handshake.

Services not intended for use by the public Internet, such as most recursive name servers, **SHOULD** be protected with access controls. Ideally, these controls are placed in the network, well before any unwanted TCP packets can reach the DNS server host or application. If this is not possible, the controls can be placed in the application itself. In some situations (e.g., attacks), it may be necessary to deploy access controls for DNS services that should otherwise be globally reachable. See also [RFC5358].

The FreeBSD and NetBSD operating systems have an "accept filter" feature ([accept_filter]) that postpones delivery of TCP connections to applications until a complete, valid request has been received. The dns_accf(9) filter ensures that a valid DNS message is received. If not, the bogus connection never reaches the application. The Linux TCP_DEFER_ACCEPT feature, while more limited in scope, can provide some of the same benefits as the BSD accept filter feature. These features are implemented as low-level socket options and are not activated automatically. If applications wish to use these features, they need to make specific calls to set the right options, and administrators may also need to configure the applications to appropriately use the features.

Per [RFC7766], applications and administrators are advised to remember that TCP **MAY** be used before sending any UDP queries. Networks and applications **MUST NOT** be configured to refuse TCP queries that were not preceded by a UDP query.

TCP Fast Open (TFO) [RFC7413] allows TCP clients to shorten the handshake for subsequent connections to the same server. TFO saves one round-trip time in the connection setup. DNS servers **SHOULD** enable TFO when possible. Furthermore, DNS servers clustered behind a single service address (e.g., anycast or load balancing) **SHOULD** either use the same TFO server key on all instances or disable TFO for all members of the cluster.

DNS clients **MAY** also enable TFO. At the time of this writing, it is not implemented or is disabled by default on some operating systems. [WIKIPEDIA_TFO] describes applications and operating systems that support TFO.

4.2. Connection Management

Since host memory for TCP state is a finite resource, DNS clients and servers **SHOULD** actively manage their connections. Applications that do not actively manage their connections can encounter resource exhaustion leading to denial of service. For DNS, as in other protocols, there is a trade-off between keeping connections open for potential future use and the need to free up resources for new connections that will arrive.

Operators of DNS server software **SHOULD** be aware that operating system and application vendors **MAY** impose a limit on the total number of established connections. These limits may be designed to protect against DDoS attacks or performance degradation. Operators **SHOULD** understand how to increase these limits if necessary and the consequences of doing so. Limits imposed by the application **SHOULD** be lower than limits imposed by the operating system so that the application can apply its own policy to connection management, such as closing the oldest idle connections first.

DNS server software **MAY** provide a configurable limit on the number of established connections per source IP address or subnet. This can be used to ensure that a single or small set of users cannot consume all TCP resources and deny service to other users. Note, however, that if this limit is enabled, it possibly limits client performance while leaving some TCP resources unutilized. Operators **SHOULD** be aware of these trade-offs and ensure this limit, if configured, is set appropriately based on the number and diversity of their users and whether users connect from unique IP addresses or through a shared Network Address Translator (NAT) [RFC3022].

DNS server software **SHOULD** provide a configurable timeout for idle TCP connections. This can be used to free up resources for new connections and to ensure that idle connections are eventually closed. At the same time, it possibly limits client performance while leaving some TCP resources unutilized. For very busy name servers, this might be set to a low value, such as a few seconds. For less busy servers, it might be set to a higher value, such as tens of seconds. DNS clients and servers **SHOULD** signal their timeout values using the edns-tcp-keepalive option [RFC7828].

DNS server software **MAY** provide a configurable limit on the number of transactions per TCP connection. This can help protect against unfair connection use (e.g., not releasing connection slots to other clients) and network evasion attacks.

Similarly, DNS server software **MAY** provide a configurable limit on the total duration of a TCP connection. This can help protect against unfair connection use, slow read attacks, and network evasion attacks.

Since clients may not be aware of server-imposed limits, clients utilizing TCP for DNS need to always be prepared to re-establish connections or otherwise retry outstanding queries.

4.3. Connection Termination

The TCP peer that initiates a connection close retains the socket in the TIME_WAIT state for some amount of time, possibly a few minutes. It is generally preferable for clients to initiate the close of a TCP connection so that busy servers do not accumulate many sockets in the TIME_WAIT state, which can cause performance problems or even denial of service. The edns-tcp-keepalive EDNS(0) option [RFC7828] can be used to encourage clients to close connections.

On systems where large numbers of sockets in TIME_WAIT are observed (as either a client or a server) and are affecting an application's performance, it may be tempting to tune local TCP parameters. For example, the Linux kernel has a "sysctl" parameter named net.ipv4.tcp_tw_reuse, which allows connections in the TIME_WAIT state to be reused in specific circumstances. Note, however, that this affects only outgoing (client) connections and has no impact on servers. In most cases, it is **NOT RECOMMENDED** to change parameters related to the TIME_WAIT state. It should only be done by those with detailed knowledge of both TCP and the affected application.

4.4. DNS over TLS

DNS messages may be sent over TLS to provide privacy between stubs and recursive resolvers. [RFC7858] is a Standards Track document describing how this works. Although DNS over TLS utilizes TCP port 853 instead of port 53, this document applies equally well to DNS over TLS. Note, however, that DNS over TLS is only defined between stubs and recursives at the time of this writing.

The use of TLS places even stronger operational burdens on DNS clients and servers. Cryptographic functions for authentication and encryption require additional processing. Unoptimized connection setup with TLS 1.3 [RFC8446] takes one additional round trip compared to TCP. Connection setup times can be reduced with TCP Fast Open, and TLS False Start [RFC7918] for TLS 1.2. TLS 1.3 session resumption does not reduce round-trip latency because no application profile for use of TLS 0-RTT data with DNS has been published at the time of this writing. However, TLS session resumption can reduce the number of cryptographic operations, and in TLS 1.2, session resumption does reduce the number of additional round trips from two to one.

4.5. Defaults and Recommended Limits

A survey of features and defaults was conducted for popular open-source DNS server implementations at the time of writing. This section documents those defaults and makes recommendations for configurable limits that can be used in the absence of any other information. Any recommended values in this document are only intended as a starting point for administrators that are unsure of what sorts of limits might be reasonable. Operators **SHOULD** use application-specific monitoring, system logs, and system monitoring tools to gauge whether their service is operating within or exceeding these limits and adjust accordingly.

Most open-source DNS server implementations provide a configurable limit on the total number of established connections. Default values range from 20 to 150. In most cases, where the majority of queries take place over UDP, 150 is a reasonable limit. For services or environments where most queries take place over TCP or TLS, 5000 is a more appropriate limit.

Only some open-source implementations provide a way to limit the number of connections per source IP address or subnet, but the default is to have no limit. For environments or situations where it may be necessary to enable this limit, 25 connections per source IP address is a reasonable starting point. The limit should be increased when aggregated by subnet or for services where most queries take place over TCP or TLS.

Most open-source implementations provide a configurable idle timeout on connections. Default values range from 2 to 30 seconds. In most cases, 10 seconds is a reasonable default for this limit. Longer timeouts improve connection reuse, but busy servers may need to use a lower limit.

Only some open-source implementations provide a way to limit the number of transactions per connection, but the default is to have no limit. This document does not offer advice on particular values for such a limit.

Only some open-source implementations provide a way to limit the duration of connection, but the default is to have no limit. This document does not offer advice on particular values for such a limit.

5. DNS-over-TCP Filtering Risks

Networks that filter DNS over TCP risk losing access to significant or important pieces of the DNS namespace. For a variety of reasons, a DNS answer may require a DNS-over-TCP query. This may include large message sizes, lack of EDNS(0) support, or DDoS mitigation techniques (including Response Rate Limiting [RRL]); additionally, perhaps some future capability that is as yet unforeseen will also demand TCP transport.

For example, [RFC7901] describes a latency-avoiding technique that sends extra data in DNS responses. This makes responses larger and potentially increases the effectiveness of DDoS reflection attacks. The specification mandates the use of TCP or DNS cookies [RFC7873].

Even if any or all particular answers have consistently been returned successfully with UDP in the past, this continued behavior cannot be guaranteed when DNS messages are exchanged between autonomous systems. Therefore, filtering of DNS over TCP is considered harmful and contrary to the safe and successful operation of the Internet. This section enumerates some of the known risks at the time of this writing when networks filter DNS over TCP.

5.1. Truncation, Retries, and Timeouts

Networks that filter DNS over TCP may inadvertently cause problems for third-party resolvers as experienced by [TOYAMA]. For example, a resolver receives queries for a moderately popular domain. The resolver forwards the queries to the domain's authoritative name servers, but those servers respond with the TC bit set. The resolver retries over TCP, but the authoritative server blocks DNS over TCP. The pending connections consume resources on the resolver until they time out. If the number and frequency of these truncated-and-then-blocked queries are sufficiently high, the resolver wastes valuable resources on queries that can never be answered. This condition is generally not easily or completely mitigated by the affected DNS resolver operator.

5.2. DNS Root Zone KSK Rollover

The plans for deploying a new root zone DNSSEC KSK highlighted a potential problem in retrieving the root zone key set [LEWIS]. During some phases of the KSK rollover process, root zone DNSKEY responses were larger than 1280 bytes, the IPv6 minimum MTU for links carrying IPv6 traffic [RFC8200]. There was some concern that any DNS server unable to receive large DNS messages over UDP, or any DNS message over TCP, would experience disruption while performing DNSSEC validation [KSK_ROLLOVER_ARCHIVES].

However, during the year-long postponement of the KSK rollover, there were no reported problems that could be attributed to the 1414 octet DNSKEY response when both the old and new keys were published in the zone. Additionally, there were no reported problems during the two-month period when the old key was published as revoked and the DNSKEY response was 1425 octets in size [ROLL_YOUR_ROOT].

6. Logging and Monitoring

Developers of applications that log or monitor DNS **SHOULD NOT** ignore TCP due to the perception that it is rarely used or is hard to process. Operators **SHOULD** ensure that their monitoring and logging applications properly capture DNS messages over TCP. Otherwise, attacks, exfiltration attempts, and normal traffic may go undetected.

DNS messages over TCP are in no way guaranteed to arrive in single segments. In fact, a clever attacker might attempt to hide certain messages by forcing them over very small TCP segments. Applications that capture network packets (e.g., with libpcap [libpcap]) **SHOULD** implement and perform full TCP stream reassembly and analyze the reassembled stream instead of the individual packets. Otherwise, they are vulnerable to network evasion attacks [phrack]. Furthermore, such applications need to protect themselves from resource exhaustion attacks by limiting the amount of memory allocated to tracking unacknowledged connection state data. dnscap [dnscap] is an open-source example of a DNS logging program that implements TCP stream reassembly.

Developers **SHOULD** also keep in mind connection reuse, query pipelining, and out-of-order responses when building and testing DNS monitoring applications.

As an alternative to packet capture, some DNS server software supports dnstap [dnstap] as an integrated monitoring protocol intended to facilitate wide-scale DNS monitoring.

7. IANA Considerations

This document has no IANA actions.

8. Security Considerations

This document, providing operational requirements, is the companion to the implementation requirements of DNS over TCP provided in [RFC7766]. The security considerations from [RFC7766] still apply.

Ironically, returning truncated DNS-over-UDP answers in order to induce a client query to switch to DNS over TCP has become a common response to source-address-spoofed, DNS denial-of-service attacks [RRL]. Historically, operators have been wary of TCP-based attacks, but in recent years, UDP-based flooding attacks have proven to be the most common protocol attack on the DNS. Nevertheless, a high rate of short-lived DNS transactions over TCP may pose challenges. In fact, [DAI21] details a class of IP fragmentation attacks on DNS transactions if the IP Identifier field (16 bits in IPv4 and 32 bits in IPv6) can be predicted and a system is coerced to fragment rather than retransmit messages. While many operators have provided DNS-over-TCP service for many years without duress, past experience is no guarantee of future success.

DNS over TCP is similar to many other Internet TCP services. TCP threats and many mitigation strategies have been well documented in a series of documents such as [RFC4953], [RFC4987], [RFC5927], and [RFC5961].

As mentioned in Section 6, applications that implement TCP stream reassembly need to limit the amount of memory allocated to connection tracking. A failure to do so could lead to a total failure of the logging or monitoring application. Imposition of resource limits creates a trade-off between allowing some stream reassembly to continue and allowing some evasion attacks to succeed.

This document recommends that DNS servers enable TFO when possible. [RFC7413] recommends that a pool of servers behind a load balancer with a shared server IP address also share the key used to generate Fast Open cookies to prevent inordinate fallback to the three-way handshake (3WHS). This guidance remains accurate but comes with a caveat: compromise of one server would reveal this group-shared key and allow for attacks involving the other servers in the pool by forging invalid Fast Open cookies.

9. Privacy Considerations

Since DNS over both UDP and TCP uses the same underlying message format, the use of one transport instead of the other does not change the privacy characteristics of the message content (i.e., the name being queried). A number of protocols have recently been developed to provide DNS privacy, including DNS over TLS [RFC7858], DNS over DTLS [RFC8094], DNS over HTTPS [RFC8484], with even more on the way.

Because TCP is somewhat more complex than UDP, some characteristics of a TCP conversation may enable DNS client fingerprinting and tracking that is not possible with UDP. For example, the choice of initial sequence numbers, window size, and options might be able to identify a particular TCP implementation or even individual hosts behind shared resources such as NATs.

10. References

10.1. Normative References

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, DOI 10.17487/RFC2181, July 1997, <https://www.rfc-editor.org/info/rfc2181>.

[RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <https://www.rfc-editor.org/info/rfc6891>.

[RFC7766] Dickinson, J., Dickinson, S., Bellis, R., Mankin, A., and D. Wessels, "DNS Transport over TCP - Implementation Requirements", RFC 7766, DOI 10.17487/RFC7766, March 2016, <https://www.rfc-editor.org/info/rfc7766>.

[RFC7828] Wouters, P., Abley, J., Dickinson, S., and R. Bellis, "The edns-tcp-keepalive EDNS0 Option", RFC 7828, DOI 10.17487/RFC7828, April 2016, <https://www.rfc-editor.org/info/rfc7828>.

[RFC7873] Eastlake 3rd, D. and M. Andrews, "Domain Name System (DNS) Cookies", RFC 7873, DOI 10.17487/RFC7873, May 2016, <https://www.rfc-editor.org/info/rfc7873>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

10.2. Informative References

[accept_filter] FreeBSD, "FreeBSD accept_filter(9)", May 2018, <https://www.freebsd.org/cgi/man.cgi?query=accept_filter>.

[APNIC] Huston, G., "DNS XL", October 2020, <https://labs.apnic.net/?p=1380>.

[AVOID_FRAGS] Fujiwara, K. and P. Vixie, "Fragmentation Avoidance in DNS", Work in Progress, Internet-Draft, draft-ietf-dnsop-avoid-fragmentation-06, 23 December 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-dnsop-avoid-fragmentation-06>.

[BIND] Internet Systems Consortium, "BIND 9", <https://www.isc.org/bind/>.

[CASTRO2010] Castro, S., Zhang, M., John, W., Wessels, D., and K. claffy, "Understanding and Preparing for DNS Evolution", DOI 10.1007/978-3-642-12365-8_1, April 2010, <https://doi.org/10.1007/978-3-642-12365-8_1>.

[CHES94] Cheswick, W. and S. Bellovin, "Firewalls and Internet Security: Repelling the Wily Hacker", First Edition, 1994.

[CLOUDFLARE] Grant, D., "Economical With The Truth: Making DNSSEC Answers Cheap", June 2016, <https://blog.cloudflare.com/black-lies/>.

[DAI21] Dai, T., Shulman, H., and M. Waidner, "DNS-over-TCP Considered Vulnerable", DOI 10.1145/3472305.3472884, July 2021, <https://doi.org/10.1145/3472305.3472884>.

[DESIGNTEAM] ICANN, "Root Zone KSK Rollover Plan", March 2016, <https://www.iana.org/reports/2016/root-ksk-rollover-design-20160307.pdf>.

[DJBDNS] Bernstein, D., "When are TCP queries sent?", November 2002, <https://cr.yp.to/djbdns/tcp.html#why>.

[dnscap] DNS-OARC, "DNSCAP", May 2018, <https://www.dns-oarc.net/tools/dnscap>.

[dnstap] "dnstap", <https://dnstap.info>.

[ECDSA] van Rijswijk-Deij, R., Sperotto, A., and A. Pras, "Making the Case for Elliptic Curves in DNSSEC", DOI 10.1145/2831347.2831350, October 2015, <https://dl.acm.org/doi/10.1145/2831347.2831350>.

[FLAGDAY2020] DNS Software and Service Providers, "DNS Flag Day 2020", October 2020, <https://dnsflagday.net/2020/>.

[FRAG_POISON] Herzberg, A. and H. Shulman, "Fragmentation Considered Poisonous", May 2012, <https://u.cs.biu.ac.il/~herzbea/security/13-03-frag.pdf>.

[HUSTON] Huston, G., "Dealing with IPv6 fragmentation in the DNS", August 2017, <https://blog.apnic.net/2017/08/22/dealing-ipv6-fragmentation-dns/>.

[KSK_ROLLOVER_ARCHIVES] ICANN, "KSK Rollover List Archives", January 2019, <https://mm.icann.org/pipermail/ksk-rollover/2019-January/date.html>.

[LEWIS] Lewis, E., "2017 DNSSEC KSK Rollover", RIPE 74, May 2017, <https://ripe74.ripe.net/presentations/25-RIPE74-lewis-submission.pdf>.

[libpcap] The Tcpdump Group, "Tcpdump and Libpcap", <https://www.tcpdump.org>.

[NETALYZR] Kreibich, C., Weaver, N., Nechaev, B., and V. Paxson, "Netalyzr: Illuminating The Edge Network", DOI 10.1145/1879141.1879173, November 2010, <https://doi.org/10.1145/1879141.1879173>.

[phrack] horizon, "Defeating Sniffers and Intrusion Detection Systems", Phrack Magazine, December 1998, <http://phrack.org/issues/54/10.html>.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

[RFC0883] Mockapetris, P., "Domain names: Implementation specification", RFC 883, DOI 10.17487/RFC0883, November 1983, <https://www.rfc-editor.org/info/rfc883>.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/info/rfc1034>.

[RFC1123] Braden, R., Ed., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, DOI 10.17487/RFC1123, October 1989, <https://www.rfc-editor.org/info/rfc1123>.

[RFC1536] Kumar, A., Postel, J., Neuman, C., Danzig, P., and S. Miller, "Common DNS Implementation Errors and Suggested Fixes", RFC 1536, DOI 10.17487/RFC1536, October 1993, <https://www.rfc-editor.org/info/rfc1536>.

[RFC1995] Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995, DOI 10.17487/RFC1995, August 1996, <https://www.rfc-editor.org/info/rfc1995>.

[RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <https://www.rfc-editor.org/info/rfc1996>.

[RFC2136] Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, DOI 10.17487/RFC2136, April 1997, <https://www.rfc-editor.org/info/rfc2136>.

[RFC2541] Eastlake 3rd, D., "DNS Security Operational Considerations", RFC 2541, DOI 10.17487/RFC2541, March 1999, <https://www.rfc-editor.org/info/rfc2541>.

[RFC2671] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, DOI 10.17487/RFC2671, August 1999, <https://www.rfc-editor.org/info/rfc2671>.

[RFC2694] Srisuresh, P., Tsirtsis, G., Akkiraju, P., and A. Heffernan, "DNS extensions to Network Address Translators (DNS_ALG)", RFC 2694, DOI 10.17487/RFC2694, September 1999, <https://www.rfc-editor.org/info/rfc2694>.

[RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, DOI 10.17487/RFC3022, January 2001, <https://www.rfc-editor.org/info/rfc3022>.

[RFC3225] Conrad, D., "Indicating Resolver Support of DNSSEC", RFC 3225, DOI 10.17487/RFC3225, December 2001, <https://www.rfc-editor.org/info/rfc3225>.

[RFC3226] Gudmundsson, O., "DNSSEC and IPv6 A6 aware server/resolver message size requirements", RFC 3226, DOI 10.17487/RFC3226, December 2001, <https://www.rfc-editor.org/info/rfc3226>.

[RFC4472] Durand, A., Ihren, J., and P. Savola, "Operational Considerations and Issues with IPv6 DNS", RFC 4472, DOI 10.17487/RFC4472, April 2006, <https://www.rfc-editor.org/info/rfc4472>.

[RFC4953] Touch, J., "Defending TCP Against Spoofing Attacks", RFC 4953, DOI 10.17487/RFC4953, July 2007, <https://www.rfc-editor.org/info/rfc4953>.

[RFC4987] Eddy, W., "TCP SYN Flooding Attacks and Common Mitigations", RFC 4987, DOI 10.17487/RFC4987, August 2007, <https://www.rfc-editor.org/info/rfc4987>.

[RFC5358] Damas, J. and F. Neves, "Preventing Use of Recursive Nameservers in Reflector Attacks", BCP 140, RFC 5358, DOI 10.17487/RFC5358, October 2008, <https://www.rfc-editor.org/info/rfc5358>.

[RFC5452] Hubert, A. and R. van Mook, "Measures for Making DNS More Resilient against Forged Answers", RFC 5452, DOI 10.17487/RFC5452, January 2009, <https://www.rfc-editor.org/info/rfc5452>.

[RFC5507] IAB, Faltstrom, P., Ed., Austein, R., Ed., and P. Koch, Ed., "Design Choices When Expanding the DNS", RFC 5507, DOI 10.17487/RFC5507, April 2009, <https://www.rfc-editor.org/info/rfc5507>.

[RFC5625] Bellis, R., "DNS Proxy Implementation Guidelines", BCP 152, RFC 5625, DOI 10.17487/RFC5625, August 2009, <https://www.rfc-editor.org/info/rfc5625>.

[RFC5927] Gont, F., "ICMP Attacks against TCP", RFC 5927, DOI 10.17487/RFC5927, July 2010, <https://www.rfc-editor.org/info/rfc5927>.

[RFC5936] Lewis, E. and A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, DOI 10.17487/RFC5936, June 2010, <https://www.rfc-editor.org/info/rfc5936>.

[RFC5961] Ramaiah, A., Stewart, R., and M. Dalal, "Improving TCP's Robustness to Blind In-Window Attacks", RFC 5961, DOI 10.17487/RFC5961, August 2010, <https://www.rfc-editor.org/info/rfc5961>.

[RFC5966] Bellis, R., "DNS Transport over TCP - Implementation Requirements", RFC 5966, DOI 10.17487/RFC5966, August 2010, <https://www.rfc-editor.org/info/rfc5966>.

[RFC6298] Paxson, V., Allman, M., Chu, J., and M. Sargent, "Computing TCP's Retransmission Timer", RFC 6298, DOI 10.17487/RFC6298, June 2011, <https://www.rfc-editor.org/info/rfc6298>.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <https://www.rfc-editor.org/info/rfc6762>.

[RFC6950] Peterson, J., Kolkman, O., Tschofenig, H., and B. Aboba, "Architectural Considerations on Application Features in the DNS", RFC 6950, DOI 10.17487/RFC6950, October 2013, <https://www.rfc-editor.org/info/rfc6950>.

[RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, DOI 10.17487/RFC7413, December 2014, <https://www.rfc-editor.org/info/rfc7413>.

[RFC7477] Hardaker, W., "Child-to-Parent Synchronization in DNS", RFC 7477, DOI 10.17487/RFC7477, March 2015, <https://www.rfc-editor.org/info/rfc7477>.

[RFC7534] Abley, J. and W. Sotomayor, "AS112 Nameserver Operations", RFC 7534, DOI 10.17487/RFC7534, May 2015, <https://www.rfc-editor.org/info/rfc7534>.

[RFC7720] Blanchet, M. and L-J. Liman, "DNS Root Name Service Protocol and Deployment Requirements", BCP 40, RFC 7720, DOI 10.17487/RFC7720, December 2015, <https://www.rfc-editor.org/info/rfc7720>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <https://www.rfc-editor.org/info/rfc7858>.

[RFC7901] Wouters, P., "CHAIN Query Requests in DNS", RFC 7901, DOI 10.17487/RFC7901, June 2016, <https://www.rfc-editor.org/info/rfc7901>.

[RFC7918] Langley, A., Modadugu, N., and B. Moeller, "Transport Layer Security (TLS) False Start", RFC 7918, DOI 10.17487/RFC7918, August 2016, <https://www.rfc-editor.org/info/rfc7918>.

[RFC8027] Hardaker, W., Gudmundsson, O., and S. Krishnaswamy, "DNSSEC Roadblock Avoidance", BCP 207, RFC 8027, DOI 10.17487/RFC8027, November 2016, <https://www.rfc-editor.org/info/rfc8027>.

[RFC8094] Reddy, T., Wing, D., and P. Patil, "DNS over Datagram Transport Layer Security (DTLS)", RFC 8094, DOI 10.17487/RFC8094, February 2017, <https://www.rfc-editor.org/info/rfc8094>.

[RFC8162] Hoffman, P. and J. Schlyter, "Using Secure DNS to Associate Certificates with Domain Names for S/MIME", RFC 8162, DOI 10.17487/RFC8162, May 2017, <https://www.rfc-editor.org/info/rfc8162>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[RFC8324] Klensin, J., "DNS Privacy, Authorization, Special Uses, Encoding, Characters, Matching, and Root Structure: Time for Another Look?", RFC 8324, DOI 10.17487/RFC8324, February 2018, <https://www.rfc-editor.org/info/rfc8324>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8467] Mayrhofer, A., "Padding Policies for Extension Mechanisms for DNS (EDNS(0))", RFC 8467, DOI 10.17487/RFC8467, October 2018, <https://www.rfc-editor.org/info/rfc8467>.

[RFC8482] Abley, J., Gudmundsson, O., Majkowski, M., and E. Hunt, "Providing Minimal-Sized Responses to DNS Queries That Have QTYPE=ANY", RFC 8482, DOI 10.17487/RFC8482, January 2019, <https://www.rfc-editor.org/info/rfc8482>.

[RFC8483] Song, L., Ed., Liu, D., Vixie, P., Kato, A., and S. Kerr, "Yeti DNS Testbed", RFC 8483, DOI 10.17487/RFC8483, October 2018, <https://www.rfc-editor.org/info/rfc8483>.

[RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <https://www.rfc-editor.org/info/rfc8484>.

[RFC8490] Bellis, R., Cheshire, S., Dickinson, J., Dickinson, S., Lemon, T., and T. Pusateri, "DNS Stateful Operations", RFC 8490, DOI 10.17487/RFC8490, March 2019, <https://www.rfc-editor.org/info/rfc8490>.

[RFC8501] Howard, L., "Reverse DNS in IPv6 for Internet Service Providers", RFC 8501, DOI 10.17487/RFC8501, November 2018, <https://www.rfc-editor.org/info/rfc8501>.

[RFC8806] Kumari, W. and P. Hoffman, "Running a Root Server Local to a Resolver", RFC 8806, DOI 10.17487/RFC8806, June 2020, <https://www.rfc-editor.org/info/rfc8806>.

[RFC8906] Andrews, M. and R. Bellis, "A Common Operational Problem in DNS Servers: Failure to Communicate", BCP 231, RFC 8906, DOI 10.17487/RFC8906, September 2020, <https://www.rfc-editor.org/info/rfc8906>.

[RFC8932] Dickinson, S., Overeinder, B., van Rijswijk-Deij, R., and A. Mankin, "Recommendations for DNS Privacy Service Operators", BCP 232, RFC 8932, DOI 10.17487/RFC8932, October 2020, <https://www.rfc-editor.org/info/rfc8932>.

[RFC8945] Dupont, F., Morris, S., Vixie, P., Eastlake 3rd, D., Gudmundsson, O., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", STD 93, RFC 8945, DOI 10.17487/RFC8945, November 2020, <https://www.rfc-editor.org/info/rfc8945>.

[ROLL_YOUR_ROOT] Müller, M., Thomas, M., Wessels, D., Hardaker, W., Chung, T., Toorop, W., and R. van Rijswijk-Deij, "Roll, Roll, Roll Your Root: A Comprehensive Analysis of the First Ever DNSSEC Root KSK Rollover", DOI 10.1145/3355369.3355570, October 2019, <https://dl.acm.org/doi/10.1145/3355369.3355570>.

[RRL] Vixie, P. and V. Schryver, "DNS Response Rate Limiting (DNS RRL)", ISC-TN-2012-1-Draft1, April 2012.

[TDNS] Zhu, L., Heidemann, J., Wessels, D., Mankin, A., and N. Somaiya, "Connection-Oriented DNS to Improve Privacy and Security", DOI 10.1109/SP.2015.18, May 2015, <https://doi.org/10.1109/SP.2015.18>.

[TOYAMA] Toyama, K., Ishibashi, K., Toyono, T., Ishino, M., Yoshimura, C., and K. Fujiwara, "DNS Anomalies and Their Impacts on DNS Cache Servers", NANOG 32, October 2004.

[VERISIGN] Thomas, M. and D. Wessels, "An Analysis of TCP Traffic in Root Server DITL Data", DNS-OARC 2014 Fall Workshop, October 2014.

[WIKIPEDIA_TFO] Wikipedia, "TCP Fast Open", February 2022, <https://en.wikipedia.org/w/index.php?title=TCP_Fast_Open&oldid=1071397204>.

Appendix A. Standards Related to DNS Transport over TCP

This section enumerates all known RFCs with a status of Internet Standard, Draft Standard, Proposed Standard, Informational, Best Current Practice, or Experimental that either implicitly or explicitly make assumptions or statements about the use of TCP as a transport for the DNS germane to this document.

A.1. IETF RFC 1035 - DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION

The Internet Standard [RFC1035] is the base DNS specification that explicitly defines support for DNS over TCP.

A.2. IETF RFC 1536 - Common DNS Implementation Errors and Suggested Fixes

The Informational document [RFC1536] states that UDP is "the chosen protocol for communication though TCP is used for zone transfers." That statement should now be considered in its historical context and is no longer a proper reflection of modern expectations.

A.3. IETF RFC 1995 - Incremental Zone Transfer in DNS

The Proposed Standard [RFC1995] documents the use of TCP as the fallback transport when Incremental Zone Transfer (IXFR) responses do not fit into a single UDP response. As with Authoritative Transfer (AXFR), IXFR messages are typically delivered over TCP by default in practice.

A.4. IETF RFC 1996 - A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)

The Proposed Standard [RFC1996] suggests that a primary server may decide to issue NOTIFY messages over TCP. In practice, NOTIFY messages are generally sent over UDP, but this specification leaves open the possibility that the choice of transport protocol is up to the primary server; therefore, a secondary server ought to be able to operate over both UDP and TCP.

A.5. IETF RFC 2181 - Clarifications to the DNS Specification

The Proposed Standard [RFC2181] includes clarifying text on how a client should react to the TC bit set on responses. It is advised that the response be discarded and the query resent using TCP.

A.6. IETF RFC 2694 - DNS extensions to Network Address Translators (DNS_ALG)

The Informational document [RFC2694] enumerates considerations for NAT devices to properly handle DNS traffic. This document is noteworthy in its suggestion that "[t]ypically, TCP is used for AXFR requests," as further evidence that helps explain why DNS over TCP may have often been treated very differently than DNS over UDP in operational networks.

A.7. IETF RFC 3225 - Indicating Resolver Support of DNSSEC

The Proposed Standard [RFC3225] makes statements indicating that DNS over TCP is "detrimental" as a result of increased traffic, latency, and server load. This document is a companion to the next document in the RFC Series expressing the requirement for EDNS(0) support for DNSSEC.

A.8. IETF RFC 3226 - DNSSEC and IPv6 A6 aware server/resolver message size requirements

Although updated by later DNSSEC RFCs, the Proposed Standard [RFC3226] strongly argues in favor of UDP messages instead of TCP, largely for performance reasons. The document declares EDNS(0) a requirement for DNSSEC servers and advocates that packet fragmentation may be preferable to TCP in certain situations.

A.9. IETF RFC 4472 - Operational Considerations and Issues with IPv6 DNS

The Informational document [RFC4472] notes that IPv6 data may increase DNS responses beyond what would fit in a UDP message. What is particularly noteworthy, but perhaps less common today than when this document was written, is that it refers to implementations that truncate data without setting the TC bit to encourage the client to resend the query using TCP.

A.10. IETF RFC 5452 - Measures for Making DNS More Resilient against Forged Answers

The Proposed Standard [RFC5452] arose as public DNS systems began to experience widespread abuse from spoofed queries, resulting in amplification and reflection attacks against unwitting victims. One of the leading justifications for supporting DNS over TCP to thwart these attacks is briefly described in Section 9.3 of [RFC5452] ("Spoof Detection and Countermeasure").

A.11. IETF RFC 5507 - Design Choices When Expanding the DNS

The Informational document [RFC5507] was largely an attempt to dissuade new DNS data types from overloading the TXT resource record type. In so doing, it summarizes the conventional wisdom of DNS design and implementation practices. The authors suggest TCP overhead and stateful properties pose challenges compared to UDP and imply that UDP is generally preferred for performance and robustness.

A.12. IETF RFC 5625 - DNS Proxy Implementation Guidelines

The Best Current Practice document [RFC5625] provides DNS proxy implementation guidance including the mandate that a proxy "**MUST** [...] be prepared to receive and forward queries over TCP" even though it suggests that, historically, TCP transport has not been strictly mandatory in stub resolvers or recursive servers.

A.13. IETF RFC 5936 - DNS Zone Transfer Protocol (AXFR)

The Proposed Standard [RFC5936] provides a detailed specification for the zone transfer protocol, as originally outlined in the early DNS standards. AXFR operation is limited to TCP and not specified for UDP. This document discusses TCP usage at length.

A.14. IETF RFC 7534 - AS112 Nameserver Operations

The Informational document [RFC7534] enumerates the requirements for operation of AS112 project DNS servers. New AS112 nodes are tested for their ability to provide service on both UDP and TCP transports, with the implication that TCP service is an expected part of normal operations.

A.15. IETF RFC 6762 - Multicast DNS

In the Proposed Standard [RFC6762], the TC bit is deemed to have essentially the same meaning as described in the original DNS specifications. That is, if a response with the TC bit set is received, "[...] the querier **SHOULD** reissue its query using TCP in order to receive the larger response."

A.16. IETF RFC 6891 - Extension Mechanisms for DNS (EDNS(0))

The Internet Standard [RFC6891] helped slow the use of and need for DNS-over-TCP messages. This document highlights concerns over server load and scalability in widespread use of DNS over TCP.

A.17. IAB RFC 6950 - Architectural Considerations on Application Features in the DNS

The Informational document [RFC6950] draws attention to large data in the DNS. TCP is referenced in the context as a common fallback mechanism and counter to some spoofing attacks.

A.18. IETF RFC 7477 - Child-to-Parent Synchronization in DNS

The Proposed Standard [RFC7477] specifies an RRType and a protocol to signal and synchronize NS, A, and AAAA resource record changes from a child-to-parent zone. Since this protocol may require multiple requests and responses, it recommends utilizing DNS over TCP to ensure the conversation takes place between a consistent pair of end nodes.

A.19. IETF RFC 7720 - DNS Root Name Service Protocol and Deployment Requirements

The Best Current Practice document [RFC7720] declares that root name service "**MUST** support UDP [RFC0768] and TCP [RFC0793] transport of DNS queries and responses."

A.20. IETF RFC 7766 - DNS Transport over TCP - Implementation Requirements

The Proposed Standard [RFC7766] instructs DNS implementors to provide support for carrying DNS-over-TCP messages in their software and might be considered the direct ancestor of this operational requirements document. The implementation requirements document codifies mandatory support for DNS-over-TCP in compliant DNS software but makes no recommendations to operators, which we seek to address here.

A.21. IETF RFC 7828 - The edns-tcp-keepalive EDNS(0) Option

The Proposed Standard [RFC7828] defines an EDNS(0) option to negotiate an idle timeout value for long-lived DNS-over-TCP connections. Consequently, this document is only applicable and relevant to DNS-over-TCP sessions and between implementations that support this option.

A.22. IETF RFC 7858 - Specification for DNS over Transport Layer Security (TLS)

The Proposed Standard [RFC7858] defines a method for putting DNS messages into a TCP-based encrypted channel using TLS. This specification is noteworthy for explicitly targeting the stub-to-recursive traffic but does not preclude its application from recursive-to-authoritative traffic.

A.23. IETF RFC 7873 - Domain Name System (DNS) Cookies

The Proposed Standard [RFC7873] describes an EDNS(0) option to provide additional protection against query and answer forgery. This specification mentions DNS over TCP as an alternative mechanism when DNS cookies are not available. The specification does make mention of DNS-over-TCP processing in two specific situations. In one, when a server receives only a client cookie in a request, the server should consider whether the request arrived over TCP, and if so, it should consider accepting TCP as sufficient to authenticate the request and respond accordingly. In another, when a client receives a BADCOOKIE reply using a fresh server cookie, the client should retry using TCP as the transport.

A.24. IETF RFC 7901 - CHAIN Query Requests in DNS

The Experimental specification [RFC7901] describes an EDNS(0) option that can be used by a security-aware validating resolver to request and obtain a complete DNSSEC validation path for any single query. This document requires the use of DNS over TCP or a transport mechanism verified by a source IP address such as EDNS-COOKIE [RFC7873].

A.25. IETF RFC 8027 - DNSSEC Roadblock Avoidance

The Best Current Practice document [RFC8027] details observed problems with DNSSEC deployment and mitigation techniques. Network traffic blocking and restrictions, including DNS-over-TCP messages, are highlighted as one reason for DNSSEC deployment issues. While this document suggests these sorts of problems are due to "non-compliant infrastructure", the scope of the document is limited to detection and mitigation techniques to avoid so-called DNSSEC roadblocks.

A.26. IETF RFC 8094 - DNS over Datagram Transport Layer Security (DTLS)

The Experimental specification [RFC8094] details a protocol that uses a datagram transport (UDP) but stipulates that "DNS clients and servers that implement DNS over DTLS **MUST** also implement DNS over TLS in order to provide privacy for clients that desire Strict Privacy [...]." This requirement implies DNS over TCP must be supported in case the message size is larger than the path MTU.

A.27. IETF RFC 8162 - Using Secure DNS to Associate Certificates with Domain Names for S/MIME

The Experimental specification [RFC8162] describes a technique to authenticate user X.509 certificates in an S/MIME system via the DNS. The document points out that the new experimental resource record types are expected to carry large payloads, resulting in the suggestion that "applications **SHOULD** use TCP -- not UDP -- to perform queries for the SMIMEA resource record."

A.28. IETF RFC 8324 - DNS Privacy, Authorization, Special Uses, Encoding, Characters, Matching, and Root Structure: Time for Another Look?

The Informational document [RFC8324] briefly discusses the common role and challenges of DNS over TCP throughout the history of DNS.

A.29. IETF RFC 8467 - Padding Policies for Extension Mechanisms for DNS (EDNS(0))

The Experimental document [RFC8467] reminds implementors to consider the underlying transport protocol (e.g., TCP) when calculating the padding length when artificially increasing the DNS message size with an EDNS(0) padding option.

A.30. IETF RFC 8482 - Providing Minimal-Sized Responses to DNS Queries That Have QTYPE=ANY

The Proposed Standard [RFC8482] describes alternative ways that DNS servers can respond to queries of type ANY, which are sometimes used to provide amplification in DDoS attacks. The specification notes that responders may behave differently, depending on the transport. For example, minimal-sized responses may be used over UDP transport, while full responses may be given over TCP.

A.31. IETF RFC 8483 - Yeti DNS Testbed

The Informational document [RFC8483] describes a testbed environment that highlights some DNS-over-TCP behaviors, including issues involving packet fragmentation and operational requirements for TCP stream assembly in order to conduct DNS measurement and analysis.

A.32. IETF RFC 8484 - DNS Queries over HTTPS (DoH)

The Proposed Standard [RFC8484] defines a protocol for sending DNS queries and responses over HTTPS. This specification assumes TLS and TCP for the underlying security and transport layers, respectively. Self-described as a technique that more closely resembles a tunneling mechanism, DoH nevertheless likely implies DNS over TCP in some sense, if not directly.

A.33. IETF RFC 8490 - DNS Stateful Operations

The Proposed Standard [RFC8490] updates the base protocol specification with a new OPCODE to help manage stateful operations in persistent sessions, such as those that might be used by DNS over TCP.

A.34. IETF RFC 8501 - Reverse DNS in IPv6 for Internet Service Providers

The Informational document [RFC8501] identifies potential operational challenges with dynamic DNS, including denial-of-service threats. The document suggests TCP may provide some advantages but that updating hosts would need to be explicitly configured to use TCP instead of UDP.

A.35. IETF RFC 8806 - Running a Root Server Local to a Resolver

The Informational document [RFC8806] describes how to obtain and operate a local copy of the root zone with examples showing how to pull from authoritative sources using a DNS-over-TCP zone transfer.

A.36. IETF RFC 8906 - A Common Operational Problem in DNS Servers: Failure to Communicate

The Best Current Practice document [RFC8906] discusses a number of DNS operational failure scenarios and how to avoid them. This includes discussions involving DNS-over-TCP queries, EDNS over TCP, and a testing methodology that includes a section on verifying DNS-over-TCP functionality.

A.37. IETF RFC 8932 - Recommendations for DNS Privacy Service Operators

The Best Current Practice document [RFC8932] presents privacy considerations to DNS privacy service operators. These mechanisms sometimes include the use of TCP and are therefore susceptible to information leakage such as TCP-based fingerprinting. This document also references an earlier draft version of this document.

A.38. IETF RFC 8945 - Secret Key Transaction Authentication for DNS (TSIG)

The Internet Standard [RFC8945] recommends that a client use TCP if truncated TSIG messages are received.

Acknowledgments

This document was initially motivated by feedback from students who pointed out that they were hearing contradictory information about filtering DNS-over-TCP messages. Thanks in particular to a teaching colleague, JPL, who perhaps unknowingly encouraged the initial research into the differences between what the community has historically said and did. Thanks to all the NANOG 63 attendees who provided feedback for an early talk on this subject.

The following individuals provided an array of feedback to help improve this document: Joe Abley, Piet Barber, Sara Dickinson, Tony Finch, Bob Harold, Paul Hoffman, Geoff Huston, Tatuya Jinmei, Puneet Sood, and Richard Wilhelm. The authors are also indebted to the contributions stemming from discussion in the TCPM Working Group meeting at IETF 104. Any remaining errors or imperfections are the sole responsibility of the document authors.

Authors' Addresses

John Kristoff
DataPlane.org
Chicago, IL 60605
United States of America
Phone: +1 312 493 0305
Email: jtk@dataplane.org
URI: https://dataplane.org/jtk/

Duane Wessels
Verisign
12061 Bluemont Way
Reston, VA 20190
United States of America
Phone: +1 703 948 3200
Email: dwessels@verisign.com
URI: https://verisign.com