

DNSSEC Working Group  
INTERNET-DRAFT  
Expires: March 1999

Donald E. Eastlake, 3rd  
IBM  
September 1998

## Secret Key Establishment for DNS (TKEY RR)

-----

Donald E. Eastlake 3rd

### Status of This Document

This draft, file name [draft-ietf-dnssec-tkey-01.txt](#), is intended to become a Proposed Standard RFC. Distribution of this document is unlimited. Comments should be sent to the DNS security mailing list <dns-security@tis.com> or to the author.

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months. Internet-Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet-Drafts as reference material or to cite them other than as a ``working draft'' or ``work in progress.''

To view the entire list of current Internet-Drafts, please check the "1id-abstracts.txt" listing contained in the Internet-Drafts Shadow Directories on ftp.is.co.za (Africa), ftp.nordu.net (Northern Europe), ftp.nis.garr.it (Southern Europe), munnari.oz.au (Pacific Rim), ftp.ietf.org (US East Coast), or ftp.isi.edu (US West Coast).

INTERNET-DRAFT

The DNS TKEY RR

September 1998

## Abstract

[[draft-ietf-dnsind-tsig-\\*.txt](#)] provides a means of authenticating and securing Domain Name System (DNS) queries and responses using shared secret keys via the TSIG resource record (RR). However, it provides no mechanism for setting up such keys other than manual exchange. This document describes a TKEY RR that can be used in a number of different modes to establish shared secret keys between a DNS resolver and server.

[changes from last draft: add IANA considerations section, make time fields module 2\*\*32, minor edits, update author info, ...]

## Acknowledgments

The substantial comments and ideas of the following persons (listed in alphabetic order) have been incorporated herein and are gratefully acknowledged:

Olafur Gudmundsson <ogud@tis.com>

Stuart Kwan <skwan@microsoft.com>

INTERNET-DRAFT

The DNS TKEY RR

September 1998

## Table of Contents

Status of This Document.....	<a href="#">1</a>
Abstract.....	<a href="#">2</a>
Acknowledgments.....	<a href="#">2</a>
Table of Contents.....	<a href="#">3</a>
<a href="#">1</a> . Introduction.....	<a href="#">4</a>
<a href="#">1.1</a> General Principles.....	<a href="#">4</a>
<a href="#">1.2</a> Overview of Contents.....	<a href="#">5</a>
<a href="#">2</a> . The TKEY Resource Record.....	<a href="#">6</a>
<a href="#">3</a> . Exchange via Resolver Query.....	<a href="#">8</a>
<a href="#">3.1</a> Query for Server Assigned Keying.....	<a href="#">8</a>
<a href="#">3.2</a> Query for Diffie-Hellman Exchanged Keying.....	<a href="#">9</a>
<a href="#">3.3</a> Query for GSS-API Established.....	<a href="#">10</a>
<a href="#">4</a> . Spontaneous Server Inclusion.....	<a href="#">11</a>
<a href="#">4.1</a> Spontaneous Server Assigned Keying.....	<a href="#">11</a>
<a href="#">4.2</a> Spontaneous Diffie-Hellman Keying.....	<a href="#">11</a>
<a href="#">4.3</a> Spontaneous GSS-API Exchange.....	<a href="#">11</a>
<a href="#">4.4</a> Spontaneous Key Deletion.....	<a href="#">12</a>
<a href="#">5</a> . TKEY Dynamic Update Requests.....	<a href="#">13</a>
<a href="#">5.1</a> Exchange via TKEY 'Add'.....	<a href="#">13</a>
<a href="#">5.2</a> TKEY Deletion.....	<a href="#">13</a>
<a href="#">6</a> . Methods of Encryption.....	<a href="#">14</a>

<a href="#">7. IANA Considerations.....</a>	<a href="#">15</a>
<a href="#">8. Security Considerations.....</a>	<a href="#">16</a>
References.....	<a href="#">17</a>
Author's Address.....	<a href="#">18</a>
Expiration and File Name.....	<a href="#">18</a>

## [1. Introduction](#)

The Domain Name System (DNS) is a hierarchical, distributed, highly available database used for mapping between domain names and addresses, for email routing, and for other information [RFC 1034, 1035]. It has been extended to provide for public key security and dynamic update [RFC 2136, [draft-ietf-dnssec-secext2](#)/\*.txt, [draft-ietf-dnssec-update2](#)/\*.txt].

[[draft-ietf-dnsind-tsig](#)/\*.txt] provides a means of more efficiently authenticating and securing DNS messages using shared secret keys via the TSIG resource record (RR) but provides no mechanism for setting up such keys other than manual exchange. This document describes a TKEY RR that can be used in a number of different modes to establish such shared secret keys between a DNS resolver and server.

### [1.1 General Principles](#)

TKEY is a meta-RR that is not stored or cached in the DNS and does not appear in zone files. It supports a variety of modes for the establishment and deletion of shared secret keys between DNS entities

such as resolvers and servers. The establishment of such a key requires that state be maintained at both the resolver and the server and the allocation of the resources to maintain such state may require mutual agreement. In the absence of such agreement, servers are free to return errors for any attempt to use TKEY and resolvers are free to ignore any TKEY RRs they receive.

In all cases herein, the term "resolver" includes that part of a server which makes full and incremental [[RFC 1995](#)] zone transfer queries as well as other queries.

Servers are not required to implement any particular mode or modes of the defined modes of TKEY shared secret key establishment or deletion and may return errors for any they do not support. Based on experience, in the future more modes may be added or some modes described herein may be deprecated.

The means by which the shared secret keying material exchanged via TKEY is actually used in any particular TSIG algorithm is algorithm dependent and is defined in connection with that algorithm.

Note that this keying material and TSIGs that use it are associated with DNS hosts. They are not tied to zones. They may be used to authenticate queries and responses but they do not provide zone stored DNS data origin authentication [[draft-ietf-dnssec-secext2-\\*.txt](#)].

Two modes of TKEY, the server assigned and resolver assigned modes, perform encryption which may effect their export or inport status for some countries. All other aspects of DNS security, including the SIG, KEY, NXT, and TSIG RRs and all other defined modes of TKEY perform authentication (signatures and signature verification) only.

## [1.2](#) Overview of Contents

[Section 2](#) below specifies the TKEY resource record (RR) and provides a high level description of its constituent fields.

[Section 3](#) discusses key exchange via queries for type TKEY. This is applicable to the server assigned, Diffie-Hellman exchange, and GSS-

API establishment modes.

[Section 4](#) discusses spontaneous inclusion of TKEY RRs in responses by servers. This is applicable to key deletion and to server assigned and Diffie-Hellman exchange key establishment.

[Section 5](#) discusses use of dynamic update requests for type TKEY. This supports optional key exchange via resolver update request, which is applicable to key deletion and to the resolver assigned mode.

[Section 6](#) describes encryption methods for transmitting secret key information.

[Section 7](#) covers IANA considerations in assignment of TKEY modes.

Finally, [Section 8](#) touches on some security considerations.

## [2.](#) The TKEY Resource Record

The TKEY resource record (RR) has the structure given below. Its RR type code is 249.

Field	Type	Comment
-----	----	-----

NAME	domain	see description below
TTYPE	u_int16_t	TKEY
CLASS	u_int16_t	ignored, should be zero
TTL	u_int32_t	SHOULD be zero
RDLEN	u_int16_t	size of RDATA
RDATA: Algorithm: domain		
Inception:	u_int32_t	
Expiration:	u_int32_t	
Mode:	u_int16_t	
Error:	u_int16_t	
Key Size:	u_int16_t	
Key Data:	octet-stream	
Other Size:	u_int16_t	
Other Data:	octet-stream	undefined by this protocol

The Name field's meaning differs somewhat with mode and context as explained in subsequent sections.

The TTL field SHOULD always be zero to be sure that older DNS implementations do not cache TKEY RRs.

The algorithm name is a domain name with the same meaning as in [[draft-ietf-dnsind-tsig-\\*.txt](#)]. The algorithm determines how the secret keying material exchanged using the TKEY RR is actually used to derive the algorithm specific key that is used.

The inception time and expiration time are in number of seconds since the beginning of 1 January 1970 GMT ignoring leap seconds treated as modulo  $2^{32}$  using ring arithmetic [[RFC 1982](#)]. In messages between a DNS resolver to a DNS server where these fields are meaningful, they are the either requested validity interval for the keying material asked for or specify the validity interval of keying material provided. To avoid reply attack, to keying material used to authenticate TKEY keying material MUST NOT have a lifetime of more than  $2^{31}$  seconds. This applies to keying material used in either a TSIG or a SIG(0) transaction or request signature.

The mode field specifies the general scheme for key agreement. Note that implementation of TKEY as a whole and of any particular mode is optional. The following values of the Mode octet are defined or reserved:

Value	Description
-----	-----
0	- reserved
1	server/responder assignment
2	Diffie-Hellman exchange
3	GSS-API negotiation
4	resolver/querier assignment
5	key deletion
6-65534	- available, see IANA considerations <a href="#">section</a>
<a href="#">65535</a>	-reserved

The error code field is an extended RCODE. The following values are defined:

Value	Description
-----	-----
0	- no error
1-15	a DNS RCODE
16	BADSIG
17	BADKEY
18	BADTIME
19	BADMODE

The key data size field is an unsigned 16 bit integer in network order which specifies the size of the key exchange data field in octets. The meaning of the key data depends on the mode.

The Other Size and Other Data fields are not used. The RDLEN field MUST equal the length of the RDATA section through the end of other data or the RR is to be considered malformed and rejected.



### [3.](#) Exchange via Resolver Query

One method for a resolver and a server to establish a shared secret key for use in TSIG is through queries from the resolver for type TKEY. Such queries MUST either be accompanied by one or more TKEY RRs in the additional information section to indicate the mode(s) in use and other information where required or the resolver and server must have a prior agreement that supplies any information that would otherwise have had to be conveyed by TKEY RR(s) in the query.

For TKEY(s) appearing in a query, the TKEY RR name SHOULD be a domain locally unique at the resolver (or globally unique), less than 128 octets long, and meaningful to the resolver to distinguish keys and/or key agreement sessions. (For resolvers not wishing to make this use of the name, it may be specified as root to minimize length.) For TKEY(s) appearing in a response to a query, the TKEY RR name SHOULD be a globally unique server assigned domain. If the TKEY in a response is the result of a query containing a TKEY with a non-root name, that query TKEY name SHOULD be incorporated as the prefix of the response TKEY name.

Type TKEY queries SHOULD NOT be flagged as recursive and servers MAY ignore the recursive header bit in TKEY queries they receive.

For every mode defined below, the inception and expiration times in a query TKEY are set to the time interval for which the resolver wishes the requested key to be valid and they are set in a successful response to the actual time interval during which the server will consider the key valid. Future modes may be defined which ignore the inception and expiration time fields.

#### [3.1](#) Query for Server Assigned Keying

In server assigned keying, the DNS server host generates the keying material and it is sent to the resolver encrypted under a resolver host key. See [section 6](#) for description of encryption methods.

A resolver sends a query for type TKEY accompanied by a TKEY RR specifying the "server assignment" mode and a resolver host KEY RR to be used in encrypting the response, both in the additional

information section. The TKEY algorithm field is set to the signature algorithm the resolver plans to use. It is recommended that any "key data" provided in the query TKEY be strongly mixed with server generated randomness [[RFC 1750](#)] to derive the keying material to be used. The KEY that appears in the query SHOULD have a zero TTL. It need not be accompanied by a SIG(KEY) and if the query is signed by the resolver host and that signature is verified, then any SIG(KEY) provided MAY be ignored for key exchange purposes. The KEY RR in

such a query SHOULD have a name that corresponds to the resolver host but it is only essential that it be a public key for which the resolver has the corresponding private key so it can decrypt the response data.

Accepting and responding to an unsigned query of this sort may drain some entropy from an entropy pool being maintained by the server and used for secret key generation and so might enable an entropy exhaustion attack. In addition, some significant amount of computational resources may be used in the public key encryption of response data. To protect against these effects, a server SHOULD require such a query to be signed and MAY rate limit responses.

The server response contains a TKEY in its answer section with the server assigned mode. If the error field is non-zero, the query failed for the reason given. If the error field is zero, the KEY RR provided in the query will be echoed back and the key data portion of the response TKEY RR will be the server assigned keying data encrypted under the public key in the KEY RR. The name of the TKEY RR will be the server assigned name of the key and SHOULD be globally unique.

### [3.2](#) Query for Diffie-Hellman Exchanged Keying

Diffie-Hellman (DH) key exchange is means whereby two parties can derive some shared secret information without requiring any secrecy of the messages they exchange [[Schneier](#)]. Provisions have been made for the storage of DH public keys in the DNS [[draft-ietf-dnssec-dhk-\\*.txt](#)].

A client sends a query for type TKEY accompanied by a TKEY RR in the additional information section specifying the "Diffie-Hellman" mode

and accompanied by a KEY RR specifying a client host Diffie-Hellman key. The TKEY algorithm field is set to the signature algorithm the resolver plans to use and any "key data" provided is ignored by the server.

Accepting and responding to an unsigned query of this sort may use significant computation at the server; however, if the server requires that the request be signed, then if no shared secret is in place to permit a TSIG to be used on the request, it would be necessary to use a SIG(0) the verification of which would impose its own computational load.

The server response contains a TKEY in its answer section with the Diffie-Hellman mode. If the error field is non-zero, the query failed for the reason given. If the error field is zero, the client host supplied Diffie-Hellman KEY should be echoed back and a server host

Diffie-Hellman KEY RR will also be present.

Both parties can calculate the same shared secret quantity from the pair of Diffie-Hellman keys used [[Schneier](#)] provided they use the same modulus. If the server host does not have an appropriate Diffie-Hellman key to use for the exchange, it should return the BADKEY error.

### [3.3](#) Query for GSS-API Established

This is described in a separate document [[draft-skwan-gss-tsig-\\*.txt](#)] which should be seen for the full description. Basically, when an acceptable symmetric key is not yet in place, the resolver can send a query for type TKEY with a TKEY specifying the GSS-API mode in the additional information section and a GSS-API token in the key data portion. The server responds with a TKEY specifying the GSS-API mode and a GSS-API token in the key data portion. The resolver and server feed these tokens to their local GSS implementation and iterate until an error is encountered or a key (GSS-API session) is established. A similar exchange can be used to delete a GSS-API session.

Any issues of possible encryption of the GSS-API token data being transmitted are handled by the GSS-API level. In addition, the GSS-

API level provides authentication so that this mode of TKEY query and response MAY be, but do not need to be, signed with TSIG or SIG(0).

#### [4.](#) Spontaneous Server Inclusion

A DNS server may include TKEY RRs spontaneously as additional information in responses. This SHOULD only be done if the server knows the querier understands TKEY and has this option implemented. This technique can be used to establish a server assigned key, a Diffie-Hellman exchange key, for GSS-API exchange, and to delete a key. A disadvantage of this technique is that there is no way for the server to get any immediate error or success indication back and, in the case of UDP, no way to even know if the DNS response reached the resolver.

##### [4.1](#) Spontaneous Server Assigned Keying

A server can include in the additional information section of a response a server assignment mode TKEY with encrypted keying material

in its key data section along with a KEY RR specifying the client public key used for the encryption. Such a response SHOULD be signed but the KEY RR need not be signed by a SIG(KEY). A server should only do this if there is sufficient room in a query and it has reason to believe the resolver will understand such additional data. The KEY RR used MUST be one for which the resolver host has the corresponding private key or it will not be able to decrypt the keying material.

#### [4.2](#) Spontaneous Diffie-Hellman Keying

A server can include in the additional information section of a response a Diffie-Hellman exchange mode TKEY along with two KEY RRs specifying the client and server host public keys used for the exchange. Such a response SHOULD be signed but the KEY RRs need not be signed by a SIG(KEY). A server should only do this if there is sufficient room in a query and it has reason to believe the resolver host will understand such additional data.

#### [4.3](#) Spontaneous GSS-API Exchange

A server can spontaneously include in the additional information section of a response, a GSS-API mode TKEY. The information in the key data section of such a TKEY is a GSS-API token which SHOULD be fed by the resolver to its local GSS-API implementation. If such a response is signed, the signature must verify before processing the data. To the extent that GSS-API provides its own security, such a response may not need to be signed. To the extent that GSS-API

handles duplicated messages, such a spontaneous TKEY can be sent repeatedly, until, perhaps, a response via a GSS-API mode TKEY query is received.

#### [4.4](#) Spontaneous Key Deletion

A server can hint to a client that it has deleted a symmetric key by spontaneously including a TKEY RR in the additional information

section of a response with the key's name and specifying the key deletion mode. Such a response SHOULD be signed. If authenticated, it deletes all keys with the given name whose effective time period overlaps the inception to expiration period given in the TKEY. (If the inception time of one symmetric key is equal to the expiration time of another, or vice versa, they do not overlap.) Failure by a client to receive or properly process such additional information in a response would simply mean that the client might use a key that the server had discarded and then get an error indication.

If a DNS server supports dynamic update [[RFC 2136](#)], then dynamic update request can be used to exchange resolver assigned symmetric keys as described in [section 5.1](#) below and to delete previously exchanged keys from a server as described in [section 5.2](#) below.

### [5.1](#) Exchange via TKEY 'Add'

Optionally, a server can accept resolver assigned keys. The keying material must be encrypted under a server host key for protection in transmission as described in [Section 6](#).

The resolver sends an update request to add a TKEY RR that specifies the keying data with a KEY RR in the additional information section specifying the server host public key used to encrypt the data. The name of the key and the keying data are completely controlled by the sending resolver so a globally unique key name SHOULD be used. The server SHOULD require that this request be signed with a TSIG, if there already exists an appropriate shared secret, or a SIG(0) by the resolver host. The KEY RR used MUST be one for which the server has the corresponding private key or it will not be able to decrypt the keying material.

### [5.2](#) TKEY Deletion

Keys established via TKEY can be treated as soft state. Since DNS transactions are originated by the resolver, the resolver can simply toss keys, although it may have to go through another key exchange if it later needs one. Similarly, the server can discard keys although that will result in an error on receiving a query with a TSIG using the discarded key.

The key expiration provided in the TKEY and the ability of each party to discard keys may be adequate but servers that support dynamic update [[RFC 2136](#)] may optionally implement key deletion whereby the server discards a key on receipt from a resolver of a delete request for a TKEY with the key's name. The mode and most fields of the TKEY being "deleted" are ignored. But, to allow for the possibility of multiple keys with the same name but different time periods, the only key deleted are those whose time period overlaps with that specified by the inception and expiration in the TKEY being "deleted".

## 6. Methods of Encryption

For the server assigned and resolver assigned key exchange, the keying material is sent within the key data field of a TKEY RR encrypted under the private key corresponding to the public key in an accompanying KEY RR [[draft-ietf-dnssec-secext2](#)-\*.\*.txt]. The secret keying material being sent will generally be fairly short, usually less than 256 bits, because that is adequate for very strong protection with modern keyed hash or symmetric algorithms.

If the KEY RR specifies the RSA algorithm, then the keying material is encrypted as per the description of RSA encryption in PKCS-1. (Note, the secret keying material being sent is directly RSA encrypted in PKCS-1 format, It is not "enveloped" under some other symmetric algorithm.) In the unlikely event that the keying material will not fit within one RSA modulus of the chosen public key, additional RSA encryption blocks are included. The length of each block is clear from the public RSA key specified and the PKCS-1 padding makes it clear what part of the encrypted data is actually keying material and what part is formatting or the required at least eight bytes of random [[RFC 1750](#)] padding.



INTERNET-DRAFT

The DNS TKEY RR

September 1998

## [7.](#) IANA Considerations

Mode field values 0x0000 through 0x00FF, and 0xFF00 through 0xFFFF can only be assigned by an IETF standards action excluding Experimental Standards (and 1 through 5 are assigned by this Proposed Standard). Special consideration should be given before the allocation of meaning for Mode field values 0x0000 and 0xFFFF.

Mode field values 0x0100 through 0x0FFF and 0xF000 through 0xFEFF are allocated by an IETF consensus (i.e., at this time, an IESG vote) excluding Experimental Standards.

Mode field values 0x1000 through 0xEFFF are allocated based on RFC documentation of their use or the issuance of an Experimental Standard.

Mode values should not be changed when the status of their use changes. I.E. a mode value assigned for an Experimental Standard should not be changed later just because that standard's status is changed to Proposed.

## 8. Security Considerations

To avoid different interpretations of the inception and expiration times in TKEY RRs, resolvers and servers exchanging them must have the same idea of what time it is. One way of doing this is with the NTP protocol [[RFC 2030](#)] but that or any other time synchronization MUST be done securely.

It is recommended that the server require TKEY queries be signed. However, for currently defined modes, relatively little damage will be done if an unsigned query of this sort is accepted and processed, as described below for each mode. In addition, requiring that a TKEY query be signed by a TSIG (if there exists an acceptable exchanged key between the parties) or a SIG(0) may itself impose significant computational requirements on the server, particularly in verifying SIG(0) public key signatures.

Responses to TKEY queries SHOULD always have DNS transaction signatures to protect the integrity of any keying data, error codes, etc. This signature, if present, MUST use a previously established secret (TSIG) or public (SIG(0)) key and MUST NOT use any key that the response to be verified is itself providing.

---

INTERNET-DRAFT

The DNS TKEY RR

September 1998

## References

PKCS-1 - RSA Encryption Standard (An RSA Laboratories Technical Note).

[RFC 1034](#) - P. Mockapetris, "Domain Names - Concepts and Facilities", STD 13, November 1987.

[RFC 1035](#) - P. Mockapetris, "Domain Names - Implementation and Specifications", STD 13, November 1987.

[RFC 1750](#) - D. Eastlake, S. Crocker & J. Schiller, "Randomness Recommendations for Security", December 1994.

[RFC 1982](#) - Robert Elz, Randy Bush, "Serial Number Arithmetic", 09/03/1996.

[RFC 1995](#) - Masatka Ohta, "Incremental Zone Transfer in DNS", August 1996.

[RFC 2030](#) - D. Mills, "Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI", October 1996.

[RFC 2136](#) - P. Vixie, S. Thomson, Y. Rekhter, J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", 04/21/1997.

[draft-ietf-dnsind-tsig](#)/\*.txt - P. Vixie, O. Gudmundsson, D. Eastlake, "Secret Key Transaction Signatures for DNS (TSIG)".

[draft-ietf-dnssec-dhk](#)/\*.txt - D. Eastlake

[draft-ietf-dnssec-update2](#)/\*.txt - Donald E. Eastlake 3rd, "Secure Domain Name System Dynamic Update".

[draft-ietf-dnssec-secext2](#)/\*.txt - Donald E. Eastlake 3rd, "Domain Name System Security Extensions".

[draft-skwan-gss-tsig](#)/\*.txt - S. Kwan, P. Garg, R. Viswanathan, "GSS Algorithm for TSIG (GSS-TSIG)"

[Schneier] - Bruce Schneier, "Applied Cryptography: Protocols, Algorithms, and Source Code in C", 1996, John Wiley and Sons

Donald E. Eastlake, 3rd

[Page 17]

---

INTERNET-DRAFT

The DNS TKEY RR

September 1998

#### Author's Address

Donald E. Eastlake 3rd  
IBM  
318 Acton Street  
Carlisle, MA 01741 USA

Telephone: +1 978 287 4877  
          +1 914 784 7913  
FAX: +1 978 371 7148  
email: dee3@us.ibm.com

## Expiration and File Name

This draft expires March 1999.

Its file name is [draft-ietf-dnssec-tkey-01.txt](#).