

DNS Working Group
INTERNET-DRAFT
Expires: June 2000
[draft-ietf-dnsind-tkey-03.txt](#)

Donald E. Eastlake, 3rd
IBM
December 1999

Secret Key Establishment for DNS (TKEY RR)

Donald E. Eastlake 3rd

Status of This Document

This draft, file name [draft-ietf-dnsind-tkey-03.txt](#), is intended to become a Proposed Standard RFC. Distribution of this document is unlimited. Comments should be sent to the DNS working group mailing list <namedroppers@internic.net> or to the author.

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#). 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.''

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>.

INTERNET-DRAFT

The DNS TKEY RR

December 1999

Abstract

[[draft-ietf-dnsind-tsig-*.txt](#)] provides a means of authenticating Domain Name System (DNS) queries and responses using shared secret keys via the TSIG resource record. 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.

Acknowledgments

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

Olafur Gudmundsson (TIS)

Stuart Kwan (Microsoft)

Brian Wellington (TIS)

INTERNET-DRAFT

The DNS TKEY RR

December 1999

Table of Contents

Status of This Document.....	1
Abstract.....	2
Acknowledgments.....	2
Table of Contents.....	3
1 . Introduction.....	4
1.1 General Principles.....	4
1.2 Overview of Contents.....	5
2 . The TKEY Resource Record.....	5
2.1 Key Naming.....	7
3 . Exchange via Resolver Query.....	8
3.1 Query for Server Assigned Keying.....	8
3.2 Query for Diffie-Hellman Exchanged Keying.....	9
3.3 Query for GSS-API Established.....	11
3.4 Query for Querier Assigned Keying.....	11
3.5 Query for TKEY Deletion.....	12
4 . Spontaneous Server Inclusion.....	12
4.1 Spontaneous GSS-API Exchange.....	12
4.2 Spontaneous Server Key Deletion.....	13
5 . Methods of Encryption.....	13
6 . IANA Considerations.....	14
7 . Security Considerations.....	14
Changes from Previous Draft.....	15
References.....	16

Author's Address.....	17
Expiration and File Name.....	17

[1](#). Introduction

The Domain Name System (DNS) is a hierarchical, distributed, highly available database used for bi-directional 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 2535, [RFC 2136](#)]. Familiarity with these RFCs is assumed.

[\[draft-ietf-dnsind-tsig-*.txt\]](#) provides a means of efficiently authenticating 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 and delete such shared secret keys between a DNS resolver and server.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC 2119](#)].

[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 MUST return errors such as NotImp or Refused for an 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.

The shared secret keying material developed by using TKEY is a plain octet sequence. The means by which this 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. For example, see [[RFC 2104](#)] for how TKEY agreed shared secret keying material is used in HMAC-MD5.SIG-ALG.REG.INT or any other HMAC algorithm.

Note that TKEY established keying material and TSIGs that use it are associated with DNS hosts. They are not associated with zones. They may be used to authenticate queries and responses but they do not provide zone based DNS data origin or denial authentication [RFC

2535].

Certain modes of TKEY perform encryption which may affect their export or import status for some countries. The affected modes specified in this document are the server assigned mode and the resolver assigned mode.

There MUST NOT be more than one TKEY RR in a DNS query or response.

[1.2](#) Overview of Contents

[Section 2](#) below specifies the TKEY resource record and provides a

description of its constituent fields.

[Section 3](#) discusses key agreement and deletion via DNS requests with the Query opcode for type TKEY. This method is applicable to all currently defined TKEY modes although in some cases it is not what would intuitively be called a "query".

[Section 4](#) discusses spontaneous inclusion of TKEY RRs in responses by servers. This is applicable to key deletion and to the GSS-API mode.

[Section 5](#) describes encryption methods for transmitting secret key information.

[Section 6](#) covers IANA considerations in assignment of TKEY modes.

Finally, [Section 7](#) 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 255 (ANY)

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 relates to naming keys. Its meaning differs somewhat with mode and context as explained in subsequent sections. [Section 2.1](#) below gives key naming guidelines.

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 agreed to 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 either the requested validity interval for the keying material asked for or specify the validity interval of keying material provided. See Security Considerations section in reference to replay attacks.

The mode field specifies the general scheme for key agreement or purpose of the TKEY DNS message. Server and resolvers supporting this specification MUST implement the Diffie-Hellman key agreement and the key deletion modes. All other modes are OPTIONAL. A server supporting TKEY that receives a TKEY request with a mode it does not support MUST return the BADMODE error. The following values of the Mode octet are defined, available, 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 section 65535
	-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
20	BADNAME
21	BADALG

When an extended RCODE appears in the TKEY in a response, the DNS header RCODE field indicates no error.

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 in this specification but may be used in future extensions. 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.

[2.1](#) Key Naming

At any host only one octet string of keying material may be in place at the same time for any particular key name. An attempt to establish another set of keying material at a server for an existing name SHOULD return a BADNAME error.

A reasonable key naming strategy is as follows:

If the key is generated as the result of a query with root as

its owner name, they the server SHOULD create a pseudo-random

[RFC 1750] based unique name ending with a name of the server host. For example 89n3mdg072pp.server.example.com. If generation of a new pseudo-random name in each case is an excessive computation load or entropy drain, a serial number prefix can be added to a single pseudo-random name generated at DNS server start time, such as 1001.89n3mdg072pp.server.example.com.

If the key is generated as the result of a query with a non-root name, say 1001.foo.example.net, then use the concatenation of that with a name of the server host. For example 1001.foo.example.net.server.example.com.

[3.](#) Exchange via Resolver Query

One method for a resolver and a server to agree about shared secret keying material for use in TSIG is through DNS requests from the resolver which are syntactically DNS queries for type TKEY. Such queries MUST be accompanied by a TKEY RR in the additional information section to indicate the mode in use and accompanied by other information where required.

For a TKEY 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 assist in distinguishing 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 a prefix of the response TKEY name. See [section 2.1](#) suggesting a more specific naming strategy.

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

[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 public key. See [section 5](#) 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" optionally provided in the query TKEY RR by the resolver be strongly mixed with server generated randomness [[RFC 1750](#)] to derive the keying material to be used. The KEY RR that appears in the query SHOULD have a zero TTL and it need not be accompanied by a SIG(KEY) RR. If the query is signed by the resolver host with a TSIG RR [[draft-ietf-dnsind-tsig-*.txt](#)] or SIG(0) RR and that signature is verified, then any SIG(KEY) provided in the query SHOULD be ignored. 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.

The server response contains a TKEY RR in its answer section with the server assigned mode and echoes back the KEY RR provided in the query in its additional information section.

If the error field is zero, the key data portion of the response TKEY RR will be the server assigned keying data encrypted under the public key in the resolver provided KEY RR. In this case, the name of the answer TKEY RR will be the server assigned name of the key and SHOULD be globally unique.

If the error field of the response TKEY is non-zero, the query failed for the reason given. FORMERR is given if the query specified no encryption key.

The inception and expiry times in the query TKEY RR are those requested for the keying material. The inception and expiry times in the response TKEY are the maximum period the server will consider the keying material valid. Servers may pre-expire keys so this is not a

guarantee.

NOTE: Accepting and responding to an unsigned query of this mode 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.

[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

Donald E. Eastlake, 3rd

[Page 9]

INTERNET-DRAFT

The DNS TKEY RR

December 1999

of the messages they exchange [[Schneier](#)]. Provisions have been made for the storage of DH public keys in the DNS [[RFC 2539](#)].

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 also in the additional information section specifying a client host Diffie-Hellman key. The TKEY RR algorithm field is set to the signature algorithm the resolver plans to use. The "key data" provided in the TKEY is used as a nonce to avoid always deriving the same keying material for the same pair of DH KEYS.

The server response contains a TKEY in its answer section with the Diffie-Hellman mode. The "key data" provided in this TKEY is used as an additional nonce to avoid always deriving the same keying material for the same pair of DH KEYS. If the error field is non-zero, the query failed for the reason given. FORMERR is given if the query included no DH KEY and BADKEY is given if the query included an incompatible DH KEY.

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 in the response. Both parties can then calculate the same shared secret quantity from the pair of Diffie-Hellman keys used [[Schneier](#)], provided they use the same modulus, and

the data in the TKEY RRs. The TKEY RR data is mixed with the DH result as follows:

```
keying material =  
    XOR ( DH value, MD5 ( query data | DH value ) |  
          MD5 ( server data | DH value ) )
```

where XOR is a byte wise left justified xor padding the shorter octet stream with zeros, DH value is the Diffie-Hellman value derived from the KEY RRs, "query data" and "server data" are the TKEY RR data fields sent by those parties, and "|" is concatenation. These "query data" and "server data" nonces are suffixed by the DH value, digested by MD5, the results concatenated, and then XORed with the DH value.

The inception and expiry times in the query TKEY RR are those requested for the keying material. The inception and expiry times in the response TKEY RR are the maximum period the server will consider the keying material valid. Servers may pre-expire keys so this is not a guarantee.

NOTE: Accepting and responding to an unsigned query of this mode may use significant computation at the server; however, if the server requires that the request be signed and if no shared secret is in place to permit a TSIG [[draft-ietf-dnsind-tsig-*.txt](#)] 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.

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

This is described in a separate document which should be seen for the full description. Basically the resolver and server can exchange queries and responses for type TKEY with a TKEY RR specifying the GSS-API mode in the additional information section and a GSS-API token in the key data portion.

Any issues of possible encryption of parts the GSS-API token data being transmitted are handled by the GSS-API level. In addition, the GSS-API level provides its own authentication so that this mode of TKEY query and response MAY be, but do not need to be, signed with

TSIG RR or SIG(0) RR.

The inception and expiry time in a GSS-API mode TKEY RR are ignored.

[3.4](#) Query for Querier Assigned Keying

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 5](#).

The resolver sends a TKEY query with a TKEY RR that specifies the keying data and a KEY RR specifying the server host public key used to encrypt the data both in the additional information section. 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 querying 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 and will return a FORMERR.

The query TKEY RR inception and expiry give the time period the querier intends to consider the keying material valid. The server can return a lesser time interval to advise that it will not maintain state for that long and can pre-expire keys in any case.

[3.5](#) Query for 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.

To avoid attempted reliance in queries on keys no longer in effect, servers MUST implement key deletion whereby the server "discards" a key on receipt from a resolver of an authenticated delete request for a TKEY RR with the key's name. If the server has no record of a key with that name, it returns BADNAME.

Key deletion TKEY queries MUST be signed. This signature may be a TSIG RR using the key to be deleted.

For querier assigned and Diffie-Hellman keys, the server MUST truly "discard" all active state associated with the key. For server assigned keys, the server MAY simply mark the key as no longer retained by the client and may re-send it in response to a future query for server assigned keying material.

[4. Spontaneous Server Inclusion](#)

A DNS server may include a TKEY RR 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 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 GSS-API Exchange](#)

A server can spontaneously include in the additional information section of a response, a GSS-API mode TKEY RR. 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, for example, a response via a GSS-API mode TKEY

[4.2](#) Spontaneous Server Key Deletion

A server can optionally tell 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" the key with the given name. The inception and expiry times of the delete TKEY RR are ignored. Failure by a client to receive or properly process such additional information in a response would mean that the client might use a key that the server had discarded and would then get an error indication.

For server assigned and Diffie-Hellman keys, the client must truly "discard" all active state associated with the key. For querier assigned keys, the queries MAY simply mark the key as no longer retained by the server and may re-send it in a future query specifying querier assigned keying material.

[5.](#) Methods of Encryption

For the server assigned and resolver assigned key agreement, the keying material is sent within the key data field of a TKEY RR encrypted under the public key in an accompanying KEY RR [[RFC 2535](#)]. This KEY RR MUST be for a public key algorithm where the public and private keys can be used for encryption and the corresponding decryption which recovers the originally encrypted data. The KEY RR MUST correspond to a name for the decrypting host such that the decrypting host has the corresponding private key to decrypt the data. 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 [[RFC 2437](#)]. (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

December 1999

6. IANA Considerations

This section is to be interpreted as provided in [[RFC 2434](#)].

Mode field values 0x0000 through 0x00FF, and 0xFF00 through 0xFFFF can only be assigned by an IETF standards action (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.

Mode field values 0x1000 through 0xEFFF are allocated based on RFC documentation of their use.

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.

7. 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.

TKEY queries SHOULD be signed and those using the querier establishment mode MUST be signed to authenticate their origin. However, for currently defined modes, relatively little damage will be done if an unsigned query of this sort is accepted and processed, as described above, 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 MUST always have DNS transaction signatures to protect the integrity of any keying data, error codes, etc. This

signature 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.

To avoid replay attacks, it is necessary that a response or querier establishment mode query involving TKEY not be valid if replayed on the order of 2^{32} second (about 136 years) later. To accomplish

Donald E. Eastlake, 3rd

[Page 14]

INTERNET-DRAFT

The DNS TKEY RR

December 1999

this, the keying material used in any TSIG or SIG(0) RR that authenticates a TKEY message MUST NOT have a lifetime of more than $2^{31} - 1$ seconds (about 68 years). Thus, on attempted replay, the authenticating TSIG or SIG(0) RR will not be verifiable due to key expiration and the replay will fail.

General protection against denial of service via the use of TKEY is not provided.

Changes from Previous Draft

Fix one letter typo in [Section 5](#).

Make default CLASS for TKEY be ANY.

INTERNET-DRAFT

The DNS TKEY RR

December 1999

References

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

[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](#) - Masataka 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 2104](#) - H. Krawczyk, M. Bellare, R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", February 1997.

[RFC 2119](#) - S. Bradner, "Key words for use in RFCs to Indicate

Requirement Levels", March 1997.

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

[RFC 2434](#) - T. Narten, H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs, October 1998.

[RFC 2437](#) - B. Kaliski, J. Staddon, "PKCS #1: RSA Cryptography Specifications Version 2.0", October 1998.

[RFC 2535](#) - D. Eastlake, "Domain Name System Security Extensions", March 1999.

[RFC 2539](#) - D. Eastlake, "Storage of Diffie-Hellman Keys in the Domain Name System (DNS)", March 1999.

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

Donald E. Eastlake, 3rd

[Page 16]

INTERNET-DRAFT

The DNS TKEY RR

December 1999

Author's Address

Donald E. Eastlake 3rd
65 Shindegan Hill Road, RR #1
Carmel, NY 10512 USA

Telephone: +1 914-276-2668 (h)
 +1 914-784-7913 (w)
FAX: +1 914-276-2947 (h)
email: dee3@torque.pothole.com

Expiration and File Name

This draft expires June 2000.

Its file name is [draft-ietf-dnsind-tkey-03.txt](#).

