

NAME

Net::DNS::RR::TSIG – DNS TSIG resource record

SYNOPSIS

```
use Net::DNS;
$tsig = create Net::DNS::RR::TSIG( $keyfile );

$tsig = create Net::DNS::RR::TSIG( $keyfile,
                                   fudge => 300
                                   );
```

DESCRIPTION

Class for DNS Transaction Signature (TSIG) resource records.

METHODS

The available methods are those inherited from the base class augmented by the type-specific methods defined in this package.

Use of undocumented package features or direct access to internal data structures is discouraged and could result in program termination or other unpredictable behaviour.

algorithm

```
$algorithm = $rr->algorithm;
$rr->algorithm( $algorithm );
```

A domain name which specifies the name of the algorithm.

key

```
$rr->key( $key );
```

Base64 representation of the key material.

keybin

```
$rr->keybin( $keybin );
```

Binary representation of the key material.

time_signed

```
$time_signed = $rr->time_signed;
$rr->time_signed( $time_signed );
```

Signing time as the number of seconds since 1 Jan 1970 00:00:00 UTC. The default signing time is the current time.

fudge

```
$fudge = $rr->fudge;
$rr->fudge( $fudge );
```

“fudge” represents the permitted error in the signing time. The default fudge is 300 seconds.

mac

```
$rr->mac( $mac );
```

Message authentication code (MAC). The programmer must call the Net::DNS::Packet **data()** object method before this will return anything meaningful.

macbin

```
$macbin = $rr->macbin;
$rr->macbin( $macbin );
```

Binary message authentication code (MAC).

prior_mac

```
$prior_mac = $rr->prior_mac;
$rr->prior_mac( $prior_mac );
```

Prior message authentication code (MAC).

prior_macbin

```
$prior_macbin = $rr->prior_macbin;
$rr->prior_macbin( $prior_macbin );
```

Binary prior message authentication code.

request_mac

```
$request_mac = $rr->request_mac;
$rr->request_mac( $request_mac );
```

Request message authentication code (MAC).

request_macbin

```
$request_macbin = $rr->request_macbin;
$rr->request_macbin( $request_macbin );
```

Binary request message authentication code.

original_id

```
$original_id = $rr->original_id;
$rr->original_id( $original_id );
```

The message ID from the header of the original packet.

error

vrfyerrstr

```
$rcode = $tsig->error;
```

Returns the RCODE covering TSIG processing. Common values are NOERROR, BADSIG, BADKEY, and BADTIME. See RFC 2845 for details.

other

```
$other = $tsig->other;
```

This field should be empty unless the error is BADTIME, in which case it will contain the server time as the number of seconds since 1 Jan 1970 00:00:00 UTC.

sig_function

```
sub signing_function {
    my ( $keybin, $data ) = @_;

    my $hmac = new Digest::HMAC( $keybin, 'Digest::MD5' );
    $hmac->add( $data );
    return $hmac->digest;
}
```

```
$tsig->sig_function( \&signing_function );
```

This sets the signing function to be used for this TSIG record. The default signing function is HMAC-MD5.

sig_data

```
$sigdata = $tsig->sig_data($packet);
```

Returns the packet packed according to RFC2845 in a form for signing. This is only needed if you want to supply an external signing function, such as is needed for TSIG-GSS.

create

```
$tsig = create Net::DNS::RR::TSIG( $keyfile );

$tsig = create Net::DNS::RR::TSIG( $keyfile,
                                   fudge => 300
                                   );
```

Returns a TSIG RR constructed using the parameters in the specified key file, which is assumed to have been generated by `dnssec-keygen`.

```
$tsig = create Net::DNS::RR::TSIG( $keyname, $key );
```

The two argument form is supported for backward compatibility.

verify

```
$verify = $tsig->verify( $data );
```

```
$verify = $tsig->verify( $packet );
```

```
$verify = $tsig->verify( $reply, $query );
```

```
$verify = $tsig->verify( $packet, $prior );
```

The boolean `verify` method will return true if the hash over the packet data conforms to the data in the TSIG itself

TSIG Keys

TSIG keys are symmetric keys generated using `dnssec-keygen`:

```
$ dnssec-keygen -a HMAC-SHA1 -b 160 -n HOST <keyname>
```

The key will be stored as a private and public keyfile pair
K<keyname>+161+<keyid>.private and K<keyname>+161+<keyid>.key

where

<keyname> is the DNS name of the key.

<keyid> is the (generated) numerical identifier used to distinguish this key.

Other algorithms may be substituted for HMAC-SHA1 in the above example.

It is recommended that the keyname be globally unique and incorporate the fully qualified domain names of the resolver and nameserver in that order. It should be possible for more than one key to be in use simultaneously between any such pair of hosts.

Although the formats differ, the private and public keys are identical and both should be stored and handled as secret data.

Configuring BIND Nameserver

The following lines must be added to the `/etc/named.conf` file:

```
key <keyname> {
    algorithm HMAC-SHA1;
    secret "<keydata>";
};
```

<keyname> is the name of the key chosen when the key was generated.

<keydata> is the key string extracted from the generated key file.

ACKNOWLEDGMENT

Most of the code in the `Net::DNS::RR::TSIG` module was contributed by Chris Turbeville.

Support for external signing functions was added by Andrew Tridgell.

TSIG verification, BIND keyfile handling and support for HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384 and HMAC-SHA512 functions was added by Dick Franks.

BUGS

A 32-bit representation of time is used, contrary to RFC2845 which demands 48 bits. This design decision will need to be reviewed before the code stops working on 7 February 2106.

COPYRIGHT

Copyright (c)2000,2001 Michael Fuhr.

Portions Copyright (c)2002,2003 Chris Reinhardt.

Portions Copyright (c)2013 Dick Franks.

All rights reserved.

Package template (c)2009,2012 O.M.Kolkman and R.W.Franks.

LICENSE

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of the author not be used in advertising or publicity pertaining to distribution of the software without specific prior written permission.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

SEE ALSO

perl, Net::DNS, Net::DNS::RR, RFC2845, RFC4635

TSIG Algorithm Names <<http://www.iana.org/assignments/tsig-algorithm-names>>