NAME

Net::DNS::RR::RRSIG – DNS RRSIG resource record

SYNOPSIS

```
use Net::DNS;
$rr = new Net::DNS::RR('name RRSIG typecovered algorithm labels
                            orgttl sigexpiration siginception
                            keytag signame signature');

use Net::DNS::SEC;
$sigrr = create Net::DNS::RR::RRSIG( \@rrset, $keypath,
                                     sigex => 20191231010101
                                     sigin => 20191201010101
                                     );

$sigrr->verify( \@rrset, $keyrr ) || die $sigrr->vrfyerrstr;
```

DESCRIPTION

Class for DNS digital signature (RRSIG) resource records.

In addition to the regular methods inherited from Net::DNS::RR the class contains a method to sign RRsets using private keys (create) and a method for verifying signatures over RRsets (verify).

The RRSIG RR is an implementation of RFC4034. See Net::DNS::RR::SIG for an implementation of SIG0 (RFC2931).

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.

typecovered

```
$typecovered = $rr->typecovered;
```

The typecovered field identifies the type of the RRset that is covered by this RRSIG record.

algorithm

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

The algorithm number field identifies the cryptographic algorithm used to create the signature.

algorithm() may also be invoked as a class method or simple function to perform mnemonic and numeric code translation.

labels

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

The labels field specifies the number of labels in the original RRSIG RR owner name.

orgttl

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

The original TTL field specifies the TTL of the covered RRset as it appears in the authoritative zone.

sigexpiration and siginception times
sigex sigin sigval

```
$expiration = $rr->sigexpiration;
$expiration = $rr->sigexpiration( $value );

$inception = $rr->siginception;
```

```
$inception = $rr->siginception( $value );
```

The signature expiration and inception fields specify a validity time interval for the signature.

The value may be specified by a string with format 'yyyymmddhhmmss' or a Perl **time()** value.

Return values are dual-valued, providing either a string value or numerical Perl **time()** value.

keytag

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

The keytag field contains the key tag value of the DNSKEY RR that validates this signature.

signame

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

The signer name field value identifies the owner name of the DNSKEY RR that a validator is supposed to use to validate this signature.

signature
sig

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

The Signature field contains the cryptographic signature that covers the RRSIG RDATA (excluding the Signature field) and the RRset specified by the RRSIG owner name, RRSIG class, and RRSIG type covered fields.

sigbin

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

Binary representation of the cryptographic signature.

create

Create a signature over a RR set.

```
use Net::DNS::SEC;

$keypath = '/home/olaf/keys/Kbla.foo.+001+60114.private';

$sigrr = create Net::DNS::RR::RRSIG( \@rrsetref, $keypath );

$sigrr = create Net::DNS::RR::RRSIG( \@rrsetref, $keypath,
                                     sigex => 20191231010101
                                     sigin => 20191201010101
                                     );
$sigrr->print;


# Alternatively use Net::DNS::SEC::Private

$private = Net::DNS::SEC::Private->new($keypath);

$sigrr= create Net::DNS::RR::RRSIG( \@rrsetref, $private );
```

create() is an alternative constructor for a RRSIG RR object.

This method returns an RRSIG with the signature over the subject rrset (an array of RRs) made with the private key stored in the key file.

The first argument is a reference to an array that contains the RRset that needs to be signed.

The second argument is a string which specifies the path to a file containing the private key as generated by dnssec-keygen.

The optional remaining arguments consist of (name => value) pairs as follows:

```
sigex  => 20191231010101,       # signature expiration
sigin  => 20191201010101,       # signature inception
sigval => 30,                   # validity window (days)
ttl    => 3600                  # TTL
```

The sigin and sigex values may be specified as Perl time values or as a string with the format 'yyyymmddhhmmss'. The default for sigin is the time of signing.

The sigval argument specifies the signature validity window in days (sigex = sigin + sigval).

By default the signature is valid for 30 days.

By default the TTL matches the RRset that is presented for signing.

verify

```
$verify = $sigrr->verify( $rrsetref, $keyrr );
$verify = $sigrr->verify( $rrsetref, [$keyrr, $keyrr2, $keyrr3] );
```

`$rrsetref` contains a reference to an array of RR objects and the method verifies the RRset against the signature contained in the `$sigrr` object itself using the public key in `$keyrr`.

The second argument can either be a Net::DNS::RR::KEYRR object or a reference to an array of such objects. Verification will return successful as soon as one of the keys in the array leads to positive validation.

Returns 0 on error and sets `$sig->`vrfyerrstr

vrfyerrstr

```
$verify = $sigrr->verify( $rrsetref, $keyrr );
print $sigrr->vrfyerrstr unless $verify;

$sigrr->verify( $rrsetref, $keyrr ) || die $sigrr->vrfyerrstr;
```

KEY GENERATION

Private key files and corresponding public DNSKEY records are most conveniently generated using dnssec-keygen, a program that comes with the ISC BIND distribution.

```
dnssec-keygen -a 10 -b 2048 -f ksk  rsa.example.
dnssec-keygen -a 10 -b 1024         rsa.example.

dnssec-keygen -a 14 -f ksk  ecdsa.example.
dnssec-keygen -a 14         ecdsa.example.
```

Do not change the name of the private key file. The create method uses the filename as generated by dnssec-keygen to determine the keyowner, algorithm, and the keyid (keytag).

REMARKS

The code is not optimised for speed. It is probably not suitable to be used for signing large zones.

If this code is still around in 2100 (not a leap year) you will need to check for proper handling of times after 28th February.

ACKNOWLEDGMENTS

Although their original code may have disappeared following redesign of Net::DNS, Net::DNS::SEC and the OpenSSL API, the following individual contributors deserve to be recognised for their significant influence on the development of the RRSIG package.

Andy Vaskys (Network Associates Laboratories) supplied code for RSA.

T.J. Mather provided support for the DSA algorithm.

Dick Franks added support for elliptic curve and Edwards curve algorithms.

Mike McCauley created the Crypt::OpenSSL::ECDSA perl extension module specifically for this development.

COPYRIGHT

LICENSE

SEE ALSO

perl, Net::DNS, Net::DNS::RR, Net::DNS::SEC, RFC4034, RFC6840, RFC3755, Net::DNS::SEC::DSA, Net::DNS::SEC::ECDSA, Net::DNS::SEC::EdDSA, Net::DNS::SEC::RSA

Algorithm Numbers <http://www.iana.org/assignments/dns-sec-alg-numbers>

BIND 9 Administrator Reference Manual <http://www.bind9.net/manuals>