

**NAME**

ldns\_dnssec\_zone\_find\_rrset, ldns\_dnssec\_zone\_new, ldns\_dnssec\_zone\_free, ldns\_dnssec\_zone\_add\_rr, ldns\_dnssec\_zone\_names\_print, ldns\_dnssec\_zone\_print, ldns\_dnssec\_zone\_add\_empty\_nonterminals - functions for ldns\_dnssec\_zone

**SYNOPSIS**

```
#include <stdint.h>
```

```
#include <stdbool.h>
```

```
#include <ldns/ldns.h>
```

```
ldns_dnssec_rrsets* ldns_dnssec_zone_find_rrset(const ldns_dnssec_zone *zone, const ldns_rdf *dname, ldns_rr_type type);
```

```
ldns_dnssec_zone* ldns_dnssec_zone_new(void);
```

```
void ldns_dnssec_zone_free(ldns_dnssec_zone *zone);
```

```
ldns_status ldns_dnssec_zone_add_rr(ldns_dnssec_zone *zone, ldns_rr *rr);
```

```
void ldns_dnssec_zone_names_print(FILE *out, const ldns_rbtree_t *tree, bool print_soa);
```

```
void ldns_dnssec_zone_print(FILE *out, const ldns_dnssec_zone *zone);
```

```
ldns_status ldns_dnssec_zone_add_empty_nonterminals(ldns_dnssec_zone *zone);
```

**DESCRIPTION**

*ldns\_dnssec\_zone\_find\_rrset()* Find the RRset with the given name and type in the zone

**zone:** the zone structure to find the RRset in

**dname:** the domain name of the RRset to find

**type:** the type of the RRset to find

Returns the RRset, or NULL if not present

*ldns\_dnssec\_zone\_new()* Creates a new dnssec\_zone structure

Returns the allocated structure

*ldns\_dnssec\_zone\_free()* Frees the given zone structure, and its rbtree of dnssec\_names Individual

ldns\_rr RRs within those names are *not* freed

**\*zone:** the zone to free

*ldns\_dnssec\_zone\_add\_rr()* Adds the given RR to the zone. It find whether there is a dnssec\_name with that name present. If so, add it to that, if not create a new one. Special handling of NSEC and RRSIG provided

**zone:** the zone to add the RR to

**rr:** The RR to add

Returns LDNS\_STATUS\_OK on success, an error code otherwise

*ldns\_dnssec\_zone\_names\_print()* Prints the rbtree of ldns\_dnssec\_name structures to the file descriptor

**out:** the file descriptor to print the names to

**tree:** the tree of ldns\_dnssec\_name structures to print

**print\_soa:** if true, print SOA records, if false, skip them

*ldns\_dnssec\_zone\_print()* Prints the complete zone to the given file descriptor

**out:** the file descriptor to print to

**zone:** the dnssec\_zone to print

*ldns\_dnssec\_zone\_add\_empty\_nonterminals()* Adds explicit dnssec\_name structures for the empty nonterminals in this zone. (this is needed for NSEC3 generation)

**zone:** the zone to check for empty nonterminals return LDNS\_STATUS\_OK on success.

## AUTHOR

The ldns team at NLnet Labs.

## REPORTING BUGS

Please report bugs to dns-team@nlnetlabs.nl or on GitHub at <https://github.com/NLnetLabs/ldns/issues>

## COPYRIGHT

Copyright (c) 2004 - 2006 NLnet Labs.

Licensed under the BSD License. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**SEE ALSO**

*ldns\_dnssec\_zone*. And **perldoc Net::DNS, RFC1034, RFC1035, RFC4033, RFC4034** and **RFC4035**.

**REMARKS**

This manpage was automatically generated from the ldns source code.