



Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!

Rocky Enterprise Linux 9.2 Manual Pages on command 'IO::Socket::SSL::PublicSuffix.3pm'

\$ man IO::Socket::SSL::PublicSuffix.3pm

IO::Socket::SSL::PublicSuffix(3User Contributed Perl DocumentatIO::Socket::SSL::PublicSuffix(3pm)

NAME

IO::Socket::SSL::PublicSuffix - provide access to Mozilla's list of effective TLD names

SYNOPSIS

```
# use builtin default

use IO::Socket::SSL::PublicSuffix;

$ps = IO::Socket::SSL::PublicSuffix->default;

# load from string

$ps = IO::Socket::SSL::PublicSuffix->from_string("*.uk\n*");

# load from file or file handle

$ps = IO::Socket::SSL::PublicSuffix->from_file($filename);

$ps = IO::Socket::SSL::PublicSuffix->from_file(\*STDIN);

# --- string in -> string out

# $rest -> whatever.host

# $tld -> co.uk

my ($rest,$tld) = $ps->public_suffix('whatever.host.co.uk');

my $tld = $ps->public_suffix('whatever.host.co.uk');
```

```
# $root_domain -> host.co.uk  
my $root_domain = $ps->public_suffix('whatever.host.co.uk', 1);  
  
# --- array in -> array out  
# $rest -> [qw(whatever host)]  
# $tld -> [qw(co uk)]  
my ($rest,$tld) = $ps->public_suffix([qw(whatever host co uk)]);
```

```
# To update this file with the current list:
```

```
perl -MIO::Socket::SSL::PublicSuffix -e 'IO::Socket::SSL::PublicSuffix::update_self_from_url()'
```

DESCRIPTION

This module uses the list of effective top level domain names from the mozilla project to determine the public top level domain for a given hostname.

Method

```
class->default(%args)
```

Returns object with builtin default. "min_suffix" can be given in %args to specify the minimal suffix, default is 1.

```
class->from_string(string,%args)
```

Returns object with configuration from string. See method "default" for %args.

```
class->from_file( file name| file handle, %args )
```

Returns object with configuration from file or file handle. See method "default" for %args.

```
$self->public_suffix( $host| \@host, [ $add ] )
```

In array context the function returns the non-tld part and the tld part of the given hostname, in scalar context only the tld part. It adds \$add parts of the non-tld part to the tld, e.g. with "\$add=1" it will return the root domain.

If there were no explicit matches against the public suffix configuration it will fall back to a suffix of length 1.

The function accepts a string or an array-ref (e.g. host split by "."). In the first case it will return string(s), in the latter case array-ref(s).

International hostnames or labels can be in ASCII (IDNA form starting with "xn--") or unicode. In the latter case an IDNA handling library needs to be available. URI is preferred, but Net::IDN::Encode, Net::LibIDN are still supported.

`($self|class)->can_idn`

Returns true if IDN support is available.

FILES

http://publicsuffix.org/list/effective_tld_names.dat

SEE ALSO

Domain::PublicSuffix, Mozilla::PublicSuffix

BUGS

Q: Why yet another module, we already have L<Domain::PublicSuffix> and L<Mozilla::PublicSuffix>.

A: Because the public suffix data change more often than these modules do, IO::Socket::SSL needs this list and it is more easy this way to keep it up-to-date.

AUTHOR

Steffen Ullrich