NAME

Authen::SASL::Perl::GSSAPI – GSSAPI (Kerberosv5) Authentication class

SYNOPSIS

```
use Authen::SASL qw(Perl);

$sasl = Authen::SASL->new( mechanism => 'GSSAPI' );

$sasl = Authen::SASL->new( mechanism => 'GSSAPI',
                           callback => { pass => $mycred });

$sasl->client_start( $service, $host );
```

DESCRIPTION

This method implements the client part of the GSSAPI SASL algorithm, as described in RFC 2222 section 7.2.1 resp. draft–ietf–sasl–gssapi–XX.txt.

With a valid Kerberos 5 credentials cache (aka TGT) it allows to connect to *service@host* given as the first two parameters to Authen::SASL's *client_start()* method. Alternatively, a GSSAPI::Cred object can be passed in via the Authen::SASL callback hash using the 'pass' key.

Please note that this module does not currently implement a SASL security layer following authentication. Unless the connection is protected by other means, such as TLS, it will be vulnerable to man-in-the-middle attacks. If security layers are required, then the Authen::SASL::XS GSSAPI module should be used instead.

CALLBACK

The callbacks used are:

authname
: The authorization identity to be used in SASL exchange

gssmech
: The GSS mechanism to be used in the connection

pass
: The GSS credentials to be used in the connection (optional)

EXAMPLE

```
#! /usr/bin/perl -w

use strict;

use Net::LDAP 0.33;
use Authen::SASL 2.10;

# -------- Adjust to your environment --------
my $adhost      = 'theserver.bla.net';
my $ldap_base   = 'dc=bla,dc=net';
my $ldap_filter = '(&(sAMAccountName=BLAAGROL))';

my $sasl = Authen::SASL->new(mechanism => 'GSSAPI');
my $ldap;

eval {
    $ldap = Net::LDAP->new($adhost,
                           onerror => 'die')
      or  die "Cannot connect to LDAP host '$adhost': '$@'";
    $ldap->bind(sasl => $sasl);
};
```

```
if ($@) {
    chomp $@;
    die   "\nBind error         : $@",
          "\nDetailed SASL error: ", $sasl->error,
          "\nTerminated";
}

print "\nLDAP bind() succeeded, working in authenticated state";

my $mesg = $ldap->search(base   => $ldap_base,
                         filter => $ldap_filter);

# -------- evaluate $mesg
```

PROPERTIES

The properties used are:

maxbuf
: The maximum buffer size for receiving cipher text

minssf
: The minimum SSF value that should be provided by the SASL security layer. The default is 0

maxssf
: The maximum SSF value that should be provided by the SASL security layer. The default is 2**31

externalssf
: The SSF value provided by an underlying external security layer. The default is 0

ssf
: The actual SSF value provided by the SASL security layer after the SASL authentication phase has been completed. This value is read-only and set by the implementation after the SASL authentication phase has been completed.

maxout
: The maximum plaintext buffer size for sending data to the peer. This value is set by the implementation after the SASL authentication phase has been completed and a SASL security layer is in effect.

SEE ALSO

Authen::SASL, Authen::SASL::Perl

AUTHORS

Written by Simon Wilkinson, with patches and extensions by Achim Grolms and Peter Marschall.

Please report any bugs, or post any suggestions, to the perl-ldap mailing list <perl-ldap@perl.org>

COPYRIGHT