

CommuniGate Pro and Open Directory: Kerberos and LDAP Integration

By Randy Saeks

Introduction

CommuniGate Pro is an E-Mail server provided by Stalker Software, Inc. In various network implementations, there may be instances when the same vendor providing directory services may not supply your email application. In such an environment, it is desirable to utilize a method to make remembering user names and passwords simpler. These disparate systems should be able to talk to each other using industry standards.

Apple utilizes Open Directory for their Directory Service. This is based on the openldap Directory Serivce. Using this as the user authentication backend, it is possible to utilize standards based protocols to tie the two systems together from the authentication standpoint.

This guide will focus on using Kerberos and LDAP to make email a simpler process from the users perspective in that they can utilize Single Sign On (SSO) for Application based access to email, or LDAP for Web-based access. These two protocols are discussed since it is conceivable that users will access email by a desktop mail application and a web-browser based on location. In having both methods setup, a more robust system can be obtained which is easier from the user perspective.

Before getting started, I would like to acknowledge those that helped me implement this setup.

The first person is David Colville; I utilized his steps for using the LDAP integration piece.

The second is Jason Mader. He provided the guidance on the Kerberos integration.

Requirements

For this process, you will need to have an LDAP server running Open Directory, as well as a Kerberos KDC. Your Open Directory Master will fill this role. In addition to the Directory Services, you will also need a CommuniGate Pro server. Furthermore, root access to the server acting as the KDC will be needed, as well as a user account in CommuniGate Pro that is at the administrative level.

The software requirements are as follows:

- XCode Tools v2.4.1

Kerberos Setup

In order to allow CommuniGate to talk to the Kerberos KDC, you will need to add service principals to the KDC. In addition, you will need to import the keytab file to the CGP Server so it knows to accept the service tickets. Since Open Directory uses a MIT Kerberos based setup, this process is relatively straightforward.

In order to complete this process, you will need to be logged into the computer using an admin account to which you can obtain root rights with.

1. Add Service Principals to the KDC. Open the kadmin.local process with the following command:

   ```
   sudo kadmin.local
   ```

2. Once you are in the kadmin.local process, you will need to add the service principals to the KDC. This will allow the specified services on the host specified to participate in the Kerberos realm. You can add the principals with the following command:

   ```
   addprinc -randkey smtp/mail.domain.edu
   addprinc -randkey imap/mail.domain.edu
   addprinc -randkey pop/mail.domain.edu
   ```

3. The above command will create SMTP, IMAP, and POP service principals for the specified server mail.domain.edu. If you are not utilizing imap, you do not need to add these service principals. After issuing these commands, you should see an entry after each along the lines of:

   ```
   WARNING: no policy specified for pop/mail.domain.edu@YOUR_REALM; defaulting to no policy
   Principal "pop/mail.domain.edu@YOUR_REALM" created.
   ```

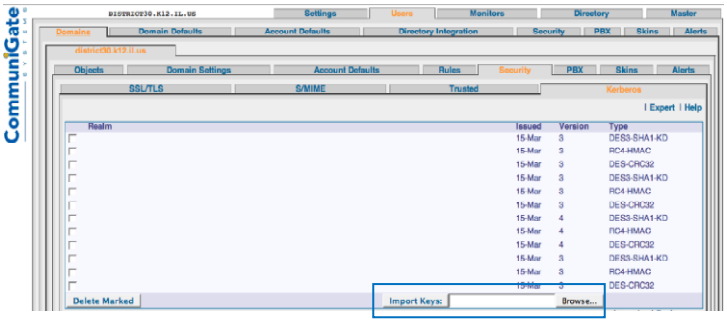
   The important part to note is that the Principal was created. The warning message is displayed since no policy has been set for the principal, and will not prevent this process from working.

4. One you have created the principals in the KDC, you will need to export the keytab to import into the CGP server. Do so with:

   ```
   ktadd -k /path/to/export/whatever.krb smtp/mail.domain.edu imap/mail.domain.edu
   ```

   When exporting the keytab, specify each principal you will be exporting in the command. Note that a space separates the principals.

5. Next, you will need copy the keytab to your CGP server. To import the keytabs, Navigate via the WebAdmin interface to: Domain > Security > Kerberos. At the bottom of the screen click the browse button, and navigate to the location of your keytab. Select the file, and click on Import Keys.



6. Once imported, you should see a listing of your Kerberos realm, and a list of the service principals that the Communigate Server will accept.

7. Finally, enable Kerberos Authentication, if it is not already, either with a user account (good for testing) or via the Account Defaults Settings. The latter will effect all accounts in the Domain. Make sure your clients are bound to the Kerberos realm, and the mail application is set to use Kerberos Authentication.

LDAP Setup

The LDAP setup of the server is utilized to allow users who access email via the Web Interface the ability to authenticate using their Directory Service password. This helps ease the login process as one password can be utilized for both server and email access.

To complete this process, you will need to install the Apple Developer Tools, as this method utilizes a perl script and some additional command line tools.

1. Install the Apple Developer Tools on your Communigate Server. You can obtain these from http://connect.apple.com. You will need to login, and creating a basic account is free. The current version of the tools is 2.4.1, and is a hefty 930MB download. When installing, you can customize the install to remove Developer Documentation.

2. Once you have the Developer Tools installed, open the Terminal Application and enter into the CPAN command line. This is a way to obtain Perl Modules. To enter the CPAN console, type in:

```
sudo cpan
```

3. Install three CPAN modules with the following command:

```
install IO::Socket
install IO::Socket::SSL
install Net::LDAP
```

   If prompted, install the dependencies needed for these packages to run.

4. Download the authLDAP.pl script from Stalker at: http://www.stalker.com/CGAUTH/. I have included a modified version at the end of this document that you can copy and paste to use as a reference as a few changes had to be made. (Changes noted in blue)

5. Put authLDAP.pl into CommuniGate Pro's folder which by default is /var/CommuniGate/ on OS X and give the script execute permission.

6. In CommuniGate Pro's web admin interface, set the External Authentication option to use this script. You can find this setting in: Settings >General >Helpers.

7. Start and stop the CommuniGate Pro process by issuing the following command in Terminal:

```
/System/Library/StartupItems/CommuniGatePro/CommuniGatePro stop
/System/Library/StartupItems/CommuniGatePro/CommuniGatePro start
```

8. Finally, set a user account to use the External Password as opposed to the CommuniGate Pro password. Under the CommuniGate Pro web administration go to Accounts > and select an account. Set the option for the CommuniGate Password to NO and set External Authentication to YES.

   ** Note, for this to work, your CommuniGate Pro username needs to match your Open Directory Username. As an example, bsample needs to exist in both locations.

9. After testing, you can change your domain defaults to Disable the CommuniGate password for everyone, and Enable the External Password.

10. In addition, if you wish to use the Pronto! Engine, you will need to disable the CRAM-MD5 Login Method. This will disable the SASL bind for Pronto!. This can be found in the Domain Settings.

```
#!/usr/bin/perl

#  Sample External Authenticaton program for CommuniGate Pro
#  that employs LDAP "bind".
#
#  See for more info:
#  <http://www.stalker.com/CommuniGatePro/Security.html#External>


#  You may need to install the following modules:
#  ASN1 from <http://www.cpan.org/modules/by-module/Convert/>
#  LDAP from <http://www.cpan.org/modules/by-module/Net/>

use Net::LDAP;

my $LDAPServerAddress =  '';   # You should define this value - LDAP Server Address
my $LDAPSearchBase = '';       # You should define this value - LDAP Server context for users

$| = 1;     #force STDOUT autoflush after each write


while(<STDIN>) {
  chomp;    # remove \n from the end of line
  my ($prefix,$method,$mode,$name,$password) = split(/ /);

  unless($mode =~ /^\(.*\)$/) {
    $password=$name; $name=$mode;
  }

  unless($prefix && $method && $name && $password) {
    print "$prefix ERROR Expected: nnn VRFY user\@domain password\n";

  } elsif($method ne 'VRFY') {
    print "$prefix ERROR Only plain text passwords supported\n";

  } else {
    my $errorMsg=checkPassword($name,$password);
    if(defined $errorMsg) {
      print "$prefix ERROR $errorMsg\n";
    } else {
      print "$prefix OK\n";
    }

  }
}

sub checkPassword() {
  my ($user,$password)=@_;


  my $name,$domain="";
  if($user =~ /(.+)\@(.+)/) {
    $name=$1;
    $domain=$2;
  } else {
    return "Full account name with \@ and domain part expected";
  }

  my $ldap = Net::LDAP->new($LDAPServerAddress,port=>389,timeout=>20)
    || return "Can't connect to $LDAPServerAddress via LDAP";

  my $result=$ldap->bind("uid=$name,$LDAPSearchBase",password=>$password)
    || return "Can't bind";

  $ldap->unbind();                        # unbind & disconnect

  $result->code && return $result->error; # return error message if failed
  return undef;                           # return "undef" on success
}
```