Testing LDAP

Adrian Kosmaczewski

2021-07-09

Testing applications that use LDAP for user authentication can be complicated. You just cannot use the production LDAP in your testing, because… reasons, so it can be difficult to make sure your application works properly before putting it in production.

Here go two simple tricks I've found to help me in those cases; I've used both and they work great.

Using Docker

This Docker container image exposes an LDAP server in port 10389, ready to be used. Launch it with the common `docker run` command.

```
$ docker run --detach --rm --publish 10389:10389 rroemhild/test-openldap
```

Here's the configuration JSON you need for your typical Express application using Passport.js for authentication:

```
const ldapConfig = {
  server: {
    url: 'ldap://localhost:10389',
    bindDN: 'cn=admin,dc=planetexpress,dc=com',
    bindCredentials: 'GoodNewsEveryone',
    searchBase: 'dc=planetexpress,dc=com',
    searchFilter: '(uid={{username}})'
  }
}
```

After connecting, just use these username/password combinations to authenticate in your application: `professor/professor`, `fry/fry`, `hermes/hermes` or `leela/leela`.

This can also work very well on a local Kubernetes cluster, of course, using Minikube or K3d.

You can get even fancier exposing your service over an ngrok TCP bridge to make it available over the Internet (requires a paid ngrok account, though, but it's totally worth it.)

Online

If for some reason you can't use Docker, here's an online LDAP server ready to use for testing purposes.

The corresponding JSON configuration would be the following:

```
const ldapConfig = {
  server: {
    url: 'ldap://ldap.forumsys.com:389',
    bindDN: 'cn=read-only-admin,dc=example,dc=com',
    bindCredentials: 'password',
    searchBase: 'dc=example,dc=com',
    searchFilter: '(uid={{username}})'
  }
}
```

And now you can use names of scientists to connect: `euler/password`, `newton/password`, `tesla/password` and you're in.