Redmine - Defect #2229

The presence of certain fields in LDAP responses breaks the LDAP parser

2008-11-25 23:49 - Tyler Bletsch

Status:	Closed	**Start date:**	2008-11-25
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	LDAP	**Estimated time:**	0.00 hour
Target version:			
Resolution:		**Affected version:**	0.7.3

Description

I am not an LDAP expert, so I'll do my best to explain this as I understand it.

I have Redmine installed & configured to authenticate against our corporate LDAP server. This is a virtual LDAP server (Oracle Virtual Directory) -- it simply mirrors content of a real LDAP cluster (RedHat LDAP). Configuring Redmine to authenticate directly against one of the real LDAP servers works, but pointing it at the virtual server causes login attempts to throw an exception:

```
Net::BER::BerError (unsupported object type: class=context_specific, encoding=primitive, tag=7):
    /vendor/plugins/ruby-net-ldap-0.0.4/lib/net/ber.rb:117:in `read_ber'
    /vendor/plugins/ruby-net-ldap-0.0.4/lib/net/ber.rb:112:in `read_ber'
    /vendor/plugins/ruby-net-ldap-0.0.4/lib/net/ldap.rb:1104:in `bind'
    /vendor/plugins/ruby-net-ldap-0.0.4/lib/net/ldap.rb:639:in `search'
    /app/models/auth_source_ldap.rb:37:in `authenticate'
    /vendor/rails/activerecord/lib/active_record/associations/association_proxy.rb:125:in `send'
    /vendor/rails/activerecord/lib/active_record/associations/association_proxy.rb:125:in `method_
missing'
    /app/models/user.rb:97:in `try_to_login'
    /app/controllers/account_controller.rb:46:in `login'
    /vendor/rails/actionpack/lib/action_controller/base.rb:1158:in `send'
    /vendor/rails/actionpack/lib/action_controller/base.rb:1158:in `perform_action_without_filters
'
    /vendor/rails/actionpack/lib/action_controller/filters.rb:697:in `call_filters'
    /vendor/rails/actionpack/lib/action_controller/filters.rb:689:in `perform_action_without_bench
mark'
    /vendor/rails/actionpack/lib/action_controller/benchmarking.rb:68:in `perform_action_without_r
escue'
    /usr/lib/ruby/1.8/benchmark.rb:293:in `measure'
    /vendor/rails/actionpack/lib/action_controller/benchmarking.rb:68:in `perform_action_without_r
escue'
    /vendor/rails/actionpack/lib/action_controller/rescue.rb:199:in `perform_action_without_cachin
g'
    /vendor/rails/actionpack/lib/action_controller/caching.rb:678:in `perform_action'
    /vendor/rails/activerecord/lib/active_record/connection_adapters/abstract/query_cache.rb:33:in
 `cache'
    /vendor/rails/activerecord/lib/active_record/query_cache.rb:8:in `cache'
    /vendor/rails/actionpack/lib/action_controller/caching.rb:677:in `perform_action'
    /vendor/rails/actionpack/lib/action_controller/base.rb:524:in `send'
    /vendor/rails/actionpack/lib/action_controller/base.rb:524:in `process_without_filters'
    /vendor/rails/actionpack/lib/action_controller/filters.rb:685:in `process_without_session_mana
gement_support'
    /vendor/rails/actionpack/lib/action_controller/session_management.rb:123:in `process'
    /vendor/rails/actionpack/lib/action_controller/base.rb:388:in `process'
    /vendor/rails/actionpack/lib/action_controller/dispatcher.rb:171:in `handle_request'
    /vendor/rails/actionpack/lib/action_controller/dispatcher.rb:115:in `dispatch'
    /vendor/rails/actionpack/lib/action_controller/dispatcher.rb:126:in `dispatch_cgi'
    /vendor/rails/actionpack/lib/action_controller/dispatcher.rb:9:in `dispatch'
    /vendor/rails/railties/lib/webrick_server.rb:112:in `handle_dispatch'
    /vendor/rails/railties/lib/webrick_server.rb:78:in `service'
    /usr/lib/ruby/1.8/webrick/httpserver.rb:104:in `service'
    /usr/lib/ruby/1.8/webrick/httpserver.rb:65:in `run'
    /usr/lib/ruby/1.8/webrick/server.rb:173:in `start_thread'
    /usr/lib/ruby/1.8/webrick/server.rb:162:in `start'
    /usr/lib/ruby/1.8/webrick/server.rb:162:in `start_thread'
    /usr/lib/ruby/1.8/webrick/server.rb:95:in `start'
    /usr/lib/ruby/1.8/webrick/server.rb:92:in `each'
    /usr/lib/ruby/1.8/webrick/server.rb:92:in `start'
    /usr/lib/ruby/1.8/webrick/server.rb:37:in `start'
    /usr/lib/ruby/1.8/webrick/server.rb:82:in `start'
    /vendor/rails/railties/lib/webrick_server.rb:62:in `dispatch'
    /vendor/rails/railties/lib/commands/servers/webrick.rb:66
    /usr/lib/ruby/site_ruby/1.8/rubygems/custom_require.rb:27:in `gem_original_require'
    /usr/lib/ruby/site_ruby/1.8/rubygems/custom_require.rb:27:in `require'
    /vendor/rails/activesupport/lib/active_support/dependencies.rb:496:in `require'
    /vendor/rails/activesupport/lib/active_support/dependencies.rb:342:in `new_constants_in'
    /vendor/rails/activesupport/lib/active_support/dependencies.rb:496:in `require'
    /vendor/rails/railties/lib/commands/server.rb:39
    /script/server:3:in `require'
    /script/server:3
```

I've attached a Wireshark trace of both the working (real LDAP) and problematic (virtual LDAP) bindResponse packet. Looking at a problematic trace, we see that the virtual LDAP server adds an additional field to the bindResponse: "serverSaslCreds: <MISSING>". This field corresponds to the "87 00" part of the hex dump (0x87=135 is the id, 00 is the (null) body). This id corresponds to the class, encoding, and tag listed in the exception. I believe that Redmine's low-level LDAP handler is flawed, in that this is a valid (though irrelevant) field that causes the parser itself to break. I have a hack to bypass this problem -- directly after the variable 'id' is assigned, I add:

```
return nil if (id == 135) # hack: ignore the 0-length context_specific primitive tag=7 field calle
d "serverSaslCreds"
```

Obviously, this only hides one specific instance of the problem, but it's enough for Redmine to work for me. The proper solution is to support this general *type* of field.

Version details:

- Platform: RHEL 5.2
- Database: MySQL 14.12 Distrib 5.0.45, for redhat-linux-gnu (i686) using readline 5.0
- Ruby: 1.8.5 (2006-08-25) [i386-linux]
- Rails: 2.1.1

History

#1 - 2010-02-18 01:33 - Eric Davis

- Category changed from Accounts / authentication to LDAP

#2 - 2011-04-06 22:03 - Tyler Bletsch

I just wanted to check in and report that this bug is still around. I did a fresh install of Redmine (1.1.2.stable) in the same environment, had the same problem, and bypassed it by inserting the same line into the same file. It looks like the LDAP module shipped with Redmine hasn't been updated -- it's still 0.0.4.

#3 - 2011-04-07 09:40 - Etienne Massip

This seems to have been fixed in rubyb-net-ldap version > 0.1.1 (dunno which version exactly).

https://rubyforge.org/tracker/index.php?func=detail&aid=18962&group_id=143&atid=633

We could consider upgrading embedded ldap adapter to latest version (0.2.2), but it requires ruby >= 1.8.7 since of 0.1.0 and latest Redmine is still supporting 1.8.6.

Maybe with 1.2.0 if we change Redmine requirement to Ruby = 1.8.7 ?

#4 - 2011-08-16 20:47 - Tyler Bletsch

Etienne Massip wrote:

> Maybe with 1.2.0 if we change Redmine requirement to Ruby = 1.8.7 ?

FYI, the LDAP module v0.0.4 is still shipping with Redmine 1.2.1, so this bug remains. I know because a package update knocked out my hack and broke our Redmine install. What would it take to include the updated LDAP module with the next point release? Etienne mentioned that this means dropping Ruby 1.8.6 support...is this still a problem?

It seems that RHEL/CentOS 5 is stuck on ruby 1.8.5, so is already out of support, but RHEL/CentOS 6 is 1.8.7, so we're good there. Ubuntu 10.04 LTS is 1.8.7, and that's over a year old. I'm not a ruby expert, but it looks to me like dropping Ruby 1.8.6 compatibility might be acceptable now, especially with RVM out there.

#5 - 2013-05-11 03:19 - Jean-Baptiste Barth

- Status changed from New to Closed

It's been a while, there's no embedded net-ldap version anymore, and Redmine relies on net-ldap v0.3.1 at least, so this bug should be fixed. I close this issue.

Files

Snap-20081125-174855.png	12 KB	2008-11-25	Tyler Bletsch
Snap-20081125-174904.png	11.3 KB	2008-11-25	Tyler Bletsch