Authentic 2 - Development #56115

LDAP Backend : check_group_to_role_mappings / get_groups_dns ne supportent pas le cas ou group_filter n'est pas défini

11 août 2021 12:18 - Benjamin Renard

Statut:	Fermé	**Début:**	11 août 2021
Priorité:	Normal	**Echéance:**	
Assigné à:	Valentin Deniaud	**% réalisé:**	0%
Catégorie:	LDAP	**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Nous avons des installations ou le mapping users/groups se fait uniquement sur les valeurs de l'attribut *memberof* (pas de recherches de groupes via *group_filter*) et du coup, nous désactivons la recherche de groupes via *group_filter* pour éviter des recherches inutiles et chronophages (*group_filter=""*). Jusqu'ici, nous avions aucun problème avec ce cas de figure, mais le commit suivant introduit un problème :

```
commit 3cdd9e7d29320796fff4fb3c5460cce5a1020eb3
Author: Serghei Mihai <smihai@entrouvert.com>
Date:   Mon Feb 15 14:32:38 2021 +0100

    ldap: log missing group dn when mapped to a role (#50928)

diff --git a/src/authentic2/backends/ldap_backend.py b/src/authentic2/backends/ldap_backend.py
index 28f212e3..1c1c74ce 100644
--- a/src/authentic2/backends/ldap_backend.py
+++ b/src/authentic2/backends/ldap_backend.py
@@ -589,6 +589,27 @@ class LDAPBackend(object):
                 message = str(vars(c))
             log.info('ldap: bind error with authz_id "%s" -> "%s"', authz_id, message)

+    @classmethod
+    def check_group_to_role_mappings(cls, block):
+        group_to_role_mapping = block.get('group_to_role_mapping')
+        if not group_to_role_mapping:
+            return
+        for conn in cls.get_connections(block):
+            existing_groups = cls.get_groups_dns(conn, block)
+            for group_dn, role_slugs in group_to_role_mapping:
+                if group_dn in existing_groups:
+                    continue
+                log.warning('ldap: unknown group "%s" mapped to a role', group_dn)
+
+    @classmethod
+    def get_groups_dns(cls, conn, block):
+        group_base_dn = block['group_basedn'] or block['basedn']
+        # 1.1 is special attribute meaning, "no attribute requested"
+        results = conn.search_s(group_base_dn, ldap.SCOPE_SUBTREE,
+                                block['group_filter'], ['1.1'])
+        results = cls.normalize_ldap_results(results)
+        return set([group_dn for group_dn, attrs in results])
+
     def authenticate(self, request=None, username=None, password=None, realm=None, ou=None):
         if username is None or password is None:
             return None
@@ -1310,6 +1331,7 @@ class LDAPBackend(object):
             if conn is None:
                 logger.warning(u'unable to synchronize with LDAP servers %s', force_text(block['u
rl']))
                 continue
+            cls.check_group_to_role_mappings(block)
```

```
            user_basedn = force_text(block.get('user_basedn') or block['basedn'])
            user_filter = force_text(block['sync_ldap_users_filter'] or block['user_filter'])
            user_filter = user_filter.replace('%s', '*')
[...]
```

Les méthodes *check_group_to_role_mappings* et *get_groups_dns* ne vérifie pas que *group_filter* est défini avant de l'utiliser ce qui fait planter la synchronisation :

```
Traceback (most recent call last):
  File "/usr/lib/authentic2/manage.py", line 20, in <module>
    execute_from_command_line(sys.argv[:1] + argv)
  File "/usr/lib/python3/dist-packages/django/core/management/__init__.py", line 364, in execute_f
rom_command_line
    utility.execute()
  File "/usr/lib/python3/dist-packages/django/core/management/__init__.py", line 356, in execute
    self.fetch_command(subcommand).run_from_argv(self.argv)
  File "/usr/lib/python3/dist-packages/django/core/management/base.py", line 283, in run_from_argv
    self.execute(*args, **cmd_options)
  File "/usr/lib/python3/dist-packages/django/core/management/base.py", line 330, in execute
    output = self.handle(*args, **options)
  File "/usr/lib/python3/dist-packages/authentic2/management/commands/sync-ldap-users.py", line 34
, in handle
    for user in LDAPBackend.get_users():
  File "/usr/lib/python3/dist-packages/authentic2/backends/ldap_backend.py", line 1541, in get_use
rs
    cls.check_group_to_role_mappings(block)
  File "/usr/lib/python3/dist-packages/authentic2/backends/ldap_backend.py", line 690, in check_gr
oup_to_role_mappings
    existing_groups = cls.get_groups_dns(conn, block)
  File "/usr/lib/python3/dist-packages/authentic2/backends/ldap_backend.py", line 700, in get_grou
ps_dns
    results = conn.search_s(group_base_dn, ldap.SCOPE_SUBTREE, block['group_filter'], ['1.1'])
  File "/usr/lib/python3/dist-packages/ldap/ldapobject.py", line 852, in search_s
    return self.search_ext_s(base,scope,filterstr,attrlist,attrsonly,None,None,timeout=self.timeou
t)
  File "/usr/lib/python3/dist-packages/ldap/ldapobject.py", line 1259, in search_ext_s
    return self._apply_method_s(SimpleLDAPObject.search_ext_s,*args,**kwargs)
  File "/usr/lib/python3/dist-packages/ldap/ldapobject.py", line 1197, in _apply_method_s
    return func(self,*args,**kwargs)
  File "/usr/lib/python3/dist-packages/ldap/ldapobject.py", line 846, in search_ext_s
    return self.result(msgid,all=1,timeout=timeout)[1]
  File "/usr/lib/python3/dist-packages/ldap/ldapobject.py", line 738, in result
    resp_type, resp_data, resp_msgid = self.result2(msgid,all,timeout)
  File "/usr/lib/python3/dist-packages/ldap/ldapobject.py", line 742, in result2
    resp_type, resp_data, resp_msgid, resp_ctrls = self.result3(msgid,all,timeout)
  File "/usr/lib/python3/dist-packages/ldap/ldapobject.py", line 749, in result3
    resp_ctrl_classes=resp_ctrl_classes
  File "/usr/lib/python3/dist-packages/authentic2/backends/ldap_backend.py", line 174, in result4
    resp_ctrl_classes=resp_ctrl_classes,
  File "/usr/lib/python3/dist-packages/ldap/ldapobject.py", line 756, in result4
    ldap_result = self._ldap_call(self._l.result4,msgid,all,timeout,add_ctrls,add_intermediates,ad
d_extop)
  File "/usr/lib/python3/dist-packages/ldap/ldapobject.py", line 329, in _ldap_call
    reraise(exc_type, exc_value, exc_traceback)
  File "/usr/lib/python3/dist-packages/ldap/compat.py", line 44, in reraise
    raise exc_value
  File "/usr/lib/python3/dist-packages/ldap/ldapobject.py", line 313, in _ldap_call
    result = func(*args,**kwargs)
ldap.PROTOCOL_ERROR: {'desc': 'Protocol error'}
```

Je joins un hot-patch qui ajoute la gestion de ce cas.

Note : Au passage, pour désactiver la recherche via *group_filter*, on est obligé de mettre *group_filter=""* car on ne peut pas bêtement mettre *group_filter=None* qui génère une erreur :

```
[...]
  File "/usr/lib/python3/dist-packages/authentic2/backends/ldap_backend.py", line 1782, in update_
```

```
default
    raise ImproperlyConfigured('LDAP_AUTH_SETTINGS: attribute %r must be a string' % d)
django.core.exceptions.ImproperlyConfigured: LDAP_AUTH_SETTINGS: attribute 'group_filter' must be
a string
```

Révisions associées

Révision b8629710 - 30 août 2021 10:25 - Valentin Deniaud

ldap_backend: do not check group dns if there is no group filter (#56115)

Historique

#1 - 23 août 2021 17:30 - Valentin Deniaud

- Assigné à mis à Valentin Deniaud

#2 - 23 août 2021 17:30 - Valentin Deniaud

- Fichier 0001-ldap_backend-do-not-check-group-dns-if-there-is-no-g.patch ajouté

- Tracker changé de Support à Development

- Statut changé de Nouveau à Solution proposée

#3 - 23 août 2021 18:18 - Serghei Mihai

- Statut changé de Solution proposée à Solution validée

#4 - 30 août 2021 10:25 - Valentin Deniaud

- Statut changé de Solution validée à Résolu (à déployer)

```
commit 73aaee270ab31b87888e2d46395eb188be8633d0
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Mon Aug 23 17:28:45 2021 +0200

    ldap_backend: do not check group dns if there is no group filter (#56115)
```

#5 - 02 septembre 2021 22:17 - Frédéric Péters

- Statut changé de Résolu (à déployer) à Solution déployée

Fichiers

0001-Fix-handling-case-when-no-group_filter-is-configured.patch	1,43 ko	11 août 2021	Benjamin Renard
0001-ldap_backend-do-not-check-group-dns-if-there-is-no-g.patch	1001 octets	23 août 2021	Valentin Deniaud