LDAP-фильтры для поиска в Active Directory

windowsnotes.ru/activedirectory/ldap-filtry-dlya-poiska-obektov-v-active-directory-chast-2

17 января 2020 г.

Продолжаем разговор об использовании LDAP-фильтров в Active Directory, начатый в предыдущей статье. Сегодня речь пойдет о расширенных фильтрах поиска.

Начнем издалека.

Типы фильтров

LDAP-фильтры условно можно разделить на четыре базовых типа — simple, present, substring и extensible. Элемент simple формируется следующим образом:

(<Атрибут><оператор><значение>)

Например, все пользователи с именем Vasya:

(cn=Vasya)

Элемент present используется для отбора записей, у которых присутствует определённый атрибут и формируется так:

(<Атрибут>=*)

Например, все пользователи с заполненным описанием (Description):

(Description=*)

Элемент substring используется для отбора по подстрокам и формируется следующим образом:

(<Атрибут>=<Начальное значение>*<Конечное значение>)

Например, все пользователи с именем, начинающимся на V и оканчивающимся на a:

(cn=V*a)

Начальное и конечное значение вовсе не обязательны, например:

(cn=V*)

Расширенные фильтры

Вот теперь мы и добрались до расширенных (extensible) фильтров. Они составляются таким образом:

(<Атрибут>:<Правило соответствия>:=<Значение>)

Правило соответствия (Matching Rule) — правило, которое будет применено к искомому компоненту. Правила соответствия определены в RFC 2251 и представляют из себя средство выражения того, как сервер должен сравнивать значение, указанное в фильтре поиска с данными, полученными в результате поиска. Правило может определяться по имени или по OID.

К сожалению, реализация LDAP от Microsoft не поддерживает большинство расширяемых правил, описанных в RFC. Поэтому в Active Directory доступны всего три правила соответствия.

Matching rule OID	**String identifier**	**Описание**
1.2.840.113556.1.4.803	LDAP_MATCHING_RULE_BIT_AND	Совпадение будет найдено только в том случае, если все биты атрибута совпадут со значением. Это правило эквивалентно операции побитового И (bitwise **AND**)
1.2.840.113556.1.4.804	LDAP_MATCHING_RULE_BIT_OR	Совпадение будет найдено в том случае, если любые биты из атрибута совпадают со значением. Это правило эквивалентно операции побитового ИЛИ (bitwise **OR)**

Matching rule OID	String identifier	Описание
1.2.840.113556.1.4.1941	LDAP_MATCHING_RULE_IN_CHAIN	Специальный расширенный оператор соответствия, проходящий по цепи наследования к корню до тех пор, пока не найдет совпадение. Выявляет вложенность групп. Применяется только к DN-атрибутам. Доступен на контроллерах домена с Windows Server 2003 SP2 и более поздних версий.

Первые два правила представляют из себя операторы побитового сравнения. Дело в том, что значения некоторых атрибутов в AD хранятся в виде битовой маски (Bitmask). Например, атрибут Group-Type (тип группы), определяющий тип и область действия группы в AD, представляет из себя битовую маску со следующими значениями:

0x00000001 (1) — BuiltIn Group;
0x00000002 (2) — Global Group ;
0x00000004 (4) — Domain Local Group ;
0x00000008 (8) — Universal Group;
0x80000000 (2147483648) — Security Group.

При использовании правил синтаксис фильтра выглядит так:

```
<Атрибут>:<rule OID>:=<Десятичное значение>
```

К примеру, нам требуется отобрать все универсальные группы. Для этого необходимо с помощью правила LDAP_MATCHING_RULE_BIT_AND проверить в значении атрибута наличие четвертого бита (0x00000008 или 00001000 в двоичном виде). Для этого воспользуемся фильтром:

(groupType:1.2.840.113556.1.4.803:=8)

```
PS C:\> Get-ADGroup -LDAPFilter "(groupType:1.2.840.113556.1.4.803:=8)" |ft -a Name,GroupScope,GroupCategory

Name                                GroupScope GroupCategory
----                                ---------- -------------
Schema Admins                        Universal      Security
Enterprise Admins                    Universal      Security
Enterprise Read-only Domain Controllers  Universal      Security
Enterprise Key Admins                Universal      Security
BackOffice                           Universal      Security
Employees                            Universal  Distribution
```

Если условий несколько, то значения можно складывать. Так если необходимо отобрать все универсальные группы безопасности, то получим значение 2147483656 (2147483648 + 8), а фильтр будет такой:

(groupType:1.2.840.113556.1.4.803:=2147483656)

```
PS C:\> Get-ADGroup -LDAPFilter "(groupType:1.2.840.113556.1.4.803:=2147483656)" |ft -a Name,GroupScope,GroupCategory

Name                                GroupScope GroupCategory
----                                ---------- -------------
Schema Admins                        Universal      Security
Enterprise Admins                    Universal      Security
Enterprise Read-only Domain Controllers  Universal      Security
Enterprise Key Admins                Universal      Security
BackOffice                           Universal      Security
```

Можно пойти от обратного и отобрать группы, не являющиеся группами безопасности (т.е. все группы распространения):

(!(groupType:1.2.840.113556.1.4.803:=2147483648))

```
PS C:\> Get-ADGroup -LDAPFilter "(!(groupType:1.2.840.113556.1.4.803:=2147483648))" |ft -a Name,GroupScope,GroupCategory

Name      GroupScope GroupCategory
----      ---------- -------------
Employees  Universal  Distribution
```

Еще один пример битовой маски — это атрибут userAccountControl, в котором хранятся свойства пользователя. Вот некоторые из значений атрибута:

0x00000002 (2) — учетная запись отключена;
0x00000010 (16) — учетная запись заблокирована;
0x00000020 (32) — пароль не требуется;
0x00000040 (64) — запретить смену пароля пользователем;
0x00000200 (512) — учетная запись по умолчанию (учетная запись пользователя со всеми снятыми флагами);
0x00010000 (65536) — срок действия пароля не ограничен;
0x00800000 (8388608) — срок действия пароля истек.

Например, выведем все отключенные (Disabled) учетные записи фильтром:

(userAccountControl:1.2.840.113556.1.4.803:=2)

```
PS C:\> Get-ADUser -LDAPFilter "(userAccountControl:1.2.840.113556.1.4.803:=2)" |ft -a Name,Enabled

Name   Enabled
----   -------
Guest    False
krbtgt   False
ivan     False
```

При необходимости значения можно суммировать между собой. Для примера с помощью правила LDAP_MATCHING_RULE_BIT_OR найдем всех пользователей, у которых пароль никогда не истекает или не требуется вообще. Значение атрибута при этом будет равно 65568 (65536 + 32):

(userAccountControl:1.2.840.113556.1.4.804:=65568)

```
PS C:\> Get-ADUser -LDAPFilter "(userAccountControl:1.2.840.113556.1.4.804:=65568)" -Properties * |ft -a name,pass*

name          PasswordExpired PasswordLastSet       PasswordNeverExpires PasswordNotRequired
----          --------------- ---------------       -------------------- -------------------
Administrator           False 10/29/2019 6:33:37 PM                True               False
Guest                   False                      True                True
kirill                  False 12/24/2019 5:00:44 PM                True               False
yulia                   False 12/30/2019 4:41:52 PM                True               False
```

Правило LDAP_MATCHING_RULE_IN_CHAIN заслуживает особого внимания, поскольку с его помощью можно находить вложенные элементы. Предположим, нам требуется найти все группы, членом которых является пользователь. Для этого потребуется написать функцию, которая будет рекурсивно обходить все группы пользователя, включая вложенные. При этом необходимо учесть ситуацию, когда группа A вложена в группу B, а группа B вложена в группу A.

Так вот, правило LDAP_MATCHING_RULE_IN_CHAIN позволяет решить данную задачу без особых затрат. Для примера найдем все группы пользователя с помощью простого фильтра:

(member=CN=Vasya,OU=Employees,DC=test,DC=local)

А затем используя правило LDAP_MATCHING_RULE_IN_CHAIN:

(member:1.2.840.113556.1.4.1941:=CN=Vasya,OU=Employees,DC=test,DC=local)

Как видите, второй фильтр нашел группу, которую пропустил первый.

```
PS C:\> Get-ADGroup -LDAPFilter "(member=CN=Vasya,OU=Employees,DC=test,DC=local)" | ft -a

DistinguishedName                     GroupCategory GroupScope Name      ObjectClass ObjectGUID
-----------------                     ------------- ---------- ----      ----------- ----------
CN=Employees,CN=Users,DC=test,DC=local      Security  Universal Employees group       828a7197-7061-4e03-a0df-600f16...
CN=Logistics,CN=Users,DC=test,DC=local      Security     Global Logistics group       97909bc0-d8a4-4641-a56b-c571dd...


PS C:\> Get-ADGroup -LDAPFilter "(member:1.2.840.113556.1.4.1941:=CN=Vasya,OU=Employees,DC=test,DC=local)" |ft -a

DistinguishedName                      GroupCategory GroupScope Name       ObjectClass ObjectGUID
-----------------                      ------------- ---------- ----       ----------- ----------
CN=BackOffice,CN=Users,DC=test,DC=local      Security  Universal BackOffice group       e3fa4956-cc94-4620-beae-ffe0...
CN=Employees,CN=Users,DC=test,DC=local       Security  Universal Employees  group       828a7197-7061-4e03-a0df-600f...
CN=Logistics,CN=Users,DC=test,DC=local       Security     Global Logistics  group       97909bc0-d8a4-4641-a56b-c571...
```

Теперь возьмем группу и тоже применим к ней два фильтра. Первый фильтр показывает только непосредственных членов группы:

(memberOf=CN=BackOffice,CN=Users,DC=test,DC=local)

А вот второй, в котором используется правило, покажет всех членов группы, включая вложенных:

(memberOf:1.2.840.113556.1.4.1941:=CN=BackOffice,CN=Users,DC=test,DC=local)

```
PS C:\> Get-ADUser -LDAPFilter "(memberOf=CN=BackOffice,CN=Users,DC=test,DC=local)" |ft -a
PS C:\> Get-ADUser -LDAPFilter "(memberOf:1.2.840.113556.1.4.1941:=CN=BackOffice,CN=Users,DC=test,DC=local)" |ft -a

DistinguishedName                     Enabled GivenName Name  ObjectClass ObjectGUID                           SamAcco
                                                                                                               untName
-----------------                     ------- --------- ----  ----------- ----------                           -------
CN=Vasya,OU=Employees,DC=test,DC=local    True Vasya     Vasya user        c3f5f243-7d80-469e-9158-7174ea8324ea vasya
```

Фильтрация с помощью ANR

Неоднозначное разрешение имен (Ambiguous Name Resolution, ANR) — алгоритм поиска объектов в Active Directory. ANR позволяет задавать сложные фильтры, включающие в себя несколько атрибутов, связанных с именами, причем в одном выражении. Его можно использовать в ситуации, когда вы знаете одно из имен объекта, но не уверены, в каком именно атрибуте это имя присутствует.

ANR включает в поиск следующий список атрибутов:

- displayName
- givenName (First Name)
- sn (Last Name)
- sAMAccountName
- legacyExchangeDN
- Relative Distinguished Name (RDN)
- physicalDeliveryOfficeName
- proxyAddresses
- mail
- mailNickname
- msExchResourceSearchProperties
- msDS-AdditionalSamAccountName
- msDS-PhoneticCompanyName
- msDS-PhoneticDepartment
- msDS-PhoneticDisplayName
- msDS-PhoneticFirstName
- msDS-PhoneticLastName

Например, нам нужно найти пользователя с фамилией Пупкин. В поисках нам поможет вот такой несложный фильтр:

(anr = Пупкин)

В этом случае Active Directory будет искать все объекты, где любой из вышеописанных атрибутов начинается со строки «Пупкин». Если сконвертировать это в стандартный LDAP-фильтр, то получится что то типа такого:

(|(displayName=Пупкин*)(givenName=Пупкин*)(legacyExchangeDN=Пупкин)(physicalDeliveryOfficeName=Пупкин*)(proxyAddresses=Пупкин*)(Name=Пупкин*)(sAMAccountName=Пупкин*)(sn=Пупкин*))

```
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(anr=Пупкин)" -Properties * |fl *name


CanonicalName     : test.local/Employees/Vasya
DisplayName       : Василий Иванович Пупкин
DistinguishedName : CN=Vasya,OU=Employees,DC=test,DC=local
GivenName         : Василий
Name              : Vasya
OtherName         :
SamAccountName    : vasya
Surname           : Пупкин
UserPrincipalName : vasya@test.local

CanonicalName     : test.local/Users/Fedor
DisplayName       : Федор Пупкин
DistinguishedName : CN=Fedor,CN=Users,DC=test,DC=local
GivenName         : Федор
Name              : Fedor
OtherName         :
SamAccountName    : fpupkin
Surname           : Пупкин
UserPrincipalName : fpupkin@test.local

CanonicalName     : test.local/Users/Василий Сергеевич Пупкин
DisplayName       : Василий Сергеевич Пупкин
DistinguishedName : CN=Василий Сергеевич Пупкин,CN=Users,DC=test,DC=local
GivenName         : Василий
Name              : Василий Сергеевич Пупкин
OtherName         :
SamAccountName    : vspupkin
Surname           : Пупкин
UserPrincipalName : vspupkin@test.local
```

Если нам известно имя пользователя, к примеру Василий, то можно немного изменить фильтр:

(anr = Василий Пупкин)

В этом случае фильтр будет искать все объекты где любой из вышеописанных атрибутов начинается со строки «Василий Пупкин» плюс объекты, у которых (givenName=Василий*) и (sn=Пупкин*) или (givenName=Пупкин*) и (sn=Василий*). При этом алгоритм поиска учитывает наличие пробела в строке, разбивая ее на два значения.

```
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(anr=Василий Пупкин)" -Properties * |fl *name


CanonicalName     : test.local/Employees/Vasya
DisplayName       : Василий Иванович Пупкин
DistinguishedName : CN=Vasya,OU=Employees,DC=test,DC=local
GivenName         : Василий
Name              : Vasya
OtherName         :
SamAccountName    : vasya
Surname           : Пупкин
UserPrincipalName : vasya@test.local

CanonicalName     : test.local/Users/Василий Сергеевич Пупкин
DisplayName       : Василий Сергеевич Пупкин
DistinguishedName : CN=Василий Сергеевич Пупкин,CN=Users,DC=test,DC=local
GivenName         : Василий
Name              : Василий Сергеевич Пупкин
OtherName         :
SamAccountName    : vspupkin
Surname           : Пупкин
UserPrincipalName : vspupkin@test.local
```

Ну и если нам удалось узнать отчество пользователя, к примеру Иванович, тогда получится такой фильтр:

(anr = Василий Иванович Пупкин)

В этом случае фильтр будет искать объекты, у которых любой из атрибутов именования соответствует Василий Иванович Пупкин* плюс объекты, у которых (givenName=Василий*) и (sn=Иванович Пупкин*), плюс объекты с (givenName=Иванович Пупкин*) и (sn=Василий*). В этом случае алгоритм при разбиении строки учитывает только первый пробел.

```
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(anr=Василий Иванович Пупкин)" -Properties * |fl *name


CanonicalName     : test.local/Employees/Vasya
DisplayName       : Василий Иванович Пупкин
DistinguishedName : CN=Vasya,OU=Employees,DC=test,DC=local
GivenName         : Василий
Name              : Vasya
OtherName         :
SamAccountName    : vasya
Surname           : Пупкин
UserPrincipalName : vasya@test.local
```

При необходимости можно изменить поведение ANR и требовать поиск точного соответствия для любого из атрибутов. Для этого надо перед значением поставить дополнительный знак равенства. Например, этот фильтр выдаст всех пользователей, у которых хотя бы один атрибут из списка точно соответствует строке Василий Иванович:

(anr == Василий Иванович)

```
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(anr==Василий Иванович)" -Properties * |fl *name
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(anr=Василий Иванович)" -Properties * |fl *name


CanonicalName     : test.local/Employees/Vasya
DisplayName       : Василий Иванович Пупкин
DistinguishedName : CN=Vasya,OU=Employees,DC=test,DC=local
GivenName         : Василий
Name              : Vasya
OtherName         :
SamAccountName    : vasya
Surname           : Пупкин
UserPrincipalName : vasya@test.local
```

Как видите, LDAP-фильтры имеют много различных возможностей. О том, как их правильно использовать, пойдет речь в следующей статье.