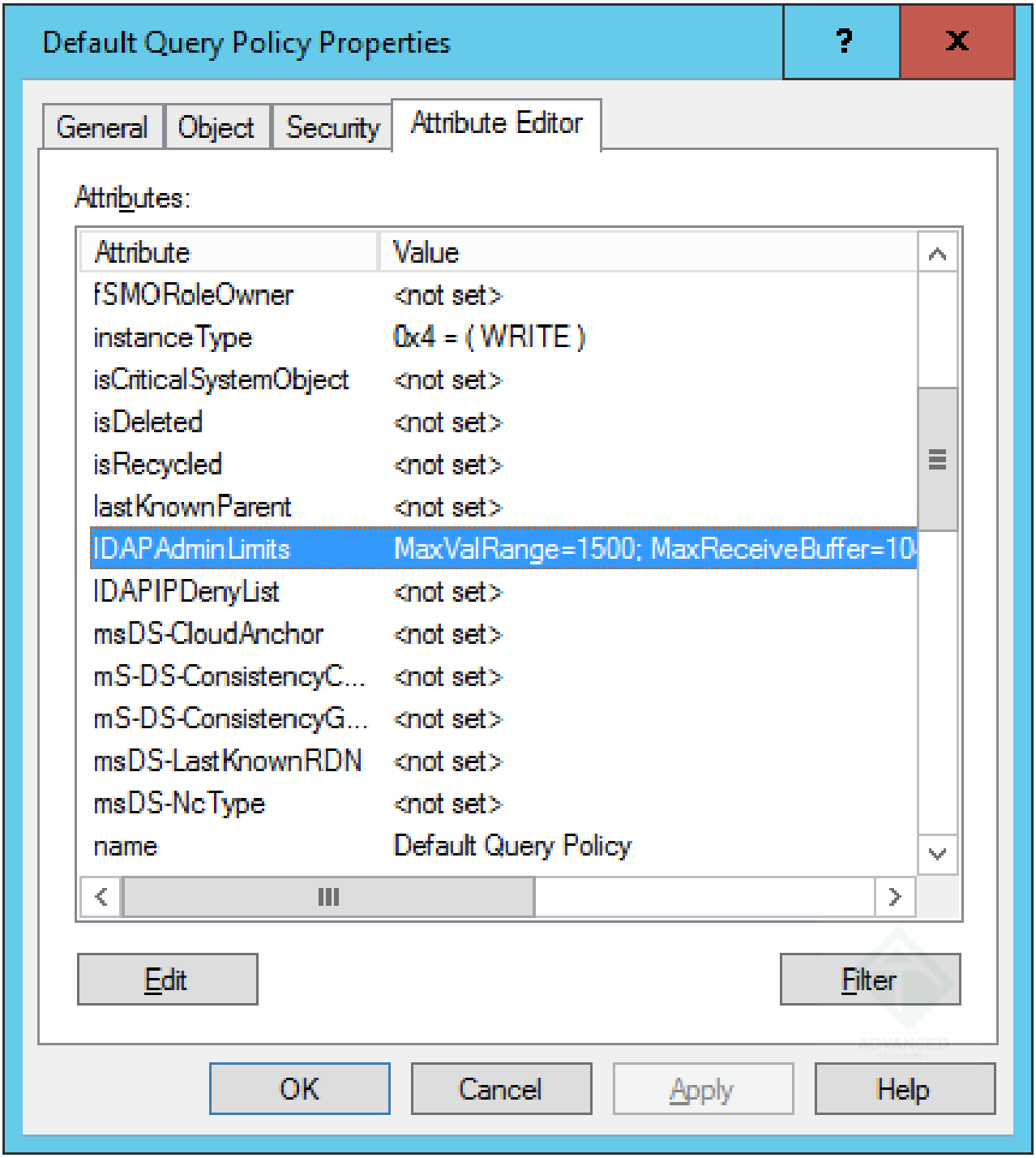
Повышение скорости, надёжности и безопасности функционирования DC и GC - всё это можно сделать через политики LDAP

atraining.ru/ldap-policy-active-directory

2015-08-23T07:12:52+08:00



Привет.

Я часто пишу про тюнинг Active Directory – данная тема интересна тем, что практически на каждом предприятии данная инфраструктурная служба есть, и в большинстве случаев из её возможностей используется малая толика. В данной

статье я расскажу про редкий и не освещаемый в простеньких авторизованных курсах функционал – **LDAP-политики** или **Query Policy**. Причина забвения данной темы, которая существует с Windows 2000 Server, тривиальна – и без настройки LDAP-политик всё работает с Default Query Policy. Однако в высоконагруженных или требующих взаимодействия с внешними LDAP-сервисами применениях знать, как работают LDAP-политики – надо.

Я предполагаю, что вы знаете, что такое CN, DN, RDN, LDAP, DC, GC, не путаете сайты Active Directory и то, что в IE, а также прониклись Active Directory настолько, что браузер для вас – это вначале ADSI.msc и только после – всё остальное. Наличие знаний хотя бы на уровне MCSA Windows Server 2012 обязательно.

Сразу предупреждение – не надо сразу бежать и применять то, что тут написано, на практике не спланировав и не подумав. Ряд приведённых настроек при быстром и решительном применении могут превратить процесс тюнинга службы каталогов в доработку резюме на сайте хехе.ру. Подумайте, спланируйте, продумайте последовательное применение и аккуратно, понемногу проводите изменения.

Мы будем изучать все существующие в природе Query Policy – начиная с Windows 2000 Server и заканчивая доступной на данный момент версией Windows Server 2016.

LDAP-политики (Query Policy) в Active Directory

- Что такое политики LDAP в Active Directory и зачем они нужны
- Базовые операции с политиками
- Как узнать, какие политики Query Policy поддерживаются
- Настраиваем параметры фильтрации доступа к LDAP
- Параметры функционирования LDAP
- Настраиваем параметры функционирования LDAP в Windows Server 2016
- Формат Query Policy
- Параметры, существующие с Windows 2000 Server
    - Параметр Query Policy – MaxActiveQueries
    - Параметр Query Policy – InitRecvTimeout
    - Параметр Query Policy – MaxConnections
    - Параметр Query Policy – MaxConnIdleTime
    - Параметр Query Policy – MaxDatagramRecv
    - Параметр Query Policy – MaxNotificationPerConn
    - Параметр Query Policy – MaxPoolThreads
    - Параметр Query Policy – MaxReceiveBuffer
    - Параметр Query Policy – MaxPageSize
    - Параметр Query Policy – MaxQueryDuration
    - Параметр Query Policy – MaxResultSetSize
    - Параметр Query Policy – MaxTempTableSize
- Параметры, существующие с Windows Server 2003
    - Параметр Query Policy – MaxValRange

- Параметры, существующие с Windows Server 2008 R2
    - Параметр Query Policy – MaxResultSetsPerConn
    - Параметр Query Policy – MinResultSets
- Параметры, существующие с Windows Server 2012
    - Параметр Query Policy – MaxBatchReturnMessages
- Параметры, существующие с Windows Server 2016
    - Параметр Query Policy – MaxDirSyncDuration
- Отключаем жестко заданные лимиты настроек
- Создаём новую Query Policy

Начнём.

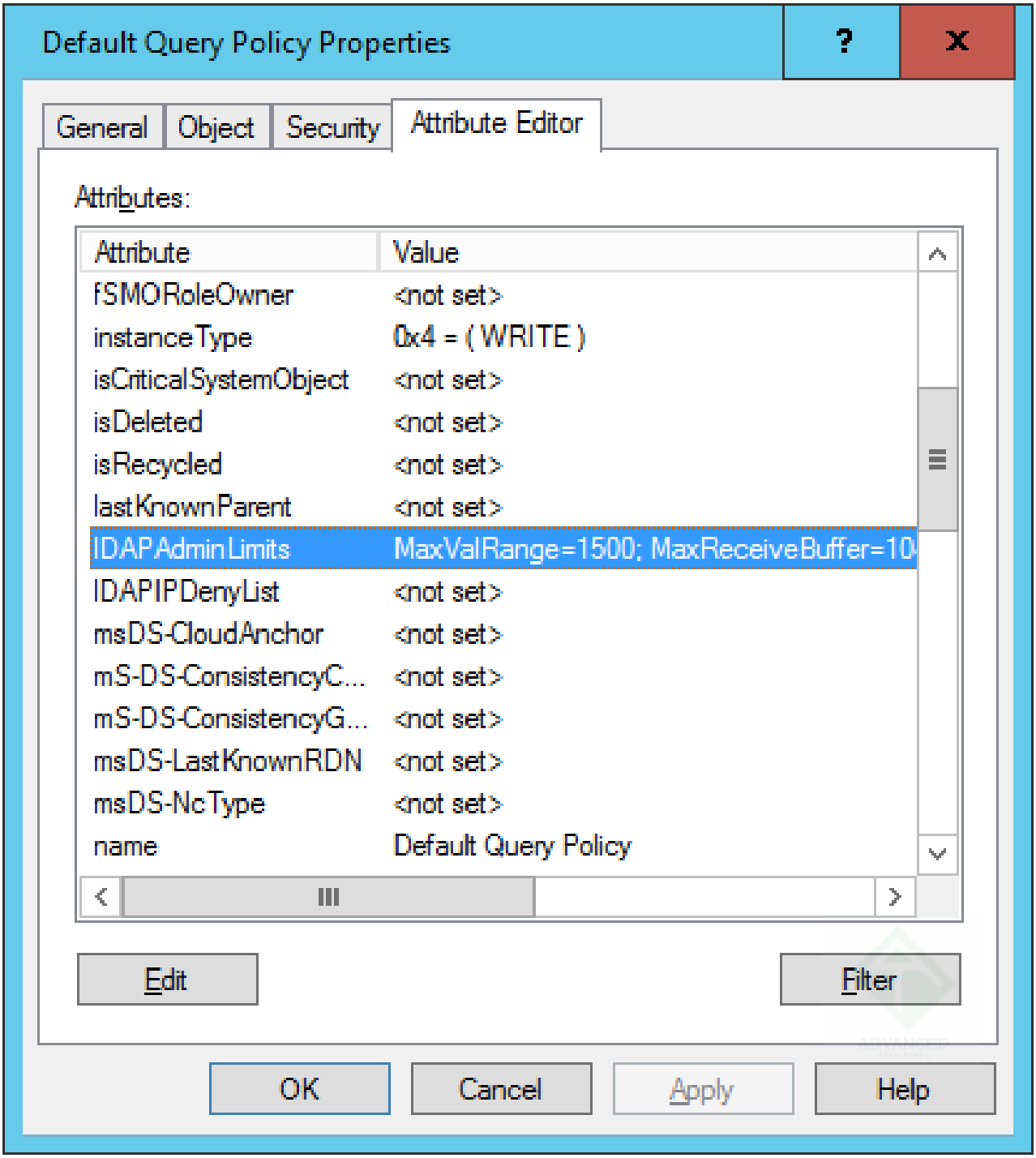
Что такое политики LDAP в Active Directory и зачем они нужны

Под термином “политики” в связи с Active Directory обычно имеются в виду групповые политики – мощное и легко расширяемое средство для централизованного управления многими настройками ОС и приложений. Некоторые ещё вспоминают “политики паролей” – PSO, которые появились с Windows Server 2008.

Однако, есть ещё одно применение термина “политики Active Directory” – благодаря объектам, которые относятся к классу `queryPolicy`, существует возможность тонкой настройки работы контроллеров домена (DC) и серверов глобального каталога (GC), а также ощутимого повышения скорости, надёжности и безопасности их функционирования. Все эти параметры будут относиться именно к клиентским запросам по LDAP и ограничивать размер ответа, занимаемую память, диапазон значений, тайм-ауты и другие подобные свойства. Давайте разберёмся что это, и как это можно (и нужно) эффективно использовать.

Базовые операции с политиками

Query Policy хранятся в специальных объектах класса `queryPolicy`. Сам класс `queryPolicy` представляет из себя объект, содержащий в себе “пачку” настроек, которые читаются LDAP-серверами. Пачка реализована как многострочные атрибуты `lDAPAdminLimits` и `lDAPIPDenyList`, которые можно редактировать вручную, прямо в объекте AD, а можно “по-красивому”, через утилиту `dsmgmt` (ранее – через `ntdsutil`).

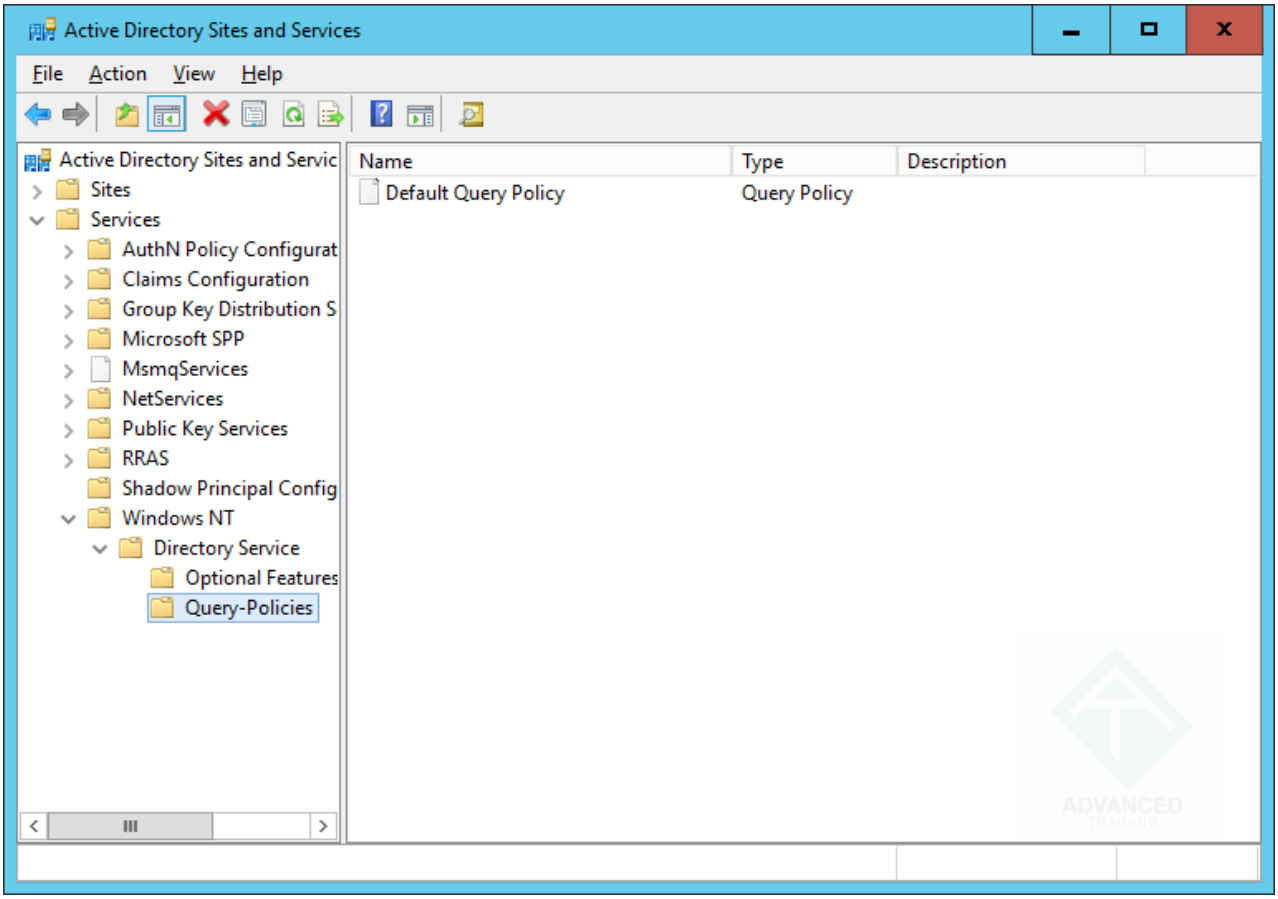


Стандартный объект Query Policy
(кликните для увеличения до 414 px на 462 px)

Утилита `dsmgmt` идеологически “более правильная” по причине, что `ntdsutil` изначально предназначался для работы именно с Active Directory, а dsmgmt позиционируется как более общее решение, которое должно работать со всеми службами LDAP-каталогов, включая AD LDS. А вообще, можно это всё править и через оснастку ADSI Editor. И через вкладку “Attributes” у объекта queryPolicy. И через древнюю утилиту ldp.exe. Особой разницы нет – LDAP – достаточно демократичная штука.

Объекты этого класса находятся в специальном контейнере, который расположен в разделе леса Configuration по DN-адресу `CN=Query-Policies,CN=Directory Service,CN=Windows NT,CN=Services,CN=Configuration,DC=`*`контекст-Вашего-леса-`*

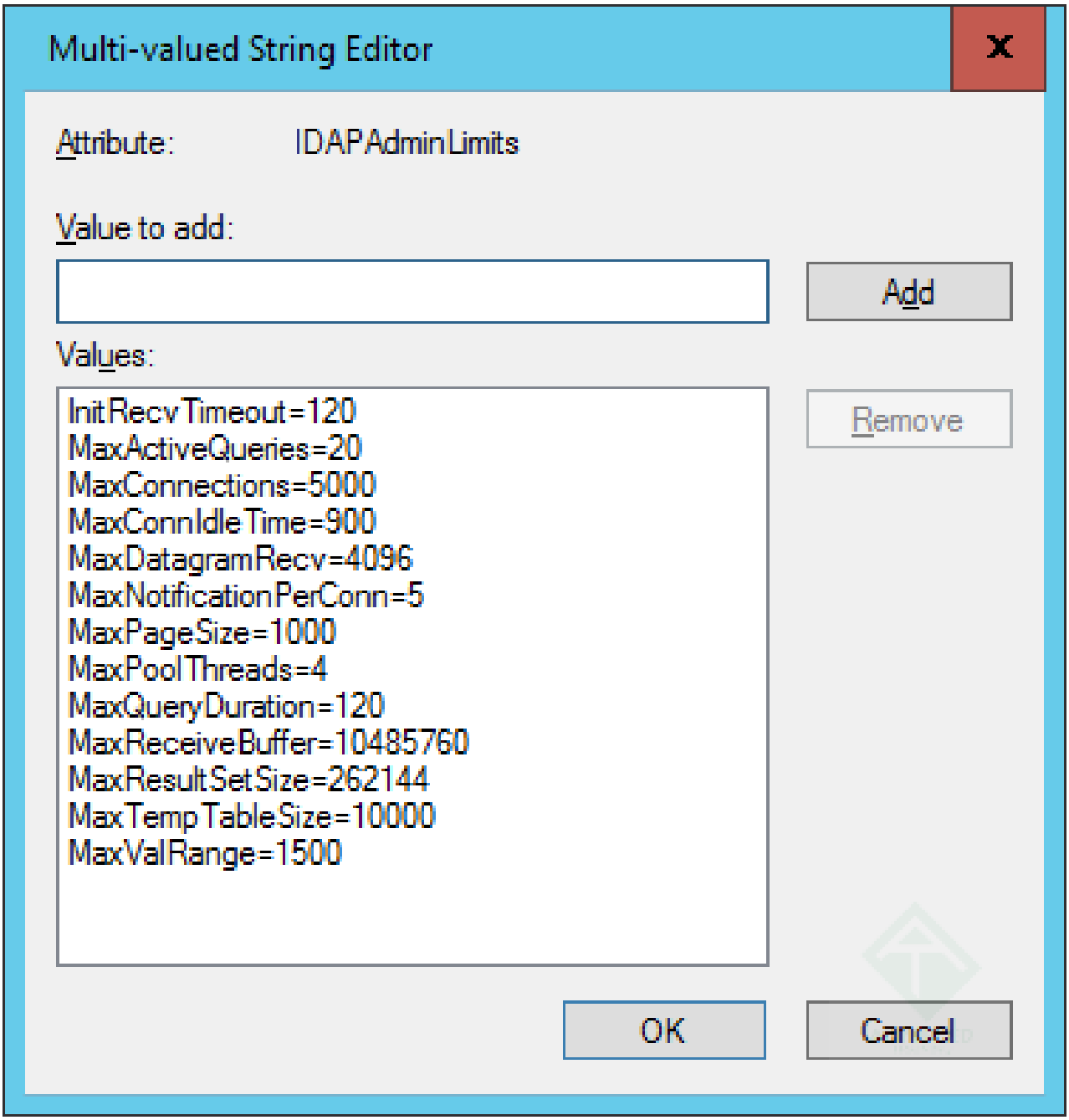
`Active-Directory`. Вы можете увидеть их глазами, если откроете стандартную оснастку Active Directory Sites and Services и не забудете включить отображение ветки лесных сервисов – View / Show Services Node:



Где в лесу Active Directory расположено хранилище Query Policy (кликните для увеличения до 768 px на 537 px)

Достаточно логично, что эти объекты находятся в разделе configuration, который реплицируется на все контроллеры в лесу. Ведь данные политики применяются не только на контроллеры конкретного домена, а и на GC, и на сайты, которые привязки к доменам не имеют.

Изначально в Active Directory создан единственный объект LDAP-политик с именем Default Query Policy, который содержит настройки “по умолчанию”. Вы его видели на первом скриншоте – а вот так выглядят его настройки для новорожденного DC на Windows Server 2016:



Настройки Default Query Policy
(кликните для увеличения до 376 px на 393 px)

Этот объект надо никогда не трогать. Почему – будет видно далее, а теперь мы поговорим о том, как эти объекты применяются на свою “целевую аудиторию” (DC/GC), редактируются и создаются.

Схема применения LDAP-политик

Объекты queryPolicy могут привязываться к сайту Active Directory, а могут и (что неочевидно) к конкретному контроллеру. Какая же будет логика применения нескольких политик queryPolicy?

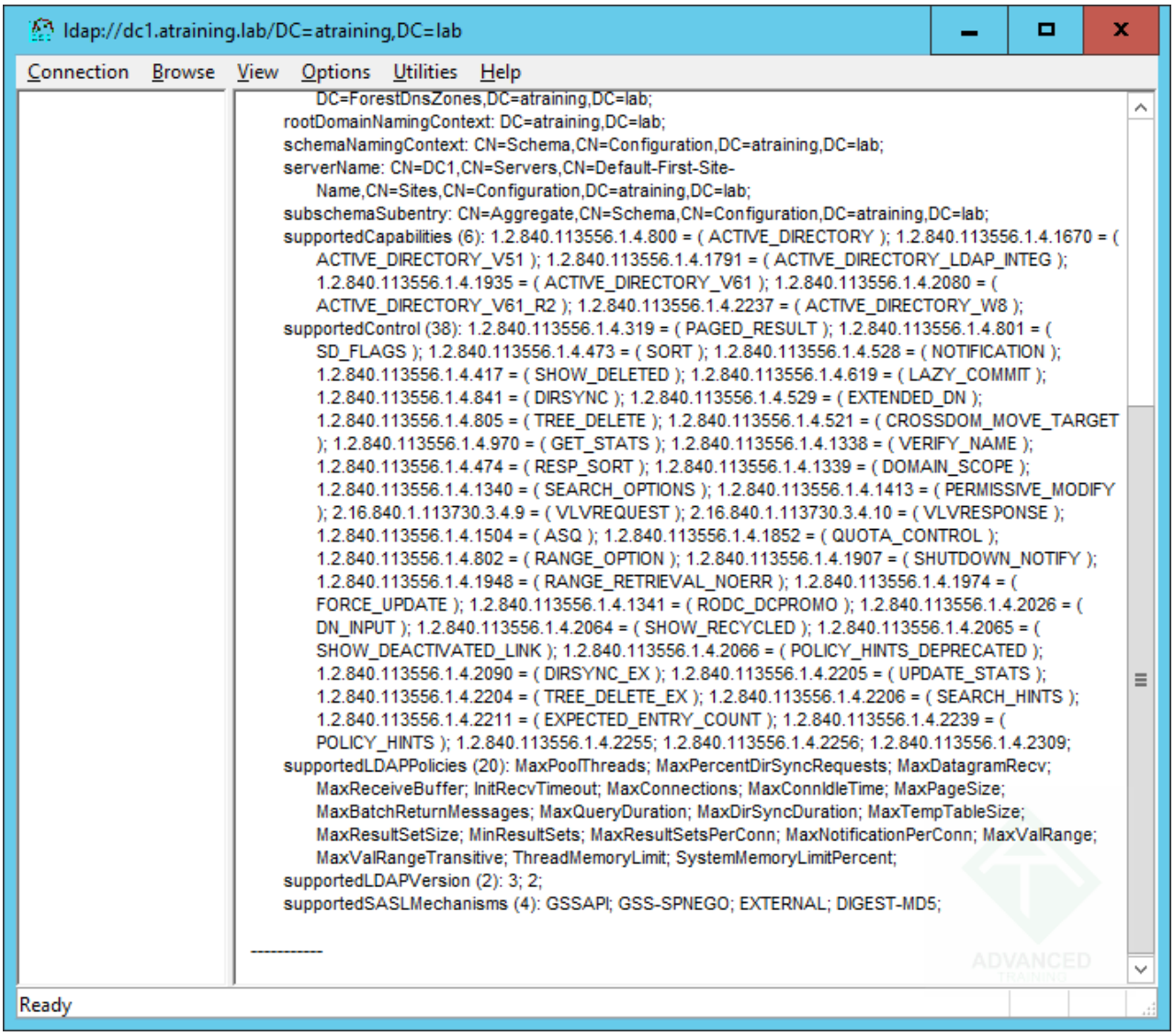
1. Контроллер домена смотрит раздел леса Configuration, открывает там контейнер Sites, находит себя и в своём дочернем объекте с CN=NTDS Settings и типом `nTDSDSA` смотрит на атрибут `queryPolicyObject`. Если там есть DN политики – настройки применяются и обработка прекращается. Т.е. не как в групповой политике, когда результирующая политика представляет собой “сумму наложений” всех политик на объекте, а сразу – раз политику нашёл, то применил и обработку остановил.
2. Если своей политики у контроллера домена нет, он берёт свой сайт, открывает его дочерний объект с CN=`NTDS Settings` и типом `ntDSSiteSettings`, находит там атрибут `queryPolicyObject` (у которого там может и не быть значения, он опциональный), и, в случае наличия, берёт DN объекта LDAP-политики из него. В этом случае обработка также останавливается.
3. Если не нашлось ничего – контроллер не унывает и идёт читать дефолтный объект с именем Default Query Policy. Поэтому, если Вы придумали свои новые и отличные настройки – всегда делайте новый объект, а не редактируйте дефолтный, чтобы иметь возможность лёгкого возврата настроек (что-то не работает – удалили ссылку на свой объект, контроллер стал читать дефолтный).

Добавлю, что, несмотря на одинаковое название – NTDS Settings, у сайта и у DC объект имеет разные типы – Site Settings (`nTDSSiteSettings`) и Domain Controller Settings (`nTDSDSA`).

По сути работа Query Policy ещё проще, чем в случае с групповыми политиками, где стандартную схему применения модицифируют многочисленные дополнительные инструменты. Теперь перейдём к самим настройкам, которые будут разделяться на две части (т.к. атрибута, в которых они задаются, тоже два – `lDAPIPDenyList` и `lDAPAdminLimits`.

Как узнать, какие политики Query Policy поддерживаются

Параметры меняются от версии к версии – и каждый контроллер с удовольствием расскажет вам, какие параметры QueryPolicy он знает и умеет обрабатывать. Для этого в LDAP 3.0 объявлен специальный объект root DSA-specific Entry (`RootDSE`), и у него есть multivalued атрибут `supportedLDAPPolicies` – вы увидите эти данные, просто подключившись к контроллеру, например, при помощи `ldp.exe`:



[Просмотр поддерживаемых Query Policy для контроллера домена Windows Server 2016
(кликните для увеличения до 712 px на 625 px)]

У некоторых параметров есть жестко зафиксированные, на уровне программного кода ядер NT 6.0 и старше, максимальные значения. В этом случае я это адресно указываю – мол, в любом случае, что бы Вы не выставили, параметр будет интерпретироваться как такой-то.

Замечу, что аутентификация не нужна – по RFC 2251 данный объект, описывающий возможности LDAP-сервера, поддерживающего LDAP 3.0, должен быть доступен любому.

Ну а теперь можно и понастраивать.

Настраиваем параметры фильтрации доступа к LDAP

Первое и самое простое – фильтрация доступа к контроллерам. Благодаря политикам LDAP Вы можете выбрать IP-адреса, доступ с которых будет заблокирован. Делается это достаточно просто – в каждом объекте queryPolicy есть

атрибут lDAPIPDenyList, в котором можно указать нужное количество блокируемых адресов. Формат атрибута – массив строк, а каждая строка должна выглядеть просто – IPv4 адрес, пробел и маска. Например строка:

```
172.16.0.0 255.240.0.0
```

заблокирует все обращения к контроллеру с частных IPv4-адресов сетей класса B. Для IPv6 данный механизм не реализован. В устаревшей документации можно встретить утверждение, что данный атрибут работает только для дефолтного объекта LDAP-политик, с `CN=Default Query Policy`. Это не так – работает и в других.

Думаю, что использование такого параметра вполне очевидно – в случае теоретической доступности контроллера со стороны “ненужных” сетей доступ можно заблокировать. Хотя на данный момент в Windows Server имеется Advanced Firewall, который сам может это сделать, функционал блокировки адресов именно на уровне LDAP-запроса присутствует. Во многом по причине того, что когда данный функционал появился (в 1999 году, вместе с рождением Active Directory), встроенного Firewall в Windows Server не было.

Настраиваем параметры функционирования LDAP в Windows Server 2016

Теперь перейдём к основному массиву настроек. Мы разобьём их список по версиям серверной ОС – от Windows 2000 Server до Windows Server 2016.

Формат Query Policy

Все параметры LDAP-политик имеют формат вида “Имя=Значение”. То есть если нужно присвоить атрибуту Parameter значение 7, результирующая строка будет `Parameter=7`.

Параметры, существующие с Windows Server 2000

Параметр Query Policy – MaxActiveQueries

Поддерживается версиями ОС

Только Windows 2000 Server; начиная с Windows Server 2003 параметр игнорируется.

Значение по умолчанию

`20`

Что делает

Когда использовался, указывал максимальное количество параллельно работающих операций поиска в LDAP на конкретном DC. По достижению этого числа DC выдавал при попытке “заказать” операцию поиска ошибку `ERROR_DS_ADMIN_LIMIT_EXCEEDED`. Сейчас не используется, потому что есть более “умная” настройка, ограничивающая количество параллельных операций не для DC, а для ядра процессора.

Параметр Query Policy – InitRecvTimeout

Поддерживается версиями ОС

С Windows 2000 Server.

Значение по умолчанию

`120`

Что делает

Устанавливает время в секундах, за которое клиент после подключения (фактически, после bind) должен отправить первый рабочий запрос на LDAP-операцию. Если тайм-аут проходит, а клиент молчит, его отключают.

Модифицируем, например, в случае использования внешнего ПО, которое любит “подключиться и думать, что сказать для начала”. К такому ПО относится ряд продуктов IBM (например, Lotus Domino), а также продуктов Microsoft – тот же ILM 2007 и FIM 2010. Ситуация достаточно проста – если у Вас есть такое ПО, то, возможно, Вам имеет смысл увеличить тайм-аут хотя бы до 240 секунд (я увеличивал в случае с FIM 2010 до 600, это давало плюсы). Это не приведёт к какой-то дополнительной нагрузке на DC, скорее, наоборот – если ПО разработано так, что оно вначале находит ближайший/лучший DC и “цепляется” за него, после где-то у себя в голове ставит галочку “ОК, подключились, теперь будем, если что, запросы кидать”, и далее штатно делает с AD операции, то не имеет смысла держать тайм-аут таким, чтобы он регулярно сбрасывал подключение этого ПО, приводя ситуацию к “странно, вроде подключились, запрос кидать пробую – ошибка. наверное, AD нерабочая…”. Это может привести в хорошем варианте к повторной аутентикации ПО, в плохом – к неработоспособности и достаточно смутным ошибкам. Учитывайте, что существование данного параметра в не-дефолтном значении будет отрабатываться только в хорошо спроектированном ПО, которое работает с LDAP достаточно качественно. Поэтому в общем случае лучше увеличить этот параметр для “доверенного” ПО – пусть подключается и висит, нагрузки от единичных пустых LDAP-сессий нет, а траблшутить Лотус, который вначале подключается и через полчаса лезет почитать, что там интересного в Active Directory, дело очень унылое.

Данный механизм предназначен для защиты от DDoS, а не для воспитательно-карательных мер в отношении легального ПО. Не имеет смысла “наказывать” специфично спроектированное ПО тем, что сбрасывать его сессию.

Но в случае контроллера, работа с которым ведётся исключительно короткими сессиями (на нём не сидит админ с открытой консолью Active Directory Users & Computers весь день, с него лишь периодически “подкачивают” политику клиентские системы), имеет смысл понизить этот тайм-аут в целях своевременного “сброса” непонятных подключений. Нормальные клиенты подключаются и сразу же делают нужные операции. Опыт настройки в различных инсталляциях показал, что снижение до 15 секунд в такой ситуации является вполне эффективным и никак не влияет на применение групповых политик и LDAP-операции клиентов (даже географически удалённых и с медленными каналами связи).

Ещё раз – это ограничение *стартового* тайм-аута – т.е. ожидания *первой* операции клиента. Idle timeout рассматривается дальше.

Параметр Query Policy – MaxConnections

Поддерживается версиями ОС

С Windows 2000 Server.

Значение по умолчанию

`5000`

Что делает

Обозначает максимально возможное количество одновременных LDAP-подключений на DC. Заметьте – не после ldap bind, а вообще, т.е. в случае с обычным DC это будет математическая сумма подключений по портам TCP 389 и TCP 636. Что интересно, при появлении “лишнего” подключения (в дефолтном случае – 5001го) контроллером будет сброшено какое-то из существующих. Критерий сброса в документации не указан, эксперименты показали, что вроде как сбрасывается самое “старое” подключение, однако не всегда – на одной и той же инсталляции DC иногда сходил с ума и начинал дропить без видимой логики выбора.
Меняем это значение, если думаем о безопасности Active Directory. По сути, данный параметр надо увеличивать хотя бы раз в 10 сразу, потому что иначе атака вида DoS на контроллер домена может быть вполне успешно и оперативно реализована – подключаясь в цикле (даже авторизуясь при этом – кто мешает, читать-то AD могут Authenticated Users, а `RootDSE` и Everyone) можно легко “догнать” суммарное количество сессий до лимита, а после, продолжая инициировать подключения, заставить контроллер сбрасывать рабочие сессии. При этом с точки зрения системы DC загружен не будет (процессор не нагружен, память не кончилась, сетевой интерфейс не загружен, очереди диска нет), а легальные клиенты испытают

проблемы. Правда, когда легальный клиент переподключится, то он “выбьет” одну из фейковых сессий LDAP и, в зависимости от нагрузки, может успеть сделать что-то полезное, пока его, в свою очередь, не “выбьет” скрипт, генерящий фейковые сессии.

Параметр Query Policy – MaxConnIdleTime

Поддерживается версиями ОС

С Windows 2000 Server.

Значение по умолчанию

`900`

Что делает

Устанавливает время бездействия, после которого подключение штатно разрывается со стороны сервера. Важно: время бездействия – это время от *поступления* на сервер последнего запроса клиента (т.е. если время бездействия выставить в 30 секунд, а клиент запросил выборку, которая по медленному каналу качалась бы больше 30 секунд, контроллер не даст команде корректно отработать – проверено).

В случае наличия ПО, которое постоянно держит открытой подключение на контроллеры домена (тот же Exchange) тайм-аут можно и увеличить, но 15 минут обычно вроде как достаточно. Замечу, что этот тайм-аут сбрасывает только уже аутентицированного клиента. Поэтому в случае, например, “жесткой” привязки Exchange на DC/GC вполне можно этот тайм-аут увеличить – уменьшите количество переподключений. В случае же работы с внешними сервисами (трудно придумать сходу – ну, например, реализуя публично доступный LDAP-каталог (lol)) тайм-аут можно сократить в разы, хоть до 15 секунд – обычно в таких сценариях клиент подключается, делает штучный запрос, и всё – его можно отключать. Надо – ещё раз придёт.

Параметр Query Policy – MaxDatagramRecv

Поддерживается версиями ОС

С Windows 2000 Server.

Значение по умолчанию

`4096 байт`

Что делает

Устанавливает максимальный размер UDP-датаграммы, которую обрабатывает DC. Т.е. если придёт датаграмма размером больше указанного, контроллер должен её отбросить.

Надо ли править? Обычно нет по банальной причине – LDAP в Active Directory работает по TCP. Никакого особого КПД найти в ограничении размера UDP-пакета не получится, а раз не получится – то надо взять за правило не трогать параметры без явно подтверждённых на то причин. Есть сценарий, в котором повышение этого размера до 64K может быть позитивным, но он мало относится к теме статьи.

Кстати, ранее (в Windows 2000 Server) это значение было меньше в 4 раза, 1024 байта.

Параметр Query Policy – MaxNotificationPerConn

Поддерживается версиями ОС

С Windows 2000 Server.

Значение по умолчанию

`5`

Что делает

Определяет максимальное количество стоящих в очереди запросов клиента. Т.е. суть проста – клиент подключается, авторизируется, и делает запрос. Клиенту подтверждают, что запрос начинает выполняться, а клиент уже запрашивает следующий. Формируется очередь LDAP-запросов, а данный параметр ограничивает её. Очередь работает по логике FIFO, а действие этого параметра – обычный tail drop, а клиент получает не подтверждение о постановке запроса в очередь, а `ERROR_DS_ADMIN_LIMIT_EXCEEDED`.

Модифицируем очень осторожно; с одной стороны – снижение этого числа может достаточно эффективно ликвидировать возможность хитрого DoS'а – фейковый клиент подключается, подтверждает подлинность и начинает доставать сервер запросами – "хочу всех пользователей, у которых поле X похоже на Y", специально подбирая запросы по таким полям, которые и индексируются, и содержат наиболее обширные данные. Да и ANR тут только ухудшит ситуацию. Вопрос вообще достаточно тонкий, требует понимания и тюнинга всей системы индексации атрибутов объектов в Active Directory и в отрыве от этого рассматриваться не должен. Но в общих чертах – да, уменьшение этого числа для DC, с которыми не работают сервисы типа Exchange, а только обычные клиенты, может оказаться превентивной мерой для описаных выше специфических атак.

Параметр Query Policy – MaxPoolThreads

Поддерживается версиями ОС

С Windows 2000 Server.

Значение по умолчанию

`4`

Что делает

Определяет количество потоков на одном логическом процессоре, доступном DC/GC. Это будут как потоки для обслуживания входящих из сети запросов, так и потоки для обработки LDAP-поиска. То есть, говоря проще, если Вы поставите DC/GC в виртуальную машину и дадите этой виртуальной машине 2 процессора, данный параметр ограничит число параллельных потоков выполнения до 2*4=8. Потоков, заметьте, а не процессов.

Вполне разумно увеличить этот параметр, если у Вас достаточно производительные процессоры. Теоретически можно придумать DoS-атаку, которая выдаст столько запросов (и достаточно “долгоиграющих”), что контроллер запустит их параллельно, и их количество будет достаточно, чтобы “положить” систему. Смотрите и думайте сами, я лишь могу предложить увеличивать этот параметр для систем с быстрыми CPU – до 10 потоков на процессор.

Параметр Query Policy – MaxReceiveBuffer

Поддерживается версиями ОС

С Windows 2000 Server.

Значение по умолчанию

`10,485,760 (10 МБайт)`

Что делает

Это – размер буфера для одиночного запроса клиента. Если думаете, что это много, просто вспомните, что запрос – это не обязательно текстовая строчка вида `(&(l=Кремль)(|(givenName=Дима)(givenName=Вова)))`, это может быть и масштабная выборка вида “надо все поля всех клиентов, у кого почта на .ru заканчивается”.

Данный параметр тесно связан с количеством одновременно возможных запросов. По сути, вся его оптимизация – это экономия потенциально выделенной памяти в количестве MaxReceiveBuffer * MaxConnections. Заметьте, память выделяется динамически, поэтому наличие этого буфера в 10МБ и количества подключений в 5000 не говорит о том, что DC/GC попытается выделить 50ГБ под буферизацию LDAP-запросов. Т.е. такое возможно в теории, на практике под каждый запрос сразу 10МБ не выделяется. Лучший вариант – мониторинг Вашей инсталляции Active Directory и, в случае отсутствия запросов подобных размеров, снижение этого числа

(которое в реальности сделано с большим запасом). На практике в достаточно больших инсталляциях (несколько тысяч хостов, порядка 20 серверов Exchange) данный параметр, будучи установленым в 1МБ, не вызывал никаких проблем.

Кстати, в своё время (в Windows Server 2003) была ошибка, связаннае с тем, что сервер некорректно выделял память под буфер, если его размер в байтах больше странного числа 10737418. Ошибку давно поправили, лечилась она форсированной установкой буфера в 10485760 байт.

У этого параметра есть верхний лимит – 20971520 (в случае попытки выставить параметр выше по факту будет применено это значение).

Параметр Query Policy – MaxPageSize

Поддерживается версиями ОС

С Windows 2000 Server.

Значение по умолчанию

`1000`

Что делает

Этот параметр достаточно неочевиден по своей настройке. Суть в том, что это не максимальный размер возвращаемых запросом результатов, а размер одной страницы с этими результатами. Если при поиске будет возвращено большее количество объектов, то они будут возвращаться блоками, которые и называются страницами. Сервер держит эти блоки в RAM пока клиент не заберёт их все либо не сделает unbind.

Модифицируем в случае, если хорошо понимаем LDAP и то, что этот параметр, по сути, меняет баланс между “сформировать много страниц ответа и отдавать по одной быстро” vs “сформировать мало страниц и отдавать подольше, но реже”. Интересным является тот факт, что “родной” клиент LDAP в Windows всегда делает paged-запросы, поэтому в принципе сломать что-то этим параметром трудно. Учтите, что тут есть строгая зависимость от того, какие запросы делает клиент – т.е. в случае наличия “глупого” ПО, которое делает не-paged запросы, Вы реально можете ему помешать работать. Если такого ПО нет, я бы рекомендовал снизить этот параметр до 100 исключительно по причине того, что очень малое число запросов к AD возвращают более 100 объектов, а выделять лишнюю память для буферизации смысла нет. Верхний лимит параметра – 20000 (в случае попытки выставить параметр выше по факту будет применено это значение).

Вообще, нужно учитывать достаточно простую логику Microsoft – все ldap-запросы должны быть с поддержкой paging, поэтому не-paged запросы в общем-то являются потенциально unsupported. Подробнее, если Вы разработчик, можно найти в документации на `ldap_create_page_control`.

Параметр Query Policy – MaxQueryDuration

Поддерживается версиями ОС

С Windows 2000 Server.

Значение по умолчанию

`120`

Что делает

То, что и в названии – максимальное время обработки одиночного запроса. Считается от момента поступления оного, а не от подключения клиента, что логично. Как только время закончится – запрос будет остановлен и клиенту будет возвращена ошибка `ERROR_INVALID_PARAMETER`.

Тонкость – это произойдёт только если запрос всё выполняется. Если же он уже выполнен, вернул много результатов и их постранично забирает клиент, то клиента не отключат. Т.е. цель этого механизма – отключать запросы, которые “перегревают” DC, а не стирать через 2 минуты результаты уже готовых, которые сформированы и лежат в буфере.

На практике этот параметр можно реально уменьшить, и сильно. Две минуты на формирование результатов запроса – это надо иметь огромный лес, сложнейший запрос к GC и очень медленный сервер. Если это не так, запрос даже в 10 секунд – сложная в реальности штука. Опять же – измеряйте Вашу реальную ситуацию и модифицируйте настройки в случае необходимости. Верхний лимит параметра: 1200 (в случае попытки выставить параметр выше по факту будет применено это значение. хотя трудно себе представляю штатный запрос к DC/GC, который в норме выполняется дольше 20 минут).

Параметр Query Policy – MaxResultSetSize

Поддерживается версиями ОС

С Windows 2000 Server.

Значение по умолчанию

`262,144`

Что делает

Устанавливает размер памяти в байтах (в байтах, а не как иногда пишут – “в объектах”), выделяемых на все результаты всех активных сейчас paged-запросов. Т.е. ещё раз – на вообще все, которые на сервере сейчас уже выполнены, и клиенты их забирают. И именно на paged, которые постранично забирают клиенты.

Если места на кэширование результатов нового запроса не хватает, то (ключевое) удаляются результаты старого. И если клиент продолжит его забирать, то надо бы его выполнить повторно.

Лучше всего увеличить этот параметр хотя бы до мегабайта. В крупных инсталляциях хорошо срабатывает увеличение до 16 (число выбрано по причине мониторинга и обнаружения ситуаций, когда в буфере лежало до 10МБ результатов и взято с небольшим запасом). Цель – отсутствие ситуации, когда результаты нового paged-запроса затирают недополученные клиентом результаты более старого paged-запроса.

Кстати, в документации Microsoft есть опечатка – там параметр иногда называется MaxResultSize. Такого параметра нет, можете проверить. Вообще, конечно, параметр логичнее было бы назвать MaxResultSetsSize или какой-нибудь MaxTotalResultSetsSize, но увы.

Параметр Query Policy – MaxTempTableSize

Поддерживается версиями ОС

С Windows 2000 Server.

Значение по умолчанию

`10000`

Что делает

Достаточно интересный параметр. Суть его в следующем – когда выполняете достаточно сложный запрос, в котором есть логические операции (например, объединения или пересечения множеств), то возникает необходимость в формировании промежуточных таблиц с результатами выборок. В них лежат так называемые candidate object'ы – объекты, часть из которых попадёт в результат. Если размер промежуточной таблицы превзойдёт указанное число записей, то система перестанет обрабатывать запрос пошагово (выбрать множество -> выделить подмножество -> из него выбрать по критерию -> и т.д.) а перейдёт к т.н. direct scan (будет обрабатывать потенциальные объекты по-одному). Вот этот параметр – это максимальный размер такой таблицы для каждого из запросов.

Подумайте, будут ли у Вас запросы, промежуточные результаты которых будут превышать это число, и, в случае наличия оных, увеличьте этот параметр. И обломайтесь – по факту он не увеличится. Производительность таких запросов серьёзно упадёт и Вы ничего сделать не сможете – в Windows Server 2008 R2 максимальное значение этого параметра как раз 10000. Увы, как ни странно, в Windows Server 2003 это можно было регулировать гибче. Т.е. верхний лимит параметра как раз и есть его значение по умолчанию, 10000 – в случае попытки выставить параметр выше по факту будет применено это значение.

Параметры, существующие с Windows Server 2003

Параметр Query Policy – MaxValRange

Поддерживается версиями ОС

С Windows Server 2003.

Значение по умолчанию

`1500`

Что делает

Устанавливает максимальное число результатов, являющихся полями одного атрибута одного объекта, которые может вернуть один LDAP запрос. В общем-то ключевое, под что параметр заточен – это атрибут members у группы. Политика появилась в Windows Server 2003 вместе с изменениями в репликации multivalued-атрибутов (если помните, именно тогда и при помощи LVR был убран штатный для Windows 2000 Server конфликт репликации multivalued-атрибутов). Ранее, в Windows 2000 Server, значение было установлено на 1000 и не могло быть изменено штатным способом.

Модифицируем, например, в случае наличия групп с количеством участников выше 1500. Хитрый DoS, блокируемый этим параметром, придумать можно, но вот проблемы с отправкой почты на адрес “Все сотрудники” в случае линейного включения всех учетных записей пользователей в группу придумываются ощутимо быстрее. Хотите упростить ситуацию – ставьте сразу на 5000 – это верхний лимит параметра.

Параметры, существующие с Windows Server 2008 R2

Параметр Query Policy – MaxResultSetsPerConn

Поддерживается версиями ОС

С Windows Server 2008 R2.

Значение по умолчанию

`10`

Что делает

Устанавливает максимальное число хранимых со стороны сервера результатов поисковых запросов клиента. Модифицируем в случае, если у нас очень много параллельных запросов. Фактически, в хостинговых сценариях (например, хостинг

Exchange). Кстати, в случае, если Ваш домен был установлен не сразу как Windows 2008 R2, параметр будет равен нулю (что по факту опять-таки превратит его на поддерживающих этот параметр контроллерах в 10).

Параметр Query Policy – MinResultSets

Поддерживается версиями ОС

С Windows Server 2008 R2.

Значение по умолчанию

```
3
```

Что делает

Задаёт минимальное число параллельных запросов для включения режима оптимизации paged-запросов, доступного в NT 6.1. Модифицируем просто – если поставить единицу, тогда режим оптимизации будет инициироваться всегда, и отработка запросов ускорится.

Параметры, существующие с Windows Server 2012

Параметр Query Policy – MaxBatchReturnMessages

Поддерживается версиями ОС

С Windows Server 2016.

Значение по умолчанию

```
1100
```

Что делает

Задаёт максимальное число ответов при заказе расширенной операции с полем “хочу пачкой (batch)” – LDAP_SERVER_BATCH_REQUEST_OID. Эти операции – это, допустим, расширенный поиск с флагами LDAP_SERVER_DOMAIN_SCOPE_OID, LDAP_SERVER_SHOW_DELETED_OID, LDAP_SERVER_SHOW_RECYCLED_OID и подобными. В ответ на такой запрос может быть отдано много одиночных LDAP Message – вот как раз их максимальное количество, которое надо закэшировать и отдавать клиенту, здесь и задаётся. Параметр можно увеличить в больших инсталляциях – но надо, конечно, для начала мониторить происходящее, чтобы увидеть, есть ли такие запросы на практике.

Параметры, существующие с Windows Server 2016

Параметр Query Policy – MaxDirSyncDuration

Отключаем жестко заданные лимиты настроек

В многих настройках я указывал, что они жёстко ограничены – можно указать любое значение, просто если оно больше некого X, то оно будет расцениваться как X.

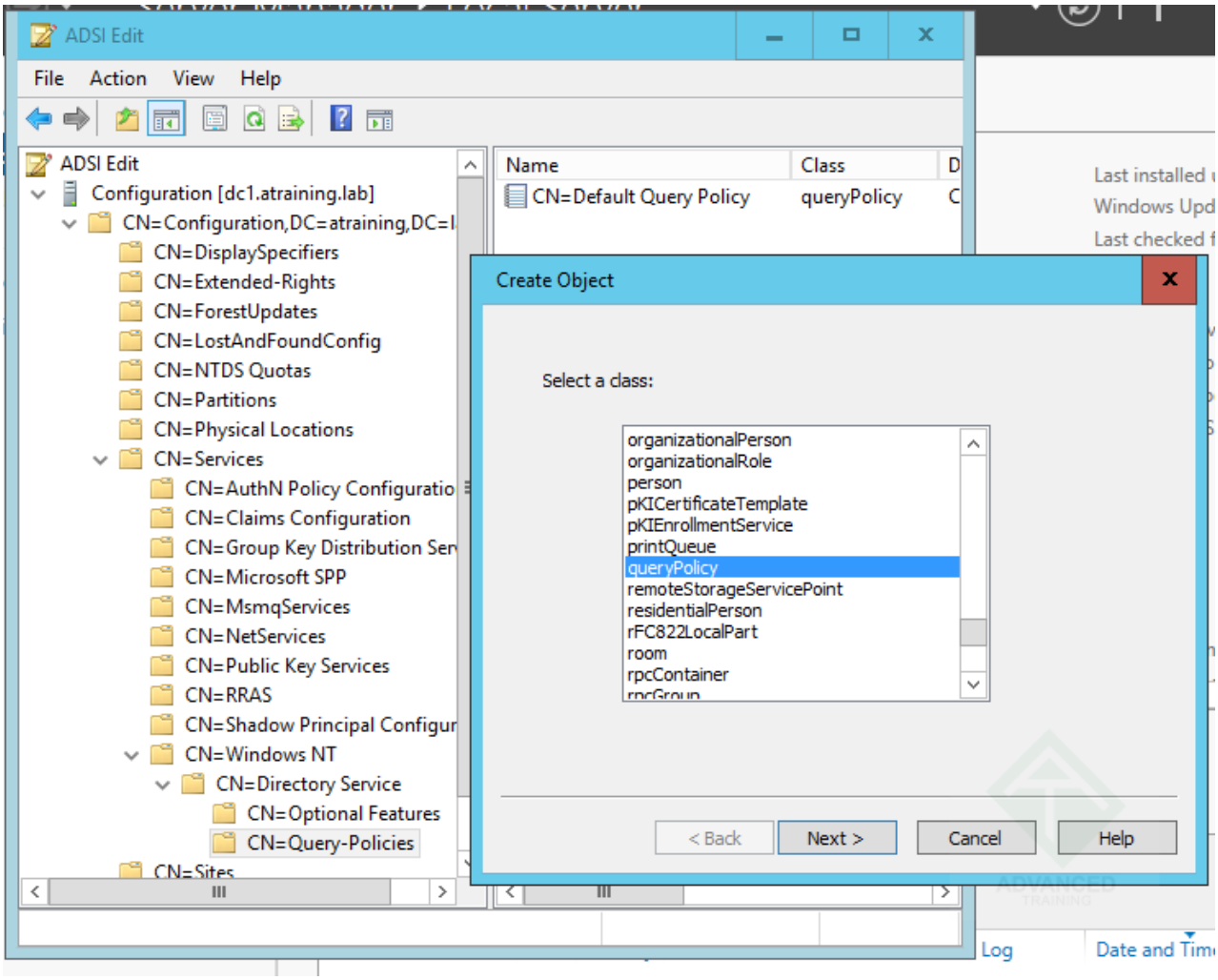
Это появилось с Windows Server 2008 и – что удивительно – это можно отключить. Последовательность действий для этого проста:

1. Открываете редактор – подойдёт ADSI;
2. Открываете редактор атрибутов для объекта **CN=Directory Service,CN=Windows NT,CN=Services,CN=Configuration,DC=*ваш лес*** – именно для него, для контейнера!
3. Берёте атрибут `dSHeuristic` и выставляете ему значение `000000001000000001`, если атрибут пустой; этим выставляется битовый флаг `fLDAPBypassUpperBoundsOnLimits`, который читается DC с Windows Server 2008 и старше;

Делайте это крайне осторожно – это атрибут масштабов леса, у него каждый бит делает что-то определённое – не ошибитесь.

Создаём новую Query Policy

Это достаточно тривиальная задача – откроем ADSI, подцепимся к разделу Configuration, откроем последовательно CN=Services, потом CN=Windows NT, потом CN=Directory Service, потом CN=Query-Policies, и, увидев одинокую политику по умолчанию, создадим новый объект queryPolicy:



[Создаём новую LDAP Query Policy для леса Active Directory на базе Windows Server 2016]
[(кликните для увеличения до 740 px на 592 px)]
У объекта надо будет только задать CN – остальные параметры правятся уже просто, редактированием соответствующего атрибута. Так как он один и в начале статьи приведён, дублировать тут не буду.

Я всё прочитал и поменял все параметры. Что теперь?

Теперь, если всё продолжает работать, можно измерить производительность того, что мы наделали. Не наоборот. Т.е. после тюнинга LDAP-политик в первую очередь всё должно продолжить функционировать, а во вторую – улучшить какие-либо характеристики.

Надеюсь, что данный материал поможет Вам эффективнее работать с инфраструктурой на базе Active Directory, а также покажет, что в данном сервисе есть достаточно много такого, что гораздо интереснее, чем унылые инструкции про “Next->Next->Finish->вроде-кое-как-заработало”.

Техники атаки и защиты LDAP-хранилищ достаточно оригинальны и разнообразны (например, неавторизованный пользователь может заказать априори огромный по количеству результатов запрос, притом запросить, чтобы он (результат запроса)

ещё и был отсортирован по неиндексированному атрибуту), поэтому грамотный инженер должен хорошо понимать возможности Active Directory по тюнингу такого функционала.

Удач!