Оглавление

Руководство по атаке на доменные трасты

Опубликовано 30 октября 2017 года, автор - harmj0y
Перевод – студия Lemma Works

Прошло немало времени (около 2 лет) с тех пор, как я написал статью о доменных трастах в Active Directory. Далее, увлёкшись вопросом области действия групп и углубившись в его изучение, я переосмыслил некоторые заблуждения, которые у меня были ранее относительно трастов и нахождения в группах. Это, в сочетании с изменениями, внесёнными в PowerView в прошлом году, убедило меня опубликовать актуальное руководство по вычислению и атаке доменных трастов. Скорее всего, это будет последняя статья, посвящённая доменным трастам, которую я опубликую на некоторое время, особенно учитывая то, что 8000 слов - это не совсем лёгкое чтение (никто не читает длинные посты :) В общем, я не просто веду блог о своих рабочих заметках - я стараюсь писать статьи, которые являются полным руководством для участников моей команды, в надежде, что эта информация будет полезна другим (причём, и с «наступательной» и с «оборонительной» точки зрения).

Я хочу, чтобы эта статья была как можно более полной, поэтому я расскажу обо всех аспектах трастов, как мы их понимаем сейчас. Как и в случае с моими предыдущими публикациями, я хочу как можно объёмнее изложить свои знания *на данный момент*. Наши знания и ремесло всегда развиваются - и трасты не исключение. У меня было несколько заблуждений относительно доменных трастов, когда я только начал писать о них. Я никогда не был системным администратором или архитектором AD: я получал свои знания постепенно, что (надеюсь) объясняет пробелы, «всплывшие» в моих прошлых статьях - и они, я уверен, будут возникать и впредь.

Поэтому я собираюсь начать всё сначала - на случай, если вы не знакомы с предыдущими статьями, которые я публиковал. Таким образом, в некоторых частях этой публикации будут переосмыслены определённые элементы и формулировки из предыдущей работы, интегрированные с обновлёнными знаниями и синтаксисом PowerView. И так как это лишь небольшая часть статьи, я уверен, что где-то есть ошибки и вещи, которые я упустил. Поэтому, когда вы найдёте их, дайте мне знать - и я всё исправлю соответствующим образом!

Самая актуальная версия PowerView всегда будет в ветке разработчика PowerSploit.

Что же это, чёрт возьми, такое - доменные трасты?

Доменный траст даёт пользователям возможность проходить проверку подлинности на ресурсах в одном домене или действовать в качестве принципала безопасности в другом домене. У Microsoft есть много информации отсюда о

доменных трастах, а также статья «Вопросы безопасности для трастов», и это может порой вносить некоторую путаницу. Согласно тексту Microsoft, «*большинство организаций, имеющих более одного домена, сталкиваются с необходимостью обеспечивать пользователям доступ к общим ресурсам другого домена*», и трасты позволяют организациям с несколькими доменами предоставлять пользователям доступ к общим ресурсам. **Доменные леса** – это наборы контейнеров доменов, которые имеют между собой определённый уровень «доверия». Также эти леса могут иметь пересекающиеся трасты. У Microsoft есть отличный пост о том, как работают доменные трасты и леса. Если вы не знакомы с этой темой, я рекомендую вам прочитать его.

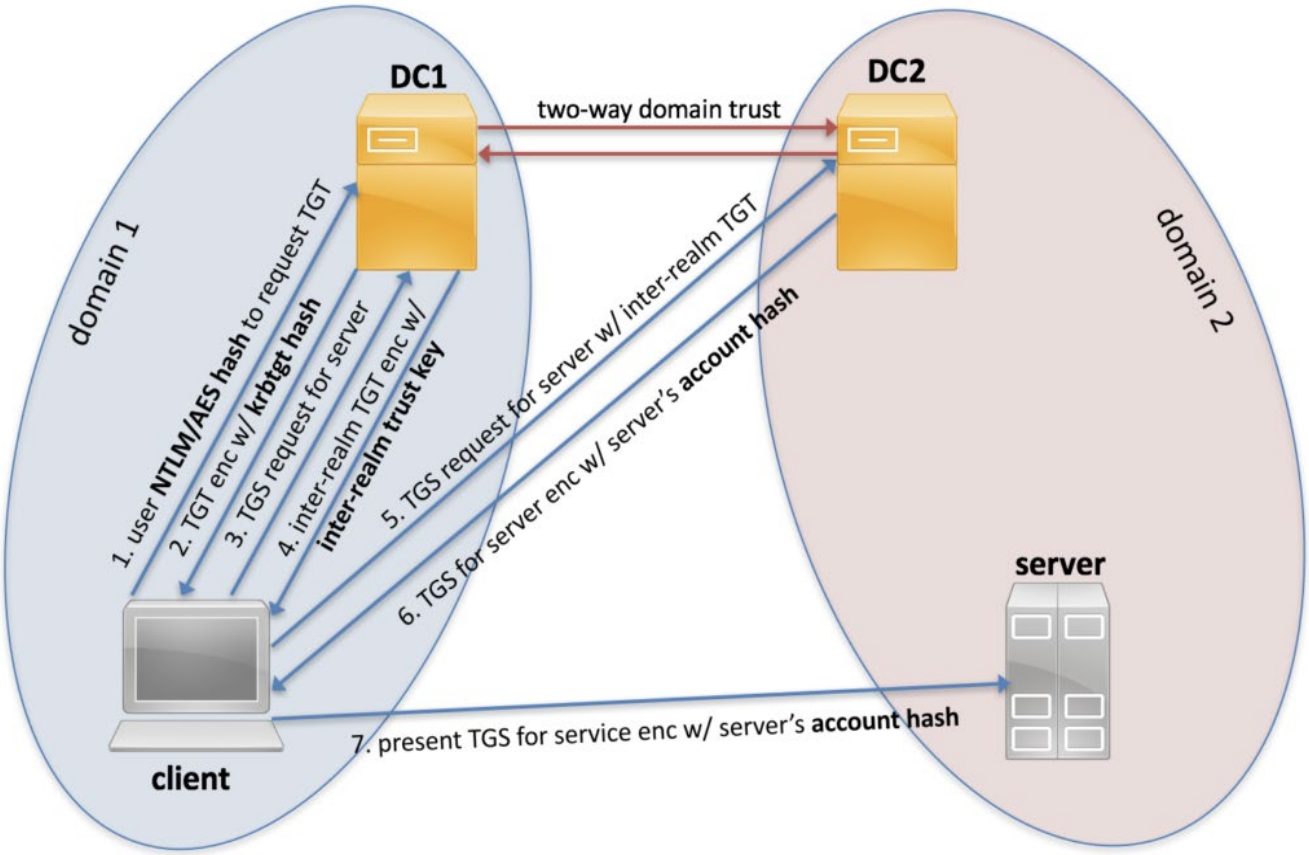
По сути, всё, что делает траст – связывает системы аутентификации двух доменов и позволяет трафику аутентификации проходить между ними через систему перенаправлений. Если пользователь запрашивает доступ к первичному имени сервиса (SPN) ресурса, находящегося за пределами домена, в котором они находятся, их контроллер домена даст отклик в виде специального билета, указывающего на центр распространения ключей (*KDC, в случае Windows - контроллер домена*) чужого домена.

Ticket-granting-ticket пользователя (TGT) включён в TGS-REP (отклик службы предоставления билета), и шифруется/подписывается межобластным ключом доверия (тем же ключом, которым ранее обменивались домены) вместо учётной записи первого домена krbtgt. Этот билет обычно называют «inter-realm ticket-granting-ticket /TGT». Затем чужой домен проверяет/дешифрует TGT, включённый в реферал, расшифровывая его с помощью ранее согласованного межобластного ключа доверия, и выполняет остальную часть обычного процесса Kerberos.

Шон Меткалф подробно разбирает этот процесс в своей статье «Всё о трасте», и описывает его следующим образом: *«Когда имеется траст между двумя доменами… служба выдачи билетов каждого домена (в Kerberos называемых «областью – realm») зарегистрирована в качестве принципала безопасности в службе другого домена Kerberos (KDC). Это позволяет службе выдачи билетов в каждом домене рассматривать её в другом домене как всего лишь ещё одну услугу, предоставляющую междоменный служебный доступ к ресурсам в другом домене»*.

Таким образом, по существу, когда чужой домен расшифровывает реферальный билет с помощью согласованного ключа доверия, он видит TGT пользователя и говорит: *«Хорошо, другой домен уже аутентифицировал этого пользователя и сообщил, что ему можно доверять. Так что я приму это как данность, потому что я доверяю домену, который выдал реферал».*

Вот картинка для визуализации процесса Kerberos через границы траста:



Цель установления траста - предоставить пользователям из одного домена доступ к ресурсам (например, к группе локальных администраторов на сервере), возможность быть включёнными в группы или иным образом участвовать в качестве принципалов безопасности другого домена (например, для списков ACL объекта AD). Единственное исключение - это доменные трасты леса доменов (доменные трасты, которые существуют в одном и том же дереве Active Directory): любой домен леса сохраняет скрытые двусторонние транзитивные доверительные отношения со всеми другими доменами этого леса. Это влечёт множество последствий, которые я распишу чуть позже в этой статье.

Но перед этим мы должны обсудить ещё несколько характеристик трастов.

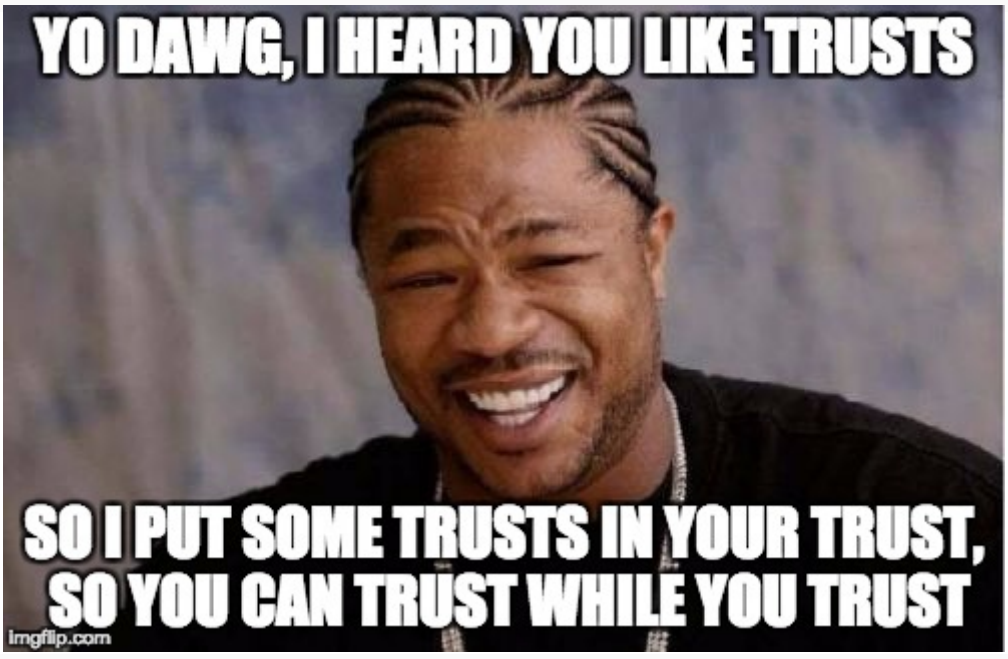
Существует несколько типов трастов, и некоторые из них имеют различные неявные агрессивные импликации:

- **Родительский/дочерний** - часть одного и того же леса - дочерний домен поддерживает скрытый двусторонний транзитивный траст со своим родителем. Это, вероятно, самый распространённый тип траста, который вам встретится.
- **Перекрёстный** - так называемый «быстрый траст» между дочерними доменами для ускорения времени перехода. Обычно подобные ссылки в сложном лесу доменов должны фильтроваться до самого корня и затем возвращаться обратно

к целевому домену, поэтому для такого сценария перекрёстные ссылки могут значительно сократить время аутентификации.

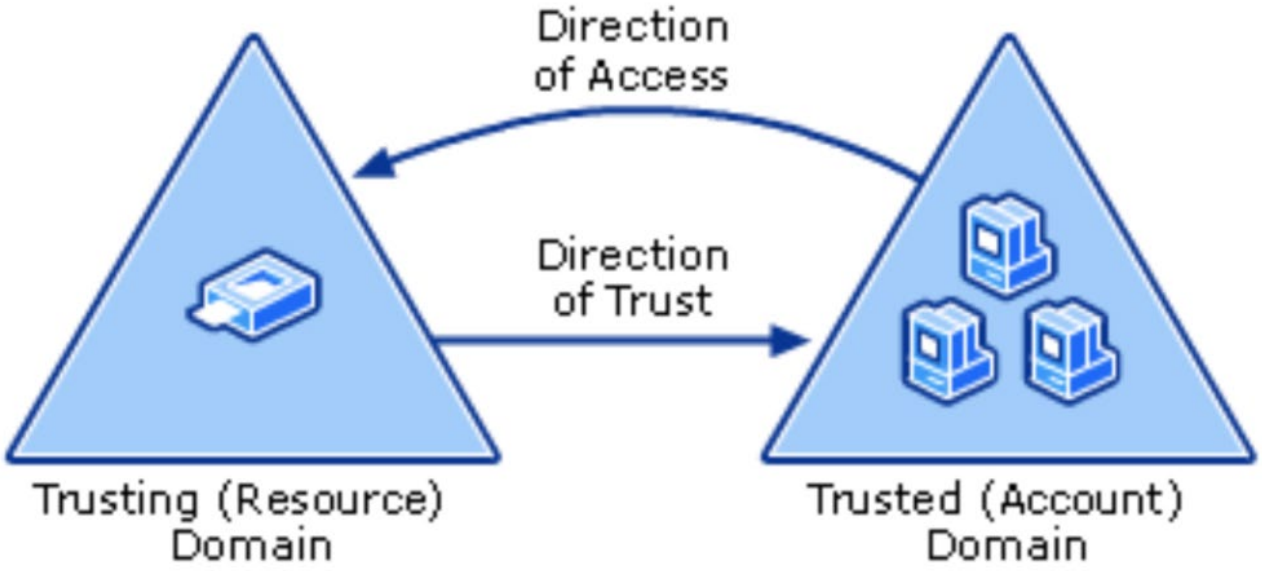
- **Внешний траст** – нетранзитивный траст, созданный между несопоставимыми доменами. «*Внешние трасты обеспечивают доступ к ресурсам в домене за пределами леса, к которому ещё не присоединился ни один траст другого леса*». Внешние трасты обеспечивают SID-фильтрацию, то есть защиту, о которой будет рассказано далее.
- **Корневой** траст – скрытый двусторонний транзитивный траст между корневым доменом леса и корневым доменом нового дерева, которое надлежит добавить. Я не слишком часто сталкивался с корневыми трастами, но, согласно документации Microsoft, они появляются, когда вы создаёте новое дерево доменов. Эти трасты сохраняют двустороннюю транзитивность, позволяя дереву иметь отдельное доменное имя (вместо child.parent.com).
- **Траст леса** – транзитивный траст между корневым доменом одного леса и корневым доменом другого леса. Трасты леса также обеспечивают фильтрацию SID.
- **MIT** - траст к домену Kerberos, не поддерживающему Windows RFC4120. Я надеюсь в будущем более тщательно разобраться в MIT-трастах.

Транзитивность, да? Ещё одним аспектом доменных трастов является то, что они бывают транзитивными или нетранзитивными. Процитируем, что написано в документации MSDN о транзитивности: «*Транзитивный траст распространяет доверительные отношения на другие домены; нетранзитивный траст не распространяет доверительные отношения на другие домены*». Это означает, что транзитивные трасты могут быть объединены в цепочку, вследствие чего пользователи потенциально могут получать доступ к ресурсам в нескольких доменах. Имеется в виду, что если домен A доверяет B, а B доверяет C, то A неявно доверяет C (мама любит папу, папа любит пиво – значит, мама любит пиво ☺).



Кроме того, трасты могут быть односторонними или двусторонними.

Двунаправленный (двусторонний) траст – в действительности это всего лишь два односторонних траста. Односторонний траст означает, что пользователи и компьютеры в *доверенном домене* потенциально могут получить доступ к ресурсам в другом *доверяющем домене*. Односторонний траст действует только в одном направлении, отсюда и название. Пользователи и компьютеры в *доверяющем* домене не могут получить доступ к ресурсам в *доверенном* домене. Microsoft хорошо разъясняет это всё визуально:



https://technet.microsoft.com/en-us/library/cc759554(v=ws.10).aspx

Именно это смущало меня, когда я только начинал разбираться с трастами: с перспективы атаки то, что нас волнует – это *направление доступа*, а не *направление доверия*. При одностороннем трасте, где **A-trusts->B**, если траст отслеживается из домена A, этот траст помечается как *исходящий*; если тот же самый траст отслеживается из домена B, то он будет отмечен как *входящий*, потенциальный же доступ к домену **A** получит домен **B**.

Почему это должно нас волновать?

Да, в самом деле - почему?

Доменные трасты нередко непреднамеренно предоставляют пути доступа между средами. Во многих организациях трасты были внедрены несколько лет назад (иногда более 10 лет) без придания должного внимания вопросам безопасности. Некоторые юридические лица, приобретающие новую компанию, часто просто «подключают» сеть Active Directory новой компании либо в качестве дочернего домена, либо в качестве внешнего траста - без учёта того, как это повлияет на безопасность.

Поскольку в прошлом было не так много инструментов, позволяющих точно вычислять и визуализировать риск, связанный с неверно настроенными трастами, многие архитекторы доменов не знают о непреднамеренном риске, который представляют их трастовые архитектуры Active Directory. Это связано с идеей «misconfiguration debt (неверной настройки)», о которой @wald0, @cptjesus и я говорили на Derbycon в этом году. Из-за этого различные «красные команды» (и, вероятно, APTz, как я полагаю) годами злоупотребляли трастами Active Directory с большим успехом.

Распространённым сценарием атаки является компрометация домена разработки или второстепенного домена и использование этого доступа для проникновения в защищённый корень доменного леса. Это также открывает возможности для постоянных атак - зачем оставлять код работающим в защищённой среде, если у вас есть внедрённая нагрузка, работающая в менее защищённом (но доверенном) домене, которая затем может быть использована как угодно для повторной атаки на вашу цель?

Траст в доменном лесу (родительский, дочерний или корневой) приводит нас к превосходному вектору атаки, который описан здесь в разделе **«Трастпокалипсис»**. Внешние трасты и трасты между лесами *не гарантируют* какого-либо привилегированного доступа, но, как минимум, наличие такого траста означает, что вы можете запрашивать любую обычную информацию Active Directory у домена, который вам доверяет (да, это означает, что в некоторых случаях вы можете использовать Kerberoast через трасты - подробнее об этом в конце статьи). В конце концов, мы рассматриваем Active Directory как базу данных с информацией ☺

Стратегия атаки траста

Прежде чем мы углубимся в технические детали того, как точно вычислять и атаковать трасты, я хотел бы немного сказать о высокоуровневой стратегии, которую я использую при аудите трастовых отношений. Когда я говорю о «стратегии атаки траста», я имею в виду способ бокового смещения из домена, в котором вы сейчас находитесь, в другой домен, на который вы нацелены.

(1) Первый шаг - вычислить все трасты, которые есть у вашего текущего домена, а также любые трасты, которые есть у *тех* доменов, и так далее. По сути, вам нужно создать схему всех доменов, к которым вы можете обратиться из вашего текущего состояния путём перехода по трастовым рефералам. Это позволит вам определить домены, через которые вам нужно пройти, чтобы добраться до вашей цели, и методы, которые вы можете использовать, чтобы (возможно) достичь её. Любые домены в данной «сетке», которые находятся в одном и том же лесу (например, отношения parent->child), представляют особый интерес из-за технологии

SIDhistory-trust-hopping, разработанной Шоном Меткалфом и Бенджамином Делпи, также описанной в разделе **«Трастпокалипсис»**.

(2) Следующим шагом является вычисление любых пользователей/групп/компьютеров (принципалов безопасности) в одном домене, которые либо (1) имеют доступ к ресурсам в другом домене (т. е. членство в группах локальных администраторов или входы DACL ACE), либо (2) состоят в группах или (если это группа) у них есть пользователи из другого домена. Суть в том, чтобы найти отношения, которые каким-то образом пересекают обозначенные трастовые границы и, следовательно, могут обеспечить тип «моста доступа» из одного домена в другой в этой сетке. Хотя междоменная вложенная связь не гарантирует облегчения доступа, трасты обычно реализованы по определённой причине, то есть чаще всего вероятно существует некий тип междоменной «вложенности» пользователя/группы/ресурса, и во многих организациях такие отношения настроены неправильно. Также стоит заметить - как уже упоминалось, применение Kerberoasting'а через трасты **может** быть ещё одним вектором для перехода границы траста. Для дополнительной информации смотрите раздел **«Ещё одна заметка на полях: Kerberoasting через доменные трасты»**.

(3) Теперь, когда вы разметили трастовую сетку, типы и междоменные вложенные отношения, у вас есть «карта» учётных записей, которые вам нужно скомпрометировать для перехода из текущего домена к вашей цели. Выполняя атаку на целевую учётную запись и используя SID-history-hopping для доменных трастов в лесу, мы смогли пройти более 7 доменов, чтобы достичь нашей цели.

По меньшей мере помните, что если домен доверяет вам, т. е. если траст является двунаправленным или является односторонним и входящим, то вы можете запросить любую информацию Active Directory из *доверяющего* домена. И помните, что все отношения parent->child (трасты внутри леса) сохраняют скрытый двусторонний транзитивный траст друг с другом. Также, благодаря тому, что добавляются дочерние домены, группа «Администраторы предприятия» автоматически добавляется в локальную группу домена «Администраторы» в каждом домене леса. Это означает, что траст «проистекает» из корня леса, определяя нашу цель - перейти от дочернего траста к корневому на любом подходящем этапе цепочки атак.

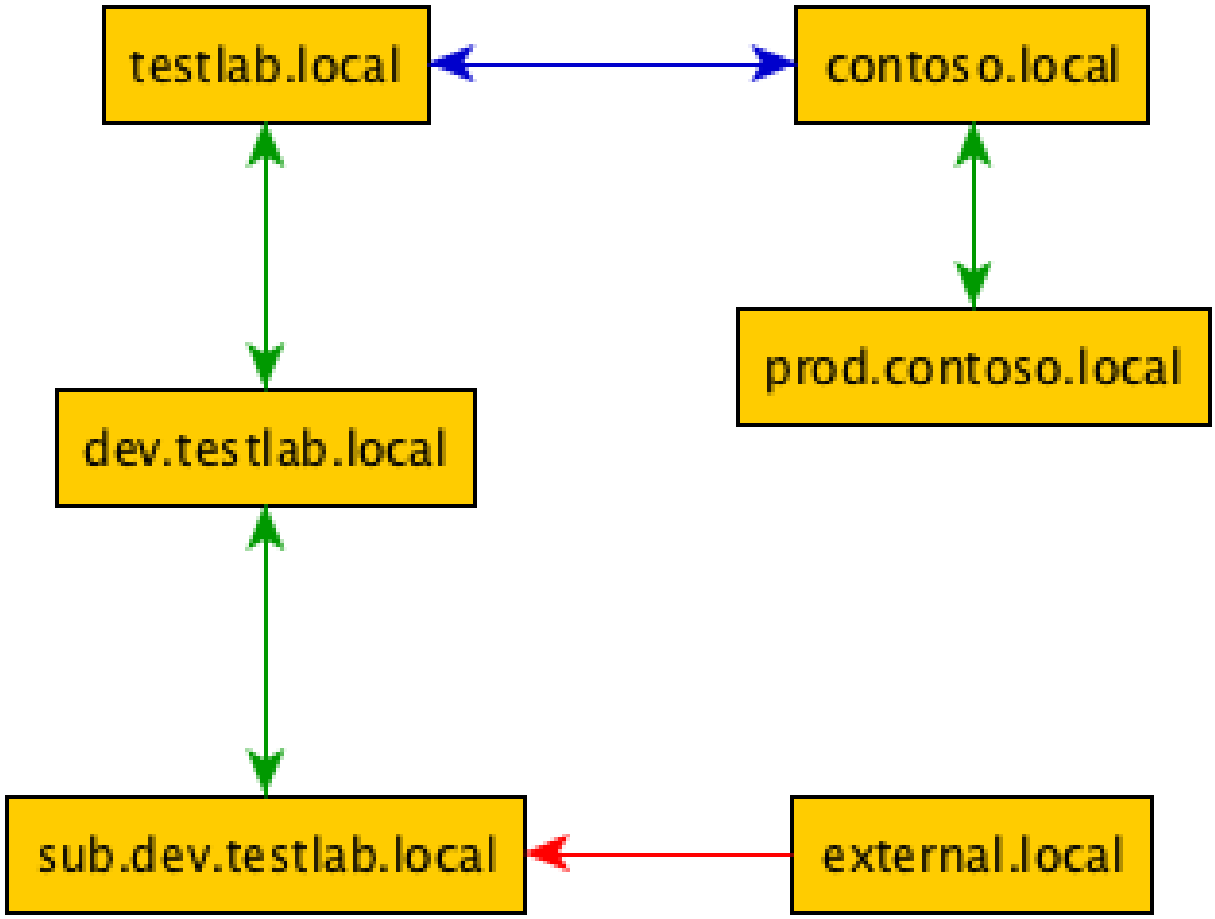
Хорошо, как мне вычислять трасты?

О'кей, Уилл, ты пробудил мой интерес. Как мне выяснить, какие доверительные отношения существуют в моём окружении?

Насколько я знаю, существуют три основных метода для вычисления трастов: вызовы Win32 API, различные методы .NET и LDAP. Удручает то, что каждый из них

выдаёт различный набор информации, и у каждого свои методы исполнения. Я расскажу обо всём: от старомодных способов до новых, от встроенных (и внешних) двоичных файлов до .NET, вызовов Win32 API, PowerShell/PowerView и BloodHound.

Пример трастовой архитектуры, которую я буду использовать в этой статье:



Это изображение было сгенерировано с помощью программы TrustVisualizer (описанной в разделе **«Визуализация доменных трастов»**). В этой новой версии **зелёные** указатели означают «в лесу», **красные** означают внешние, а **синие** означают доверительные отношения между лесами. Как и в случае с @sixdub'sDomainTrustExplorer, направления стрелок для одностороннего траста означают *направление доступа*, а не *направление траста*.

Методы .NET

.NET предоставляет нам несколько хороших методов, которые могут определить значительную часть информации о домене и трасте леса. Это был первый метод, реализованный PowerView ещё до перехода к Win32 API и методам LDAP.

Пространство имён [System.DirectoryServices.ActiveDirectory.Domain] имеет статический метод GetCurrentDomain(), который даёт отклик в виде экземпляра класса System.DirectoryServices.ActiveDirectory.Domain. В этом классе реализован метод GetAllTrustRelationships(), который деликатно *«извлекает все доверительные отношения для этого домена»*. Одним из преимуществ этого метода является его простота - информация выкладывается так, чтобы её было легко читать и понимать. Одним из недостатков является то, что он не содержит дополнительной информации, которую могут дать другие методы вычисления.

```
([System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()).GetAllTrustRelationships()
```

Ранее метод **Get-DomainTrust** применялся в PowerView по умолчанию. Недавно я изменил его на LDAP, поскольку этот метод (.NET) не отображает лесные трасты по умолчанию, в отличие от LDAP. Таким образом, чтобы им воспользоваться, вам нужно будет запустить **Get-DomainTrust -NET**.

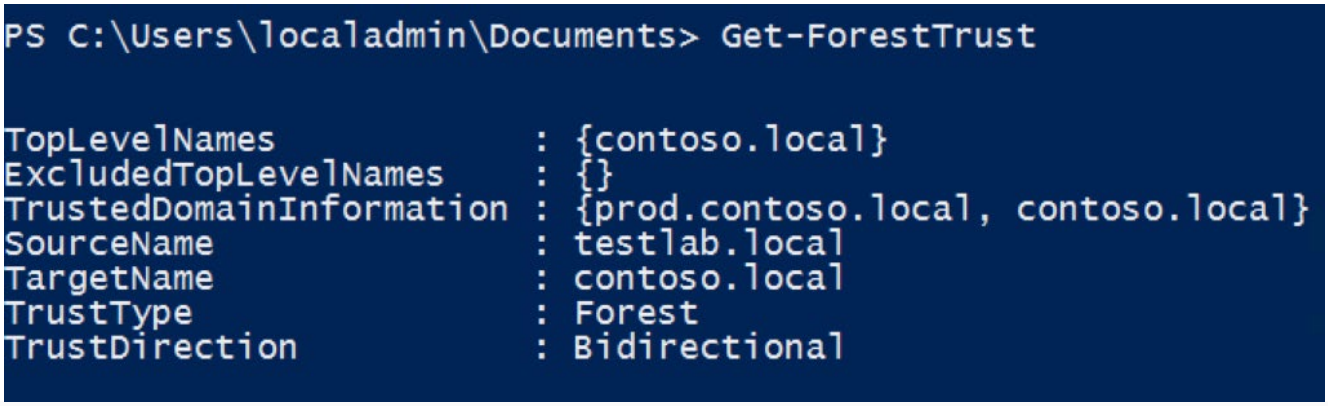
Вот как это будет выглядеть на моём примере настройки домена, при работе из **sub.dev.testlab.local**:

```
PS C:\Users\localadmin\Documents> $ENV:USERDNSDOMAIN
SUB.DEV.TESTLAB.LOCAL
PS C:\Users\localadmin\Documents> Get-DomainTrust -NET

SourceName            TargetName          TrustType TrustDirection
----------            ----------          --------- --------------
sub.dev.testlab.local dev.testlab.local ParentChild  Bidirectional
sub.dev.testlab.local external.local       External       Outbound
```

Трасты леса функционально отличаются от доменных. Поэтому, если вы хотите вычислить любые трасты current forest->forest, нужно использовать [System.DirectoryServices.ActiveDirectory.Forest]. Полученные объекты также имеют свой собственный метод GetAllTrustRelationships(), который будет отображать любые трасты текущего леса доменов:

```
([System.DirectoryServices.ActiveDirectory.Forest]::GetCurrentForest()).GetAllTrustRelationships()
```

Это методы отображения по умолчанию для PowerView-функции **Get-ForestTrust**. Вот как это будет выглядеть на моём примере настройки домена, при работе из **sub.dev.testlab.local**:



Win32API

Вы также можете вычислить доменные трасты с помощью вызова Win32 API-функции DsEnumerateDomainTrusts(), которая возвращает структуру DS_DOMAIN_TRUSTS. Хотя получаемая информация немного сложнее, чем методы .NET, она отображает SID и GUID целевого домена, а также некоторые полезные флаги и атрибуты. Флаги задокументированы здесь, и они сообщат вам направление траста, находится ли траст в том же лесу и т.д. Атрибуты задокументированы здесь в соответствии со спецификацией **TrustAttributes**, и включают в себя такие вещи, как WITHIN_FOREST, NON_TRANSITIVE, FILTER_SIDS и другие. FILTER_SIDS является эквивалентом QUARANTINED_DOMAIN, если вы когда-либо видели эту номенклатуру.

Этот метод можно использовать с помощью **Get-DomainTrust-API** (тот же домен - sub.dev.testlab.local):

```
PS C:\Users\localadmin\Documents> Get-DomainTrust -API

SourceName         : SUB.DEV.TESTLAB.LOCAL
TargetName         : dev.testlab.local
TargetNetbiosName  : DEV
Flags              : IN_FOREST, DIRECT_OUTBOUND, DIRECT_INBOUND
ParentIndex        : 2
TrustType          : UPLEVEL
TrustAttributes    : WITHIN_FOREST
TargetSid          : S-1-5-21-260219439-3323821292-2673346075
TargetGuid         : 3a0e48b9-a7ec-4dc7-b096-902e7e0a83c0

SourceName         : SUB.DEV.TESTLAB.LOCAL
TargetName         : external.local
TargetNetbiosName  : EXTERNAL
Flags              : DIRECT_OUTBOUND
ParentIndex        : 0
TrustType          : UPLEVEL
TrustAttributes    : FILTER_SIDS
TargetSid          : S-1-5-21-2345519959-2041131721-4244767943
TargetGuid         : 00000000-0000-0000-0000-000000000000

SourceName         : SUB.DEV.TESTLAB.LOCAL
TargetName         : testlab.local
TargetNetbiosName  : TESTLAB
Flags              : IN_FOREST, TREE_ROOT
ParentIndex        : 0
TrustType          : UPLEVEL
TrustAttributes    : 0
TargetSid          : S-1-5-21-3283595427-545770840-1968231694
TargetGuid         : 3d8f91b7-44c9-4a98-9cb0-becc0b93c512

SourceName         : SUB.DEV.TESTLAB.LOCAL
TargetName         : sub.dev.testlab.local
TargetNetbiosName  : SUB
Flags              : IN_FOREST, PRIMARY, NATIVE_MODE
ParentIndex        : 0
TargetGuid         : 3d8f91b7-44c9-4a98-9cb0-becc0b93c512
```

Отметим, что это похоже на то, что **nltest.exe** использует со своим флагом **/trusted_domains**:

```
PS C:\Users\localadmin\Documents> nltest /trusted_domains
List of domain trusts:
    0: DEV dev.testlab.local (NT 5) (Forest: 2) (Direct Outbound) (Direct Inbound) ( Attr: 0x20 )
    1: EXTERNAL external.local (NT 5) (Direct Outbound) ( Attr: quarantined )
    2: TESTLAB testlab.local (NT 5) (Forest Tree Root)
    3: SUB sub.dev.testlab.local (NT 5) (Forest: 0) (Primary Domain) (Native)
The command completed successfully
PS C:\Users\localadmin\Documents> _
```

Этот метод также использует BloodHound для вычисления доменных трастов. Его можно запустить новым инжектором SharpHound.ps1, применив следующий синтаксис: **Invoke-BloodHound -CollectionMethod trusts**. Обратите внимание, что

он также может быть использован вместе с **-Domain <foreign.domain.fqdn>** для вычисления чужого траста.

LDAP

Доменные трасты хранятся в Active Directory как «trusted domain objects», и имеют класс **trustedDomain**. Это означает, что вы можете использовать любой тип LDAP-запросов и фильтр LDAP **(objectClass=trustedDomain)**, чтобы узнать информацию о любых доменных трастах, которые там присутствуют.

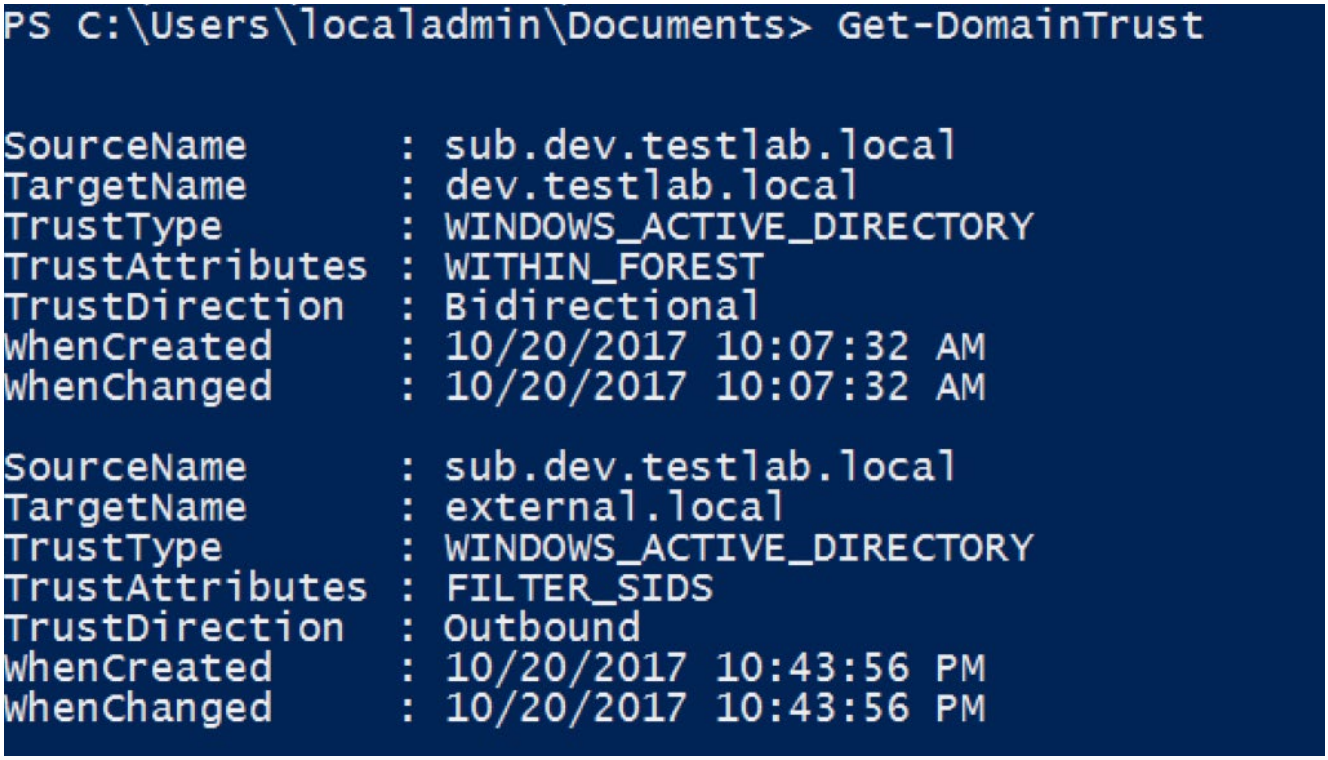
Вот пример dsquery (доступно только на серверах Windows):

```
dsquery * -filter "(objectClass=trustedDomain)" -attr *
```

```
PS C:\Users\localadmin\Documents> dsquery * -filter "(objectClass=trustedDomain)" -attr *
objectClass: top
objectClass: leaf
objectClass: trustedDomain
cn: dev.testlab.local
distinguishedName: CN=dev.testlab.local,CN=System,DC=sub,DC=dev,DC=testlab,DC=local
instanceType: 4
whenCreated: 10/20/2017 10:07:32
whenChanged: 10/20/2017 10:07:32
uSNCreated: 8233
uSNChanged: 8234
showInAdvancedViewOnly: TRUE
name: dev.testlab.local
objectGUID: {57E3017F-9F5C-4955-8A04-40B9EAF33F46}
securityIdentifier: 0x01 0x04 0x00 0x00 0x00 0x00 0x00 0x05 0x15 0x00 0x00 0x00 0x2f 0xa2 0x82 0x0f 0xec 0x7c 0
0x1b 0x06 0x58 0x9f
trustDirection: 3
trustPartner: dev.testlab.local
trustPosixOffset: -2147483648
trustType: 2
trustAttributes: 32
flatName: DEV
objectCategory: CN=Trusted-Domain,CN=Schema,CN=Configuration,DC=testlab,DC=local
isCriticalSystemObject: TRUE
dSCorePropagationData: 10/20/2017 10:09:01
dSCorePropagationData: 01/01/1601 00:00:01
ADsPath: LDAP://secure.sub.dev.testlab.local/CN=dev.testlab.local,CN=System,DC=sub,DC=dev,DC=testlab,DC=local
objectClass: top
objectClass: leaf
objectClass: trustedDomain
cn: external.local
distinguishedName: CN=external.local,CN=System,DC=sub,DC=dev,DC=testlab,DC=local
instanceType: 4
whenCreated: 10/20/2017 22:43:56
whenChanged: 10/20/2017 22:43:56
uSNCreated: 14272
uSNChanged: 14273
showInAdvancedViewOnly: TRUE
```

Равноценный синтаксис в Adwind от Joeware будет выглядеть так: **.\adfind.exe -f objectclass=trusteddomain**.

И, наконец, PowerView, использующий этот LDAP-метод по умолчанию для **Get-DomainTrust**:



Поскольку рассматриваемый LDAP-метод теперь используется по умолчанию для **Get-DomainTrust** в PowerView, я немного раскрою некоторые свойства отклика, которые могут сбить вас с толку.

TrustType:

- **DOWNLEVEL** (0x00000001) – доверенный домен Windows, в котором НЕ РАБОТАЕТ Active Directory. Он отображается как **WINDOWS_NON_ACTIVE_DIRECTORY** в PowerView для тех, кто не очень знаком с терминологией.
- **UPLEVEL** (0x00000002) – доверенный домен Windows, в котором РАБОТАЕТ Active Directory. Он выводится как **WINDOWS_ACTIVE_DIRECTORY** в PowerView для тех, кто не очень знаком с терминологией.
- **MIT** (0x00000003) – доверенный домен с запущенной *nix-системой, совместимый с RFC4120-Kerberos дистрибутивом. Он помечается как MIT.

TrustAttributes:

- **NON_TRANSITIVE** (0x00000001) – этот траст не может быть использован транзитивно. То есть, если DomainA доверяет DomainB, а DomainB доверяет DomainC, то DomainA не будет автоматически доверять DomainC. Кроме того, если траст не является транзитивным, вы не сможете запрашивать какую-либо информацию Active Directory из трастов по цепочке из нетранзитивной точки. Внешние трасты косвенным образом нетранзитивны.
- **UPLEVEL_ONLY** (0x00000002) – только операционные системы от Windows 2000 и выше могут использовать этот траст.

- **QUARANTINED_DOMAIN** (0x00000004) – фильтрация SID включена (подробнее об этом позже). Выводится в PowerView как **FILTER_SIDS**.
- **FOREST_TRANSITIVE** (0x00000008) – траст между корнями двух доменных лесов, работающих с доменом функционального уровня 2003 или выше.
- **CROSS_ORGANIZATION** (0x00000010) – траст к домену или лесу, который не является частью организации, которая добавляет OTHER_ORGANIZATION SID. Это немного странный траст. Я не встречал этот флаг в процессе работы, но, согласно этому посту, он означает, что включена защита безопасности выборочной аутентификации. Для получения дополнительной информации стоит изучить вот этот документ MSDN.
- **WITHIN_FOREST** (0x00000020) – доверенный домен находится в том же лесу, что означает отношение parent->child или cross-link.
- **TREAT_AS_EXTERNAL** (0x00000040) – этот траст следует рассматривать как внешний (в целях ограничения доверия к домену). Согласно документации, *«Если присутствует этот бит, то межлесный траст к домену следует рассматривать как внешний - в целях SID-фильтрации.* ***Трасты между лесами фильтруются более строго, чем внешние****. Этот атрибут ослабляет межлесные доверительные отношения, чтобы приравнять их к внешним»*. Это звучит заманчиво, но я не уверен на все 100% в последствиях для безопасности ¯_(ツ)_/¯ Я дополню этот пост, если всплывёт что-то новое.
- **USES_RC4_ENCRYPTION** (0x00000080) – если TrustType - MIT, этот атрибут указывает, что траст поддерживает ключи RC4.
- **USES_AES_KEYS** (0x00000100) – такое свойство не упомянуто в связанной документации Microsoft. Однако, согласно некоторым документам, которые мне удалось найти в Интернете, оно определяет, что для шифрования KRB TGT используются ключи AES.
- **CROSS_ORGANIZATION_NO_TGT_DELEGATION** (0x00000200) – «*Если установлен этот бит, то билеты, предоставленные в рамках этого траста, НЕ ДОЛЖНЫ быть доверены для делегирования*». Это описано более подробно в [MS-KILE] 3.3.5.7.5 (Междоменный траст и рефералы).
- **PIM_TRUST** (0x00000400) – «*Если этот бит и бит TATE (рассматривается как внешний) установлены, то межлесный траст к домену следует рассматривать как траст управления привилегированными идентификационными данными в целях фильтрации SID*». Согласно [MS-PAC] 4.1.2.2 (Фильтрация SID и трансформация запросов), «*Домен может управляться доменом, находящимся за пределами леса. Доверяющий домен позволяет SIDам своего леса проходить через траст PrivilegedIdentityManagement*». Хотя мне не встречался подобный траст, поддерживаемый только доменами функционального уровня 2012R2 и выше, он также требует дальнейшего изучения :)

Все эти методы могут быть использованы против домена, который доверяет вам (на текущий момент). Это значит, что, если ваш текущий домен имеет двунаправленный траст с доменом FOREIGN или траст является односторонним и входящим (имеется в виду, что у вас есть доступ), вы можете применить эти методы против указанного домена, чтобы найти трасты для ЭТОГО домена. Если вы хотите сделать это через PowerView, просто укажите параметр **-Domain <domain.fqdn>**, более подробно описанный в следующем разделе.

Вычисление данных через трасты с помощью PowerView

В прошлом году я расписал мою полную переделку PowerView. Одним из упомянутых изменений было то, что теперь любая функция **Get-Domain*** использует LDAP. Это значит, что мы можем извлечь указанную информацию из домена, который нам доверяет. Это делается с помощью параметра **-Domain <domain.fqdn>**:

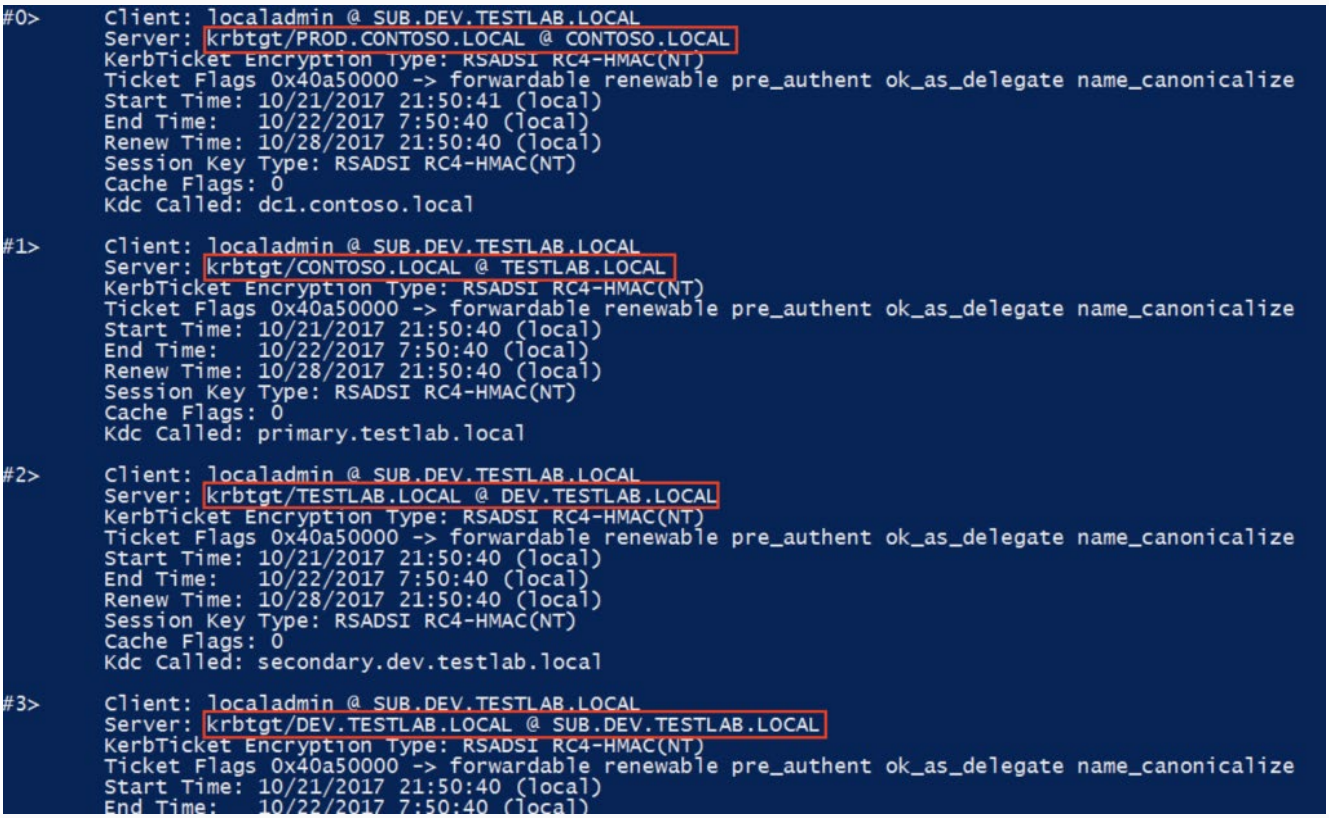
```
PS C:\Users\localadmin\Documents> $ENV:USERDNSDOMAIN
SUB.DEV.TESTLAB.LOCAL
PS C:\Users\localadmin\Documents> Get-DomainComputer -Domain testlab.local


pwdlastset                    : 10/20/2017 2:31:01 AM
logoncount                    : 15
serverreferencebl             : CN=primary,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=t
                                C=local
badpasswordtime               : 12/31/1600 4:00:00 PM
distinguishedname             : CN=primary,OU=Domain Controllers,DC=testlab,DC=local
objectclass                   : {top, person, organizationalPerson, user...}
lastlogontimestamp            : 10/20/2017 2:31:25 AM
name                          : primary
objectsid                     : S-1-5-21-3283595427-545770840-1968231694-1001
samaccountname                : primary$
localpolicyflags              : 0
codepage                      : 0
samaccounttype                : MACHINE_ACCOUNT
whenchanged                   : 10/20/2017 9:36:34 AM
accountexpires                : NEVER
countrycode                   : 0
operatingsystem               : Windows Server 2012 R2 Datacenter
instancetype                  : 4
msdfsr-computerreferencebl    : CN=primary,CN=Topology,CN=Domain System
                                Volume,CN=DFSR-GlobalSettings,CN=System,DC=testlab,DC=local
objectguid                    : ac80d537-65af-49ff-a05f-4f65dc1bd242
operatingsystemversion        : 6.3 (9600)
lastlogoff                    : 12/31/1600 4:00:00 PM
objectcategory                : CN=Computer,CN=Schema,CN=Configuration,DC=testlab,DC=local
dscorepropagationdata         : {10/20/2017 9:30:46 AM, 1/1/1601 12:00:01 AM}
serviceprincipalname          : {Dfsr-12F9A27C-BF97-4787-9364-D31B6C55EB04/primary.testlab.local,
                                ldap/primary.testlab.local/DomainDnsZones.testlab.local,
                                ldap/primary.testlab.local/ForestDnsZones.testlab.local, TERMSRV/primary...}
usncreated                    : 12293
```

Так что же на самом деле происходит там, за ширмой?

Долгое время я думал, что данное действие «отразит» LDAP-запросы через контроллер домена в вашем текущем домене и на контроллеры домена в доверяющем домене. Это был бы отличный способ обойти границы сети - но, к сожалению, я ошибся. На самом деле происходит вот что: перекрёстная ссылка возвращается контроллером домена, с которым вы в данный момент общаетесь, и даёт указание связаться с чужим доменом (т.е. с основным контроллером/PDC для того домена). Если существуют доверительные отношения с чужим доменом, тогда будет возвращён межобластной TGT. Это можно использовать при контакте с чужим доменом.

Из этого можно сделать вывод: если существует сегментация сети между компьютером-источником и PDC доверяющего домена, то вы не сможете получить никаких результатов >_<

Если смотреть из-за ширмы со стороны Kerberos, то мы увидим, что автоматически выдаётся серия межобластных реферальных билетов, и это позволяет нашему пользователю в конечном итоге запросить билет службы LDAP у целевого домена. Если мы используем наш образец архитектуры домена и в данный момент размещаемся в **sub.dev.testlab.local,** то при запросе **prod.contoso.local** вывод klist будет выглядеть так:



На этом примере видно, как межобластные билеты фильтруют цепочку трастов к **testlab.local**, далее к **contoso.local** и, в конечном итоге, к **prod.contoso.local**.

Нанесение доменных трастов на «карту»

Мне знакомы несколько способов, как отобразить «сетку» одного или нескольких трастов, существующих в вашем окружении. Первый способ - через глобальный каталог. Я немного говорил об этом в своей статье «Пентестерское руководство по групповому анализу», но сейчас повторю кое-что из той информации.

Глобальный каталог - это частичная копия всех объектов в лесу Active Directory. То есть в нём содержатся некоторые свойства объекта (но не все). Эти данные реплицируются среди всех контроллеров домена, помеченных как глобальные каталоги для леса. Объекты доверенных доменов реплицируются в глобальном каталоге, и именно поэтому, запустив **Get-DomainTrust -SearchBase "GC://$($ENV:USERDNSDOMAIN)"** через PowerView, мы можем очень быстро

определить каждый внутренний и внешний траст в текущем лесу - но только с помощью анализа трафика к нашему текущему PDC. Вот как это выглядит при запуске вышеназванной функции из домена **sub.dev.testlab.local**:

```
PS C:\Users\localadmin\Documents> Get-DomainTrust -SearchBase "GC://$($ENV:USERDNSDOMAIN)"


SourceName      : sub.dev.testlab.local
TargetName      : dev.testlab.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : WITHIN_FOREST
TrustDirection  : Bidirectional
WhenCreated     : 10/20/2017 10:07:32 AM
WhenChanged     : 10/20/2017 10:07:32 AM

SourceName      : dev.testlab.local
TargetName      : testlab.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : WITHIN_FOREST
TrustDirection  : Bidirectional
WhenCreated     : 10/20/2017 9:47:47 AM
WhenChanged     : 10/20/2017 10:10:10 AM

SourceName      : dev.testlab.local
TargetName      : sub.dev.testlab.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : WITHIN_FOREST
TrustDirection  : Bidirectional
WhenCreated     : 10/20/2017 10:07:32 AM
WhenChanged     : 10/20/2017 10:10:11 AM

SourceName      : testlab.local
TargetName      : dev.testlab.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : WITHIN_FOREST
TrustDirection  : Bidirectional
WhenCreated     : 10/20/2017 9:47:47 AM
WhenChanged     : 10/20/2017 10:10:11 AM

SourceName      : testlab.local
TargetName      : contoso.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : FOREST_TRANSITIVE
```

Здесь намного больше результатов, чем в простом **Get-DomainTrust**!

Второй метод медленнее, но отдача от него будет ещё больше. Поскольку мы можем вычислять любые трасты, которые есть в нашем текущем контексте домена, и посредством ссылок через LDAP можем запрашивать любые **(objectClass=trustedDomain)** объекты из доменов, которые сейчас доверяют нашему домену, то мы можем продолжать эти запросы для получения любых результатов и «ползать» по любым доступным доменам. Домены, помеченные как нетранзитивные, могут подпортить нам статистику, но всё равно количество результатов будет впечатляющее.

Функция PowerView для этого процесса - **Get-DomainTrustMapping** (ранее Invoke-MapDomainTrust). Полученные результаты можно экспортировать в виде файла CSV, направив команду **Get-DomainTrustMapping** на | **Export-CSV -NoTypeInformation trusts.csv**.

Последний способ - через BloodHound/SharpHound. Повторюсь – можно использовать новый инжектор SharpHound.ps1, используя синтаксис **Invoke-BloodHound -CollectionMethod trusts**. Также можно комбинировать с **-Domain <foreign.domain.fqdn>** для вычисления чужих трастов.

Важно помнить, что полученное точное расположение будет зависеть от домена, в котором вы находитесь. Поскольку траст между доменами **external.local** и **sub.dev.testlab.local** является односторонним *нетранзитивным* внешним трастом, то при запросе от **external.local** вы не сможете увидеть трасты, которые есть у **contoso.local**, потому что **sub.dev.testlab.local** не будет переупаковывать ваш TGT в межобластной TGT, который, как мы помним, может быть перенаправлен на любой другой домен. Кроме того, если вы пытаетесь отследить трасты в чужом домене, вам необходимо будет иметь возможность связываться с контроллером чужого домена, который вы запрашиваете. Таким образом, даже если существует транзитивный траст, который позволит вам запрашивать информацию, но при этом сегментация сети не позволяет вам общаться с целевым чужим доменом - вам не повезло.

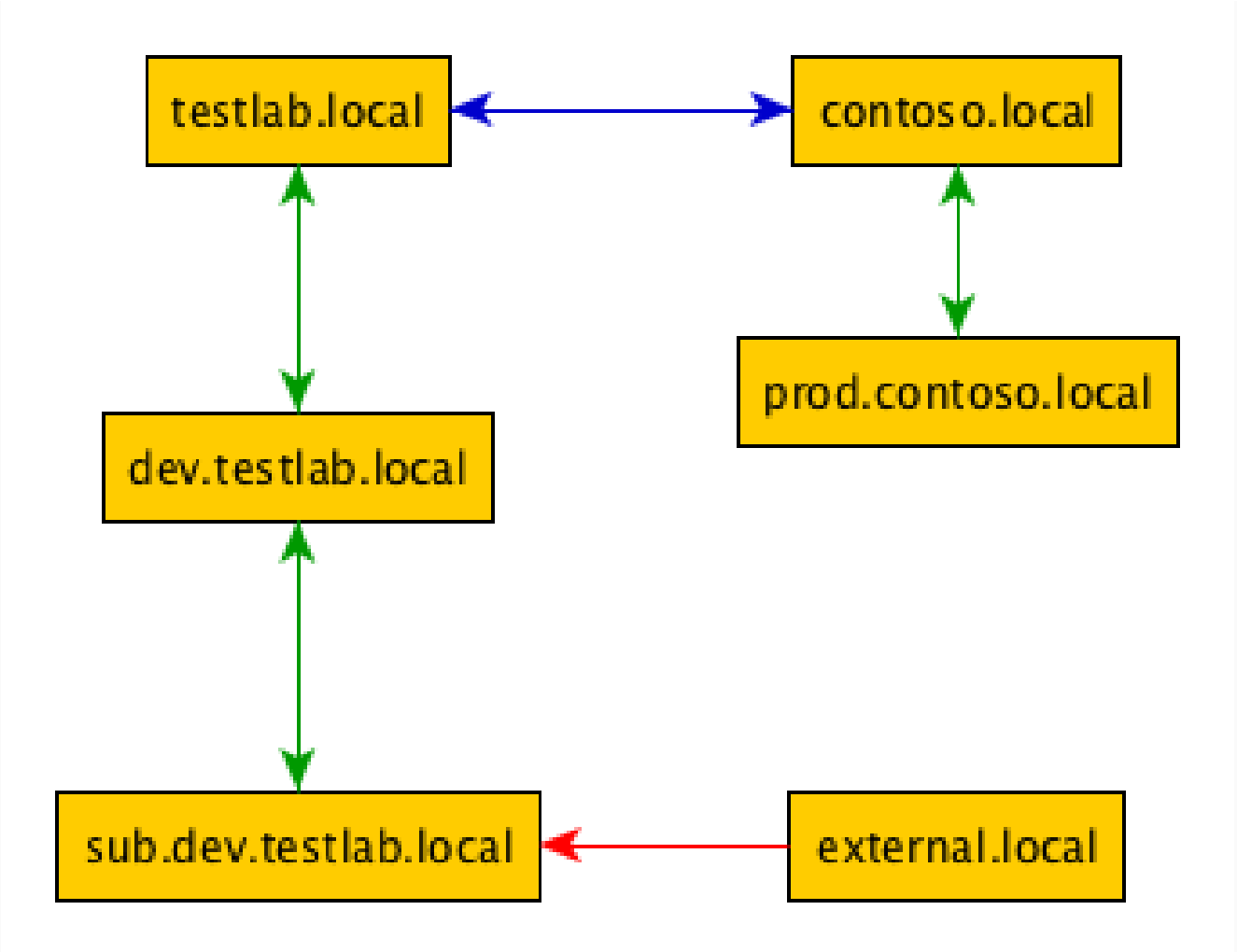
Визуализация доменных трастов

Данные - это одно, а визуализация - это другое. Несколько лет назад один из моих бывших коллег по работе, Джастин Уорнер, создал инструмент под названием DomainTrustExplorer, который может выполнять узловой анализ и визуализацию с помощью отслеженных данных трастов от PowerView.

Поскольку изменились выходные данные траста по умолчанию, и BloodHound уже позаботился об узловом анализе, я переписал проект Джастина в упрощённую форму, которая умеет принимать форматы .CSV. Это - TrustVisualizer:

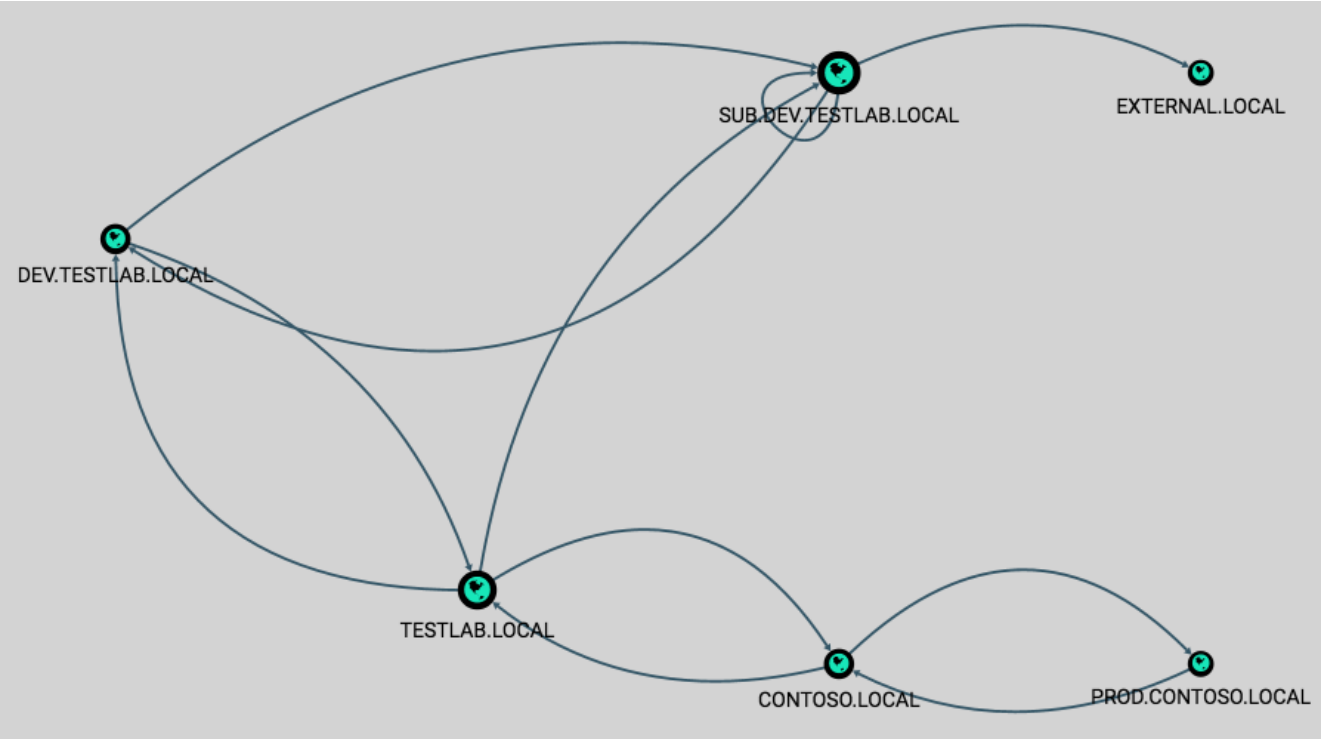
```
root@student:/tmp/TrustVisualizer# ./TrustVisualizer.py trusts.csv

[+] Graphml writte to 'trusts.csv.graphml'

[*] Note: green = within forest, red = external, blue = forest to forest, black
= MIT, violet = unrecognized
```

Полученный graphml можно визуализировать с помощью yEd, как описано здесь. Именно так я и создал предыдущую визуализацию, которую привёл в качестве примера архитектуры траста:



Опять же, в этом графике **зелёные** указатели означают «внутри леса», **красные** – внешние трасты, а **синие** – доверительные отношения между лесами. Как и в случае с @sixdub's DomainTrustExplorer, направления для одностороннего траста означают *направление доступа*, а не *направление траста*.

При использовании SharpHound для сбора данных траста и BloodHound для их визуализации те же самые данные будут выглядеть так:



Определение посторонних отношений

Теперь, когда мы обозначили все доменные трасты, доступные для компьютера, с которого мы производим запрос, следующий шаг планирования атаки затрагивает несколько ветвей, в зависимости от конкретных типов трастов, с которыми мы столкнулись. Эти шаги необходимо выполнить для перехода от одного домена к другому в цепи атаки.

Если следующий домен находится в том же лесу, что и домен, от которого/через который мы начинаем путь, ***и*** мы можем скомпрометировать хэш krbtgt дочернего домена, то можно использовать метод, описанный в разделе **«Трастпокалипсис»**, чтобы скомпрометировать коренной домен леса доменов.

Если мы не можем получить расширенный доступ в текущем дочернем домене, или если следующий шаг на пути атаки траста - это внешний/лесной траст, тогда нам нужно определить, какие пользователи (если они есть) домена, частью которого мы являемся, состоят в группах в целевом домене (или же делаем следующий шаг на пути атаки домена).

К сожалению, этот шаг зависит от многих деталей. Точная природа доверительных отношений, которые ваш текущий домен поддерживает с *доверяющим* доменом, запрашиваемым вами, повлияет на то, какую информацию вы можете получить, и на методы, которые вы можете использовать. Грубо говоря, это вычисление в значительной степени зависит от того, запрашиваете ли вы чужой домен в том же

лесу или по внешнему/межлесному трасту. Я постараюсь как можно подробнее объяснить все тонкости.

Существует три **основных** способа, которыми принципалы безопасности (пользователи/группы) из одного домена могут получить доступ к ресурсам в другом чужом/*доверяющем* домене:

- Их можно добавить в локальные группы на отдельных машинах, то есть в локальную группу «Администраторы» на сервере.
- Их можно добавить в группы в чужом домене. Есть несколько важных моментов, в зависимости от типа траста и области действия группы. Я опишу их ниже.
- Они могут быть добавлены в качестве принципалов в список управления доступом, что наиболее интересно для нас как ACE-принципалов в DACL. Для получения дополнительной информации о списках ACL/DACL/ACE обратитесь к техническому документу «An ACE Up the Sleeve».

Случай 1: Участие в локальной группе

В этом случае нам нужно определить локальное членство одной или нескольких систем через удалённую SAM (SAMR) или через корреляцию GPO. Я не буду подробно останавливаться на этом случае, но отмечу, что в прошлом мы добились успеха на поприще целевого вычисления SAMR высокопроизводительных серверов или контроллеров домена в целях переключения траста. Также для этого есть функция PowerView: **Get-NetLocalGroupMember <server>**. BloodHound сделает всё за вас.

Случай 2: Участие в чужих группах

Такой случай участия в группе немного сложнее. Свойство **member** группы Active Directory и свойство **memberOf** объекта user/group имеют особый тип отношений, называемых связанными атрибутами. Я рассматривал их более подробно в предыдущей статье; в случае со связанными атрибутами, Active Directory определяет значение конкретного атрибута (например, **memberOf**), по значению другого атрибута, называемого прямой ссылкой (например, **member**). Суть в том, что участие в группе в конечном итоге сохраняется в свойстве **member**, и всё это в итоге выходит достаточно сложным. Надеюсь, вскоре мы разберёмся с этой проблемой и раскроем её подробнее. Будет ли отражаться в сохранённом свойстве объекта user/group **memberOf** участие в чужой группе, зависит от природы траста и масштаба чужой группы.

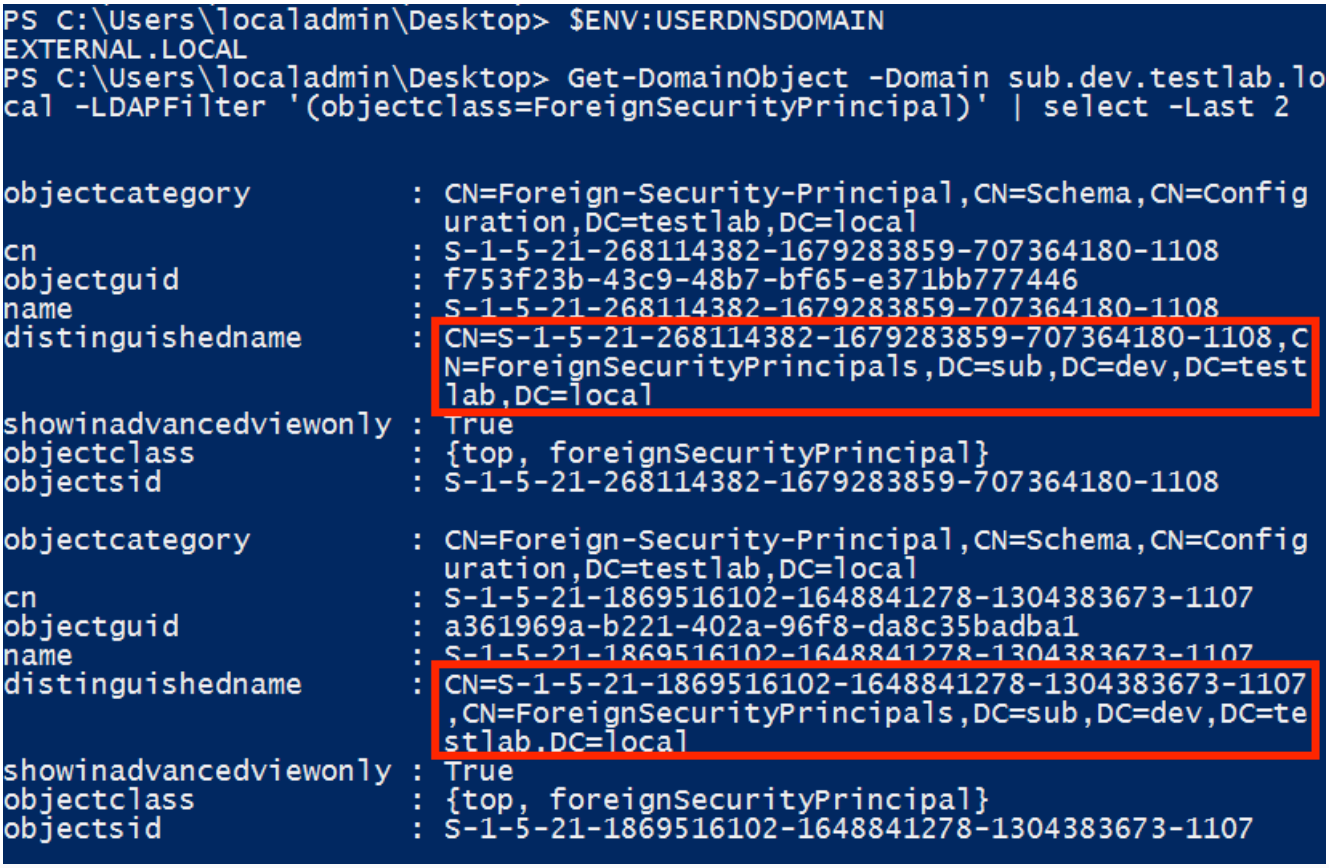
Типы «внешнего» участия можно разбить на три группы:

- **Локальные группы домена.** Участвовать в них могут пользователи между доменами как внутри леса (пользователи в том же лесу, что и группа), так и между лесами (принципалы внешней безопасности).

- **Глобальные группы.** В них не может быть какого-либо членства между доменами даже в одном и том же лесу. Для нас они бесполезны – можно их смело игнорировать!
- **Универсальные группы.** Их участниками могут быть любые пользователи в лесу, но «принципалы внешней безопасности» (то есть пользователи из лесных/внешних трастов) не могут состоять в них.

Если пользователь/группа вложены в какую-либо группу в другом домене того же леса (к примеру, «домен локальный» или «универсальная группа»), то, в зависимости от области действия целевой группы, членство *может* быть обновлено в свойстве **memberOf** пользователя/группы. Группы с «универсальной» областью действия реплицируют свои членства в глобальном каталоге для леса, то есть элемент **memberOf** пользователя будет обновлён. Если область действия группы, в которую добавлен пользователь, является «доменом локальным», то элемент **memberOf** пользователя НЕ будет обновлён (в глобальном каталоге), так как у группы с областью действия «домен локальный» нет своих членств, реплицированных в лесу. Таким образом, единственная возможность с первого взгляда на пользовательский объект определить, состоит ли пользователь в чужой группе, появляется при добавлении его в универсальную группу в одном и том же лесу. Однако это также означает, что, если мы можем связаться с глобальным каталогом леса – у нас легко получится вычислить все эти специфические отношения между доменами.

Если пользователь вложен в группу в домене через лесной/внешний траст, тогда всё будет немного по-другому. Пользователи, существующие через внешние или лесные трасты, всё так же могут добавляться в группы **домена локального** в указанном домене. Такие пользователи отображаются в виде новых записей в **CN=ForeignSecurityPrincipals,DC=domain,DC=com** в домене, к которому они присоединяются, используемых в качестве прокси-сервера, позволяющего добавлять к ресурсам домена идентификаторы внешней безопасности.



Вот как Microsoft объясняет это: *«Когда устанавливается траст между доменом в лесу и доменом вне этого леса, принципалы безопасности из внешнего домена могут получить доступ к ресурсам во внутреннем домене. Active Directory создаёт объект принципала внешней безопасности во внутреннем домене для представления каждого принципала безопасности из доверенного внешнего домена. Эти принципалы внешней безопасности* ***могут состоять в группах домена локального*** *во внутреннем домене»*. Если вы не знаете, что такое «домен локальный» или «групповое определение», почитайте мою предыдущую статью на эту тему.

Слишком много букв ☺ Как я понимаю, эти принципалы внешней безопасности действуют как псевдонимы для «реального» пользователя, внешнего по отношению к домену/лесу, и это принципал внешней безопасности фактически добавляется в группы в целевом домене. SID данного принципала внешней безопасности идентичен SID внешнего пользователя, что упрощает фильтрацию в дальнейшем.

Случай 3: Внешние ACL-принципалы

К счастью, по большей части свойство **ntSecurityDescriptor** объектов Active Directory (1) доступно любому пользователю, прошедшему аутентификацию в домене, и (2) реплицировано в глобальном каталоге. Это означает, что, исходя из текущего контекста домена, вы можете запросить списки DACL для всех объектов в

доверяющем домене и отфильтровать любые записи ACE, где принципал внешней безопасности имеет данное право на вычисляемый вами объект.

Вы можете воспользоваться функцией PowerView **Get-DomainObjectACL -Domain <domain.fqdn>** для получения этих ACE. Но для того, чтобы найти междоменные отношения DACL, вам потребуется отфильтровать принципалов/идентификаторов безопасности, не совпадающих с SID запрашиваемого вами домена. Я расскажу об этом в будущей статье **PowerView PowerUsage**.

Оперативное руководство

Примечание: я также разберу все необходимые шаги позже в этой статье в разделе **«Тематическое исследование»** - на тот случай, если что-то из написанного здесь останется неясным.

Если вы сейчас находитесь в дочернем домене в лесу и *У ВАС ЕСТЬ* расширенные права доступа в упомянутом дочернем домене, перейдите в раздел **«Трастпокалипсис»**.

Если вы сейчас находитесь в дочернем домене в лесу и *У ВАС НЕТ* расширенных прав доступа в указанный дочерний домен, то стоит воспользоваться функцией **Get-DomainForeignUser** в PowerView, чтобы вычислить пользователей, входящих в группы за пределами текущего домена пользователя. Это «исходящий» доступ домена, т. е. его формируют пользователи/группы, у которых может быть какой-либо доступ к другим группам домена *в одном и том же лесу*. Эта функция хороша также для «картографирования» других отношений пользователя/группы внутри домена леса:

```
PS C:\Users\localadmin\Documents> $ENV:USERDNSDOMAIN
SUB.DEV.TESTLAB.LOCAL
PS C:\Users\localadmin\Documents> Get-DomainForeignUser


UserDomain              : SUB.DEV.TESTLAB.LOCAL
UserName                : subuser1
UserDistinguishedName   : CN=subuser1,CN=Users,DC=sub,DC=dev,DC=testlab,DC=local
GroupDomain             : dev.testlab.local
GroupName               : DevUniversalGroup
GroupDistinguishedName  : CN=DevUniversalGroup,CN=Users,DC=dev,DC=testlab,DC=loc
                          al
```

Если вы нацелены на внешний/лесной домен или целевой домен в том же лесу, можно задействовать функцию **Get-DomainForeignGroupMember -Domain <target.domain.fqdn>** в PowerView. Она вычисляет *группы* в целевом домене, в которых состоят *пользователи/группы*, в нём не находящиеся. Это «входящий»

доступ домена, т.е. его образуют группы целевого домена с входящими отношениями участия:

```
PS C:\Users\localadmin\Documents> Get-DomainForeignGroupMember -Domain dev.testl
ab.local | Select -Last 3


GroupDomain             : dev.testlab.local
GroupName               : DevDomainLocalGroup
GroupDistinguishedName  : CN=DevDomainLocalGroup,CN=Users,DC=dev,DC=testlab,DC=
                          local
MemberDomain            : dev.testlab.local
MemberName              : S-1-5-21-3718572571-2675599208-1761761885-1108
MemberDistinguishedName : CN=S-1-5-21-3718572571-2675599208-1761761885-1108,CN=
                          ForeignSecurityPrincipals,DC=dev,DC=testlab,DC=local

GroupDomain             : dev.testlab.local
GroupName               : DevDomainLocalGroup
GroupDistinguishedName  : CN=DevDomainLocalGroup,CN=Users,DC=dev,DC=testlab,DC=
                          local
MemberDomain            : sub.dev.testlab.local
MemberName              : subuser1
MemberDistinguishedName : CN=subuser1,CN=Users,DC=sub,DC=dev,DC=testlab,DC=loca
                          l

GroupDomain             : dev.testlab.local
GroupName               : DevUniversalGroup
GroupDistinguishedName  : CN=DevUniversalGroup,CN=Users,DC=dev,DC=testlab,DC=lo
                          cal
MemberDomain            : sub.dev.testlab.local
MemberName              : subuser1
MemberDistinguishedName : CN=subuser1,CN=Users,DC=sub,DC=dev,DC=testlab,DC=loca
                          l
```

Также, к счастью для нас, принципалы внешней безопасности реплицированы в глобальном каталоге, как и объекты доверенного домена (о них говорится в разделе **«Нанесение доменных трастов на «карту»**). Поэтому, если вы хотите быстро вычислить всех принципалов внешней безопасности (это любые входящие чужие группы/пользователи), которые являются участниками групп в домене внутри текущего/целевого леса, можно запросить любой глобальный каталог через фильтр LDAP **'(objectclass=foreignSecurityPrincipal)'**. И поскольку эти внешние принципалы могут добавляться только в группы с областью домена локального, мы можем извлечь домен, к которому был добавлен чужой пользователь, из distinguishedname, запросить этот домен непосредственно для групп локальной области домена с участниками, полагая, что у нас есть некоторый тип прямого или транзитивного траста с этим целевым доменом. Это позволит нам сравнить членство каждой из этих групп домена локального со списком чужих пользователей:

```
$ForeignUsers = Get-DomainObject -Properties objectsid,distinguishedname -SearchBase "GC://sub.dev.testlab.local" -LDAPFilter '(objectclass=foreignSecurityPrincipal)' | ? {$_.objectsid -match '^S-1-5-.*-[1-9]\d{2,}$'} | Select-Object -ExpandProperty distinguishedname

$Domains = @{}
```

```
$ForeignMemberships = ForEach($ForeignUser in $ForeignUsers) {

    # extract the domain the foreign user was added to

    $ForeignUserDomain = $ForeignUser.SubString($ForeignUser.IndexOf('DC=')) -replace 'DC=','' -replace ',','.'

    # check if we've already enumerated this domain

    if (-not $Domains[$ForeignUserDomain]) {

        $Domains[$ForeignUserDomain] = $True

        # enumerate all domain local groups from the given domain that have any membership set

        Get-DomainGroup -Domain $ForeignUserDomain -Scope DomainLocal -LDAPFilter '(member=*)' -Properties distinguishedname,member | ForEach-Object {

            # check if there are any overlaps between the domain local groups and the foreign users

            if ($($_.member | Where-Object {$ForeignUsers  -contains $_})) {

                $_

            }

        }

    }

}


$ForeignMemberships | fl
```

Смотрите **rawgc_foreign_local_groups.ps1** на **GitHub**

```
PS C:\Users\localadmin\Documents> $ForeignMemberships | fl


distinguishedname : CN=SubDomainLocalGroup,CN=Users,DC=sub,DC=dev,DC=testlab,DC
                    =local
member            : {CN=S-1-5-21-1869516102-1648841278-1304383673-1107,CN=Forei
                    gnSecurityPrincipals,DC=sub,DC=dev,DC=testlab,DC=local, CN=
                    S-1-5-21-268114382-1679283859-707364180-1108,CN=ForeignSecu
                    rityPrincipals,DC=sub,DC=dev,DC=testlab,DC=local}

distinguishedname : CN=DevDomainLocalGroup,CN=Users,DC=dev,DC=testlab,DC=local
member            : {CN=S-1-5-21-3718572571-2675599208-1761761885-1108,CN=Forei
                    gnSecurityPrincipals,DC=dev,DC=testlab,DC=local,
                    CN=subuser1,CN=Users,DC=sub,DC=dev,DC=testlab,DC=local}

distinguishedname : CN=TestlabDomainLocalGroup,CN=Users,DC=testlab,DC=local
member            : {CN=S-1-5-21-1869516102-1648841278-1304383673-1107,CN=Forei
                    gnSecurityPrincipals,DC=testlab,DC=local, CN=S-1-5-21-37185
                    72571-2675599208-1761761885-1108,CN=ForeignSecurityPrincipa
                    ls,DC=testlab,DC=local,
                    CN=devuser1,CN=Users,DC=dev,DC=testlab,DC=local}
```

Мы быстро получаем разметку всех вложенных отношений чужих пользователей/групп, входящих в наш текущий (или целевой) лес.

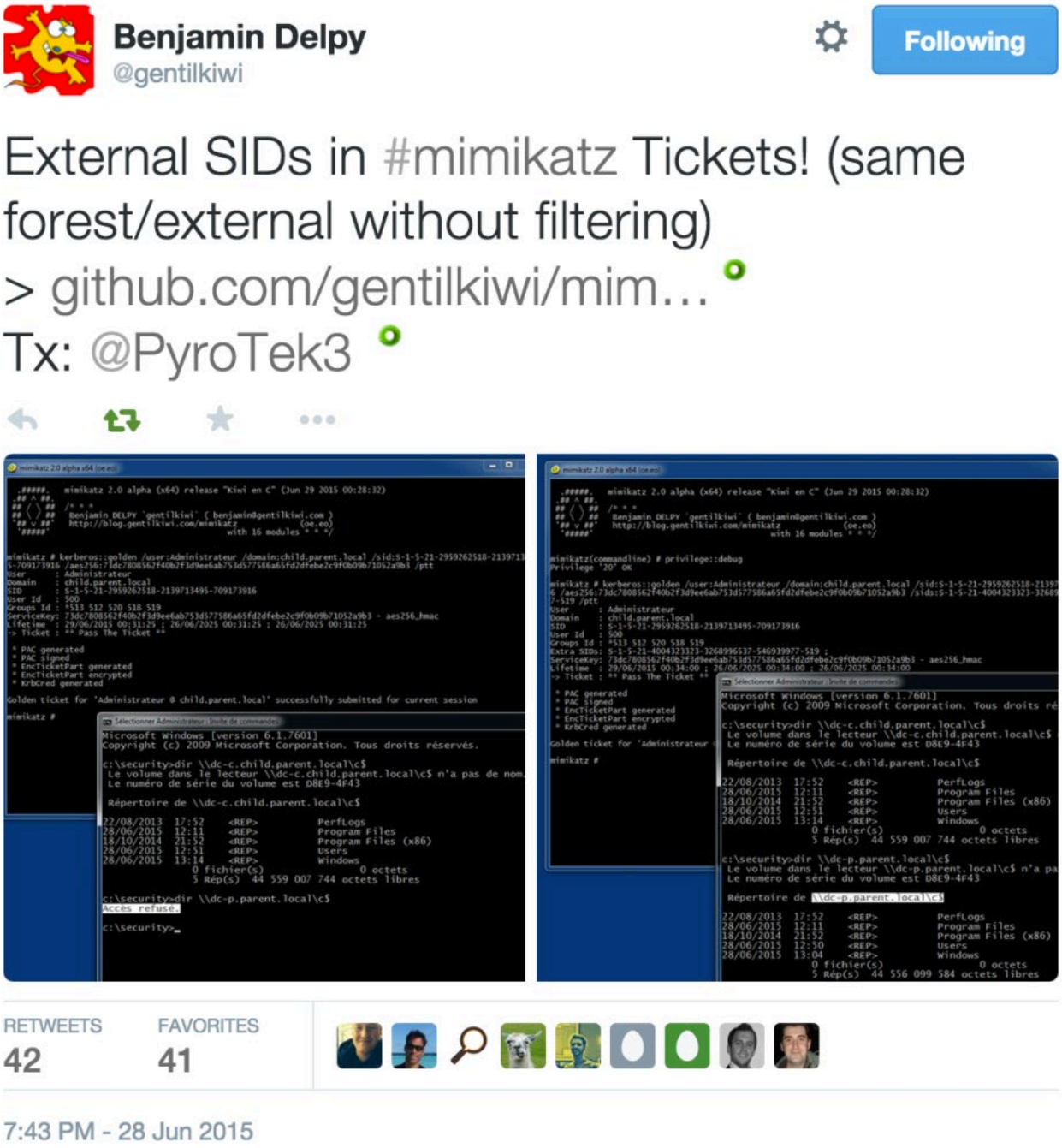
Если вы применяете BloodHound с его новым инжектором SharpHound, то можно использовать также и **-Domain <domain.fqdn>** с инжектором в сочетании с параметрами **-CollectionMethod** для «Group», «LocalGroup» и/или «ACL». BloodHound моделирует узлы пользователя/группы с синтаксисом **name@<domain.fqdn>** в схеме. Это избавляет от необходимости выполнять сложную аналитику для того, чтобы извлечь эти отношения после сбора данных. Если **user@dev.testlab.local** является членом **group@testlab.local**, то отношение memberOf моделируется автоматически. Если это отношение вложенных групп появляется на каких-либо путях атаки, оно будет автоматически включено в ваш график без дополнительных усилий.

Стало понятно, верно? :) Это сложная тема - и, если вы с ней не знакомы, тогда перечитайте предыдущий раздел несколько раз. Пока не станет понятно, как «выманить» эти междоменные отношения. Почитайте раздел **«Тематическое исследование»** для реалистичного знакомства с эталонной архитектурой, которую я использовал в этой статье.

Трастпокалипсис – SID взламывает трасты внутри леса

Это одна из самых потрясающих вещей в области безопасности, которые я открыл для себя за последние несколько лет. Точно так же, как многие помнят момент, когда они впервые увидели, как Mimikatz извлекает из памяти незашифрованный пароль, воспоминание о том, как до меня дошло, что же влечёт за собой эта атака, впечаталось в мою память навсегда.

Это началось, как и многие из моих крышесносящих моментов, с увиденного в июне 2015 года твита Бенджамина Делпи и, поначалу, непонимания его последствий:



Пообщавшись в чате с Бенджамином, чтобы подтвердить свои догадки о последствиях, я получил ответ: «Жалко вашу голову :)»

Всё это произошло благодаря разработке, над которой трудились Бенджамин и Шон Меткалф, чтобы сделать золотые билеты ещё более «золотыми». После их релиза я написал о нём в августе 2015 года статью под названием «Трастпокалипсис».

До этой работы наша стратегия была в том, чтобы разметить на «карте» членство чужих пользователей/групп и переходить от дочернего траста к корню леса «вручную», что часто является кропотливым процессом в больших средах с множеством доменов. Как описано в разделе **«Стратегия атаки траста»**, мы всегда интерпретировали траст как «перетекание» от корня леса к дочерним доменам с использованием группы «Администраторы предприятия». Несмотря на многолетние заявления Microsoft, что «*лес является границей безопасности для Active Directory*», атаки на домены внутри леса известны по крайней мере с 2005 года.

Но сначала, для полного понимания, я должен объяснить **sidHistory**, механизм безопасности фильтрации SID и значение всего этого для доменов внутри леса.

Атрибут **sidHistory** появился в Windows 2000 Active Directory – его «миссией» было облегчение миграции пользователей из одного домена в другой. Если пользователь мигрирует, его прежний идентификатор безопасности (SID), вместе с SIDами любой из групп, в которых он раньше участвовал, может быть дополнительно добавлен в

атрибут sidHistory его новой учётной записи. Когда новый пользователь пытается получить доступ к ресурсу, «*если SID или история SID совпадают, доступ к ресурсу предоставляется или запрещается в соответствии с допуском, указанным в ACL.*» То есть любой SID группы/прежнего пользователя, который установлен в свойстве sidHistory пользователя, предоставляет им доступ, *как если бы они были тем пользователем или участником тех групп*.

Благодаря тому, как трасты работают в лесу Active Directory, свойство **sidHistory** («ExtraSids» в PAC) учитывается в доменах леса, поскольку эти SIDы не отфильтровываются в междоменных ссылках защитой «SID Filtering». Таким образом, любой пользователь в дочернем домене, у которого **sidHistory/ExtraSids** настроены, скажем, на SID «Администраторы предприятия» (группа, существующая только в корне леса), будет эффективно функционировать *так, как если бы он был администратором предприятия*. Поскольку Microsoft знает, что это проблема, и широкой аудитории об этом стало известно, по крайней мере, со времени выхода этой статьи 2005 года об ITPro Windows, и почти наверняка было известно ещё раньше, **sidHistory** сейчас - защищённый атрибут, и его чрезвычайно трудно изменить.

Ранее его атака была довольно сложным процессом, включавшим в себя изменение **sidHistory** в базе данных Active Directory (ntds.dit) соответствующего домена. Более подробно о том, почему и как это работает, читайте в разделе **«Эпилог: фильтрация SID»**.

ВОТ ПОЧЕМУ ЛЕС – «ГРАНИЦА ТРАСТА» В ACTIVE DIRECTORY, А НЕ В ДОМЕНЕ!

Бенджамин и Шон поняли, что с введением золотых билетов Mimikatz злоумышленник может настроить раздел **ExtraSids** структуры KERB_VALIDATION_INFO, созданной для билета (той, что «*определяет информацию для входа и авторизации пользователя, предоставляемую DC*»). Раздел **ExtraSids** описан как «*Указатель списка структур KERB_SID_AND_ATTRIBUTES, которые содержат список SID, соответствующих группам в доменах, отличных от домена учётной записи, к которой принадлежит принципал*» (структура KERB_SID_AND_ATTRIBUTES определена здесь).

Что всё это значит? Если злоумышленник скомпрометирует права «Администратора домена» (или эквивалентные им – по сути любую учётную запись, которая может использовать DCSync;) в ЛЮБОМ дочернем домене в лесу за какие-нибудь пять минут, то он может заполучить хэш krbtgt дочернего домена, и тогда **/sids:<sid_of_enterprise_admins_in_forest_root>** может быть добавлен в созданный билет Mimikatz без изменения базы данных Active Directory. Это даст злоумышленнику возможность «взломать» доверительные отношения леса и скомпрометировать корневой домен леса.

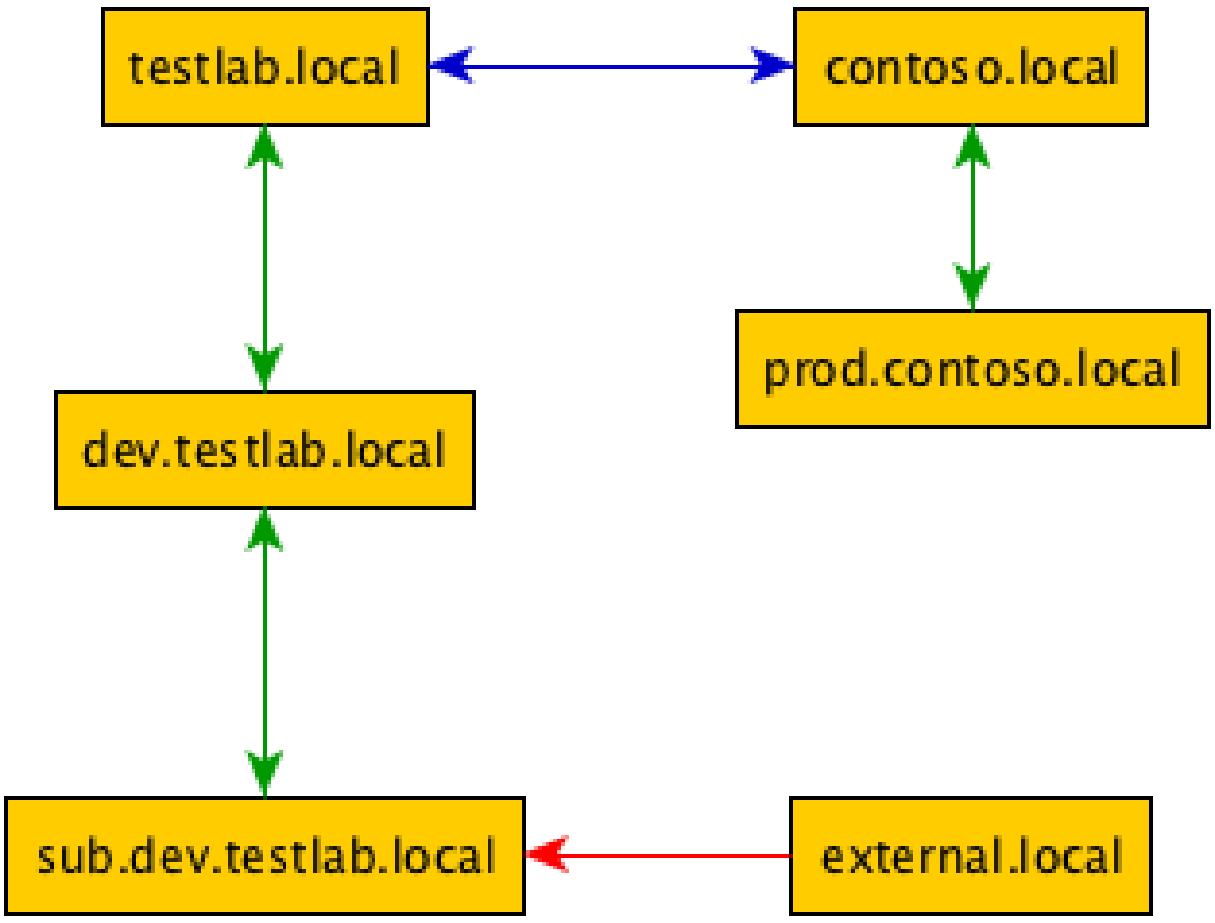
Если вы слышите об этой технике впервые, то - процитирую Бенджамина - «Жалко вашу голову. :)».

Для нашей стратегии оперативной атаки это означает следующее: если мы сейчас находимся в дочернем домене и можем скомпрометировать доступ DCSync/DA, или если мы можем скомпрометировать этот уровень доступа в любом дочернем домене в нашей цепи атаки, мы можем пренебречь нудным вычислением чужих отношений для взлома траста, чтобы тут же скомпрометировать корень леса. Мы сделали это на месте - и да, это круто! :) Для оперативного совета/руководства смотрите мою предыдущую статью, «Трастпокалипсис», от 2015 года.

Это сработает только для переключения между трастами ***в лесу***, но не годится для внешних или межлесных трастов из-за фильтрации SID, более детально описанной в разделе **«Эпилог: фильтрация SID»** в конце этой статьи.

Тематическое исследование

Итак, давайте снова рассмотрим примерную диаграмму траста:



Допустим, мы заполучили аккаунт в **external.local**. Поскольку **sub.dev.testlab.local** *доверяет* **external.local**, **external.local** может запрашивать информацию из **sub.dev.testlab.local**, тогда как SUB не может делать то же самое с EXTERNAL. Из внешнего контекста мы можем запрашивать трасты, которые есть в SUB:

```
PS C:\Users\localadmin\Desktop> $Env:USERDNSDOMAIN
EXTERNAL.LOCAL
PS C:\Users\localadmin\Desktop> Get-DomainTrust


SourceName      : external.local
TargetName      : sub.dev.testlab.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : FILTER_SIDS
TrustDirection  : Inbound
WhenCreated     : 10/27/2017 6:10:04 AM
WhenChanged     : 10/27/2017 6:10:04 AM



PS C:\Users\localadmin\Desktop> Get-DomainTrust -Domain sub.dev.testlab.local


SourceName      : sub.dev.testlab.local
TargetName      : dev.testlab.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : WITHIN_FOREST
TrustDirection  : Bidirectional
WhenCreated     : 10/27/2017 5:00:12 AM
WhenChanged     : 10/27/2017 5:00:12 AM

SourceName      : sub.dev.testlab.local
TargetName      : external.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : FILTER_SIDS
TrustDirection  : Outbound
WhenCreated     : 10/27/2017 6:10:04 AM
WhenChanged     : 10/27/2017 6:10:04 AM
```

Но это даст результат только в виде прямых трастов, которые есть у **sub.dev.testlab** с другими доменами (**dev.testlab.local** и **external.local**). О, если бы мы могли запросить глобальный каталог (не всегда доступный) из **sub.dev.testlab** и вычислить все доменные трасты во всём лесу!

```
PS C:\Users\localadmin\Documents>
PS C:\Users\localadmin\Documents> Get-DomainTrust -SearchBase "gc://sub.dev.testlab.local"


SourceName      : sub.dev.testlab.local
TargetName      : dev.testlab.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : WITHIN_FOREST
TrustDirection  : Bidirectional
WhenCreated     : 10/20/2017 10:07:32 AM
WhenChanged     : 10/20/2017 10:07:32 AM

SourceName      : dev.testlab.local
TargetName      : testlab.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : WITHIN_FOREST
TrustDirection  : Bidirectional
WhenCreated     : 10/20/2017 9:47:47 AM
WhenChanged     : 10/20/2017 10:10:10 AM

SourceName      : dev.testlab.local
TargetName      : sub.dev.testlab.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : WITHIN_FOREST
TrustDirection  : Bidirectional
WhenCreated     : 10/20/2017 10:07:32 AM
WhenChanged     : 10/20/2017 10:10:11 AM

SourceName      : testlab.local
TargetName      : dev.testlab.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : WITHIN_FOREST
TrustDirection  : Bidirectional
WhenCreated     : 10/20/2017 9:47:47 AM
WhenChanged     : 10/20/2017 10:10:11 AM

SourceName      : testlab.local
TargetName      : contoso.local
TrustType       : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes : FOREST_TRANSITIVE
```

Заметьте: так как это односторонний нетранзитивный внешний траст к **sub.dev.testlab.local**, мы не можем запрашивать трасты, которые есть у **contoso.local** из контекста EXTERNAL, поскольку наш трафик Kerberos будет направлен не так, как надо. Вот что обычно означает эта ошибка, если вы с ней столкнулись:

```
PS C:\Users\localadmin\Documents> Get-DomainTrust -Domain contoso.local
Exception calling "FindAll" with "0" argument(s): "A referral was returned from the server.
"
At C:\Users\localadmin\Documents\powerview.ps1:19676 char:24
+                 else { $Results = $TrustSearcher.FindAll() }
+                        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [], MethodInvocationException
    + FullyQualifiedErrorId : DirectoryServicesCOMException

PS C:\Users\localadmin\Documents> _
```

Итак, отсюда нам тогда следует запустить **Get-DomainForeignGroupMember -Domain sub.dev.testlab.local**, чтобы увидеть, есть ли участники в EXTERNAL в каких-либо группах в SUB:

```
PS C:\Users\localadmin\Desktop> $ENV:USERDNSDOMAIN
EXTERNAL.LOCAL
PS C:\Users\localadmin\Desktop> Get-DomainForeignGroupMember -Domain sub.de
v.testlab.local | select -last 2


GroupDomain             : sub.dev.testlab.local
GroupName               : SubDomainLocalGroup
GroupDistinguishedName  : CN=SubDomainLocalGroup,CN=Users,DC=sub,DC=dev,DC
                          =testlab,DC=local
MemberDomain            : sub.dev.testlab.local
MemberName              : S-1-5-21-1869516102-1648841278-1304383673-1107
MemberDistinguishedName : CN=S-1-5-21-1869516102-1648841278-1304383673-110
                          7,CN=ForeignSecurityPrincipals,DC=sub,DC=dev,DC=
                          testlab,DC=local

GroupDomain             : sub.dev.testlab.local
GroupName               : SubDomainLocalGroup
GroupDistinguishedName  : CN=SubDomainLocalGroup,CN=Users,DC=sub,DC=dev,DC
                          =testlab,DC=local
MemberDomain            : sub.dev.testlab.local
MemberName              : S-1-5-21-268114382-1679283859-707364180-1108
MemberDistinguishedName : CN=S-1-5-21-268114382-1679283859-707364180-1108,
                          CN=ForeignSecurityPrincipals,DC=sub,DC=dev,DC=te
                          stlab,DC=local



PS C:\Users\localadmin\Desktop> "S-1-5-21-268114382-1679283859-707364180-11
08" | ConvertFrom-SID
EXTERNAL\externaluser1
```

С этого места мы попытаемся скомпрометировать целевой аккаунт, чтобы взломать траст к **sub.dev.testlab.local**. Команда **Get-DomainForeignUser** будет здесь бесполезна из-за предостережений о репликации связанных значений и доверенных, описанных в разделе **«Определение посторонних отношений»**.

Однако, поскольку **external.local -> sub.dev.testlab.local** является внешним отношением доверия, оно скрытно нетранзитивно, поэтому не может запрашивать участие в группе домена локального для **dev.testlab.local** или **testlab.local**.

Если бы мы затем могли скомпрометировать учётные данные администратора домена (или равноценные им) в файле **sub.dev.testlab.local**, у нас бы получилось создать золотой билет sidHistory-trust-hopping, как описано в разделе **«Трастпокалипсис»**, чтобы скомпрометировать домен корня леса **testlab.local**. Если нам не удалось обеспечить расширенный доступ, тогда следует запустить **Get-DomainForeignUser**, чтобы увидеть, есть ли у пользователей из **sub.dev.testlab.local** доступ к другим группам в лесу. Опять же, помните, что предыдущая информация об участии в *универсальных* группах только для определения области действия будет отражена здесь:

```
PS C:\Users\localadmin\Documents> $ENV:USERDNSDOMAIN
SUB.DEV.TESTLAB.LOCAL
PS C:\Users\localadmin\Documents> Get-DomainForeignUser


UserDomain              : SUB.DEV.TESTLAB.LOCAL
UserName                : subuser1
UserDistinguishedName   : CN=subuser1,CN=Users,DC=sub,DC=dev,DC=testlab,DC=local
GroupDomain             : dev.testlab.local
GroupName               : DevUniversalGroup
GroupDistinguishedName  : CN=DevUniversalGroup,CN=Users,DC=dev,DC=testlab,DC=loc
                          al
```

Нам также следует запустить **Get-DomainForeignGroupMember -Domain dev.testlab.local** и **Get-DomainForeignGroupMember -Domain testlab.local**, чтобы увидеть, что группы в других доменах леса имеют «входящий» доступ:

```
PS C:\Users\localadmin\Documents> Get-DomainForeignGroupMember -Domain dev.testl
ab.local | select -last 2


GroupDomain             : dev.testlab.local
GroupName               : DevDomainLocalGroup
GroupDistinguishedName  : CN=DevDomainLocalGroup,CN=Users,DC=dev,DC=testlab,DC=
                          local
MemberDomain            : sub.dev.testlab.local
MemberName              : subuser1
MemberDistinguishedName : CN=subuser1,CN=Users,DC=sub,DC=dev,DC=testlab,DC=loca
                          l

GroupDomain             : dev.testlab.local
GroupName               : DevUniversalGroup
GroupDistinguishedName  : CN=DevUniversalGroup,CN=Users,DC=dev,DC=testlab,DC=lo
                          cal
MemberDomain            : sub.dev.testlab.local
MemberName              : subuser1
MemberDistinguishedName : CN=subuser1,CN=Users,DC=sub,DC=dev,DC=testlab,DC=loca
                          l
```

```
PS C:\Users\localadmin\Documents> Get-DomainForeignGroupMember -Domain testlab.l
ocal | select -last 2


GroupDomain             : testlab.local
GroupName               : TestlabDomainLocalGroup
GroupDistinguishedName  : CN=TestlabDomainLocalGroup,CN=Users,DC=testlab,DC=loc
                          al
MemberDomain            : dev.testlab.local
MemberName              : devuser1
MemberDistinguishedName : CN=devuser1,CN=Users,DC=dev,DC=testlab,DC=local

GroupDomain             : testlab.local
GroupName               : TestlabUniversalGroup
GroupDistinguishedName  : CN=TestlabUniversalGroup,CN=Users,DC=testlab,DC=local
MemberDomain            : dev.testlab.local
MemberName              : devuser1
MemberDistinguishedName : CN=devuser1,CN=Users,DC=dev,DC=testlab,DC=local
```

Однажды/если бы у нас получилось скомпрометировать, частично или полностью, корень леса **testlab.local** с помощью любого из предыдущих подходов, нам затем следовало бы запустить **Get-DomainForeignGroupMember -Domain**

contoso.local и **Get-DomainForeignGroupMember -Domain prod.contoso.local**, чтобы проверить, были ли в лесу TESTLAB какие-либо пользователи, имеющие членство в чужих группах в лесу CONTOSO.

```
PS C:\Users\localadmin\Documents> Get-DomainForeignGroupMember -Domain contoso.l
ocal


GroupDomain             : contoso.local
GroupName               : ContosoDomainLocalGroup
GroupDistinguishedName  : CN=ContosoDomainLocalGroup,CN=Users,DC=contoso,DC=loc
                          al
MemberDomain            : contoso.local
MemberName              : S-1-5-21-1892815843-4059891285-1948810830-1107
MemberDistinguishedName : CN=S-1-5-21-1892815843-4059891285-1948810830-1107,CN=
                          ForeignSecurityPrincipals,DC=contoso,DC=local

GroupDomain             : contoso.local
GroupName               : ContosoDomainLocalGroup
GroupDistinguishedName  : CN=ContosoDomainLocalGroup,CN=Users,DC=contoso,DC=loc
                          al
MemberDomain            : contoso.local
MemberName              : S-1-5-21-63204185-2989465941-3437209448-1107
MemberDistinguishedName : CN=S-1-5-21-63204185-2989465941-3437209448-1107,CN=Fo
                          reignSecurityPrincipals,DC=contoso,DC=local
```

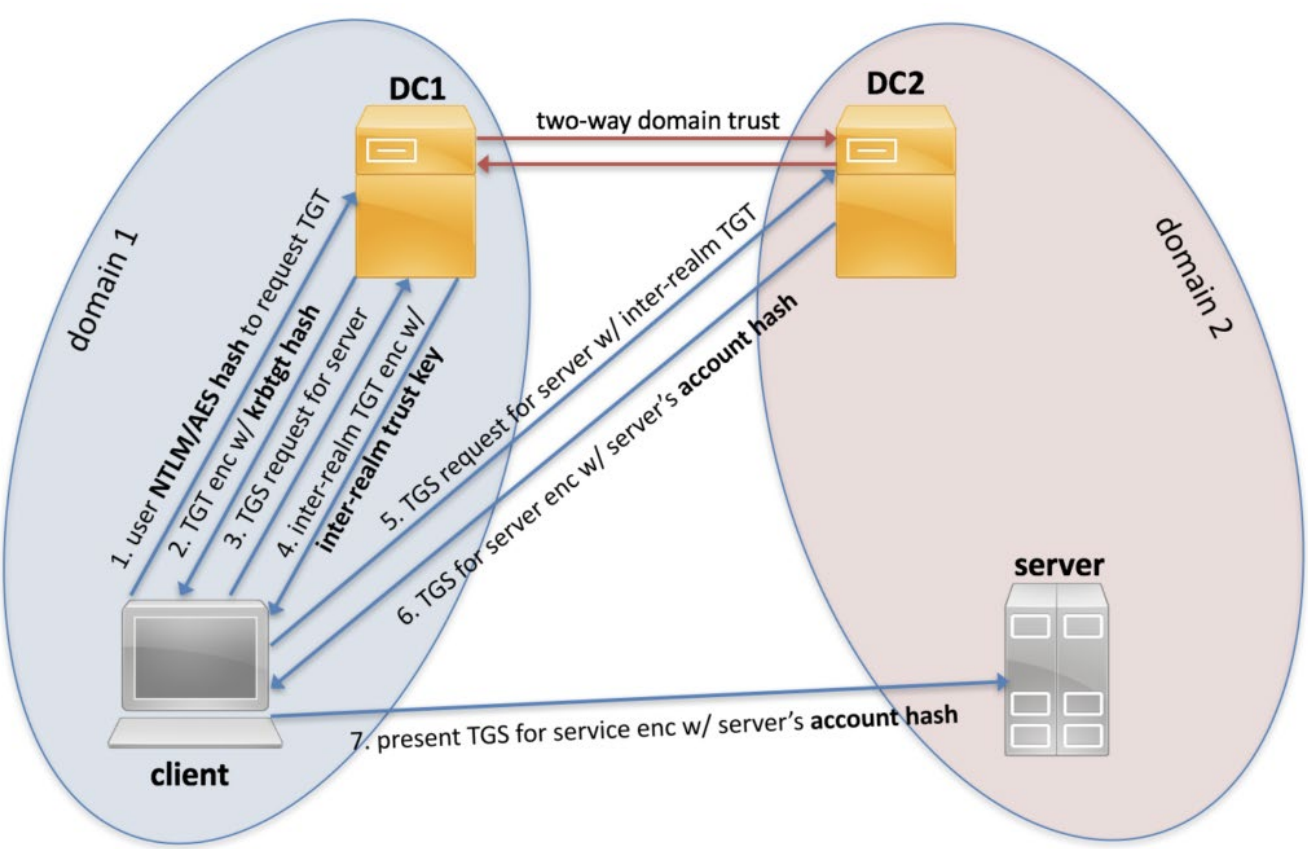
Попутно мы бы могли запустить **Get-NetLocalGroupMember <foreign,server>** против целевой выборки серверов (включая контроллеры домена), чтобы увидеть, заходят ли сюда через границу какие-либо пользователи через локальные группы компьютеров. Мы также могли бы использовать целевой **Get-DomainObjectACL -Domain <foreign.domain>** с различными фильтрами для проверки чужого участия в ACL.

Или мы могли бы просто вытянуть всё через BloodHound, и довериться схеме моделирования взломов между лесами. :)

Заметка на полях: Подделка билетов доверия между областями

Можно подделать билеты доверия между областями, чтобы воспользоваться доверительными отношениями. Так как Шон превосходно осветил это в своей статье «Всё о трастах», я отошлю вас к его документации за более подробной информацией и просто расскажу о последствиях этой техники и о том, как она вписывается в нашу стратегию атак на трасты.

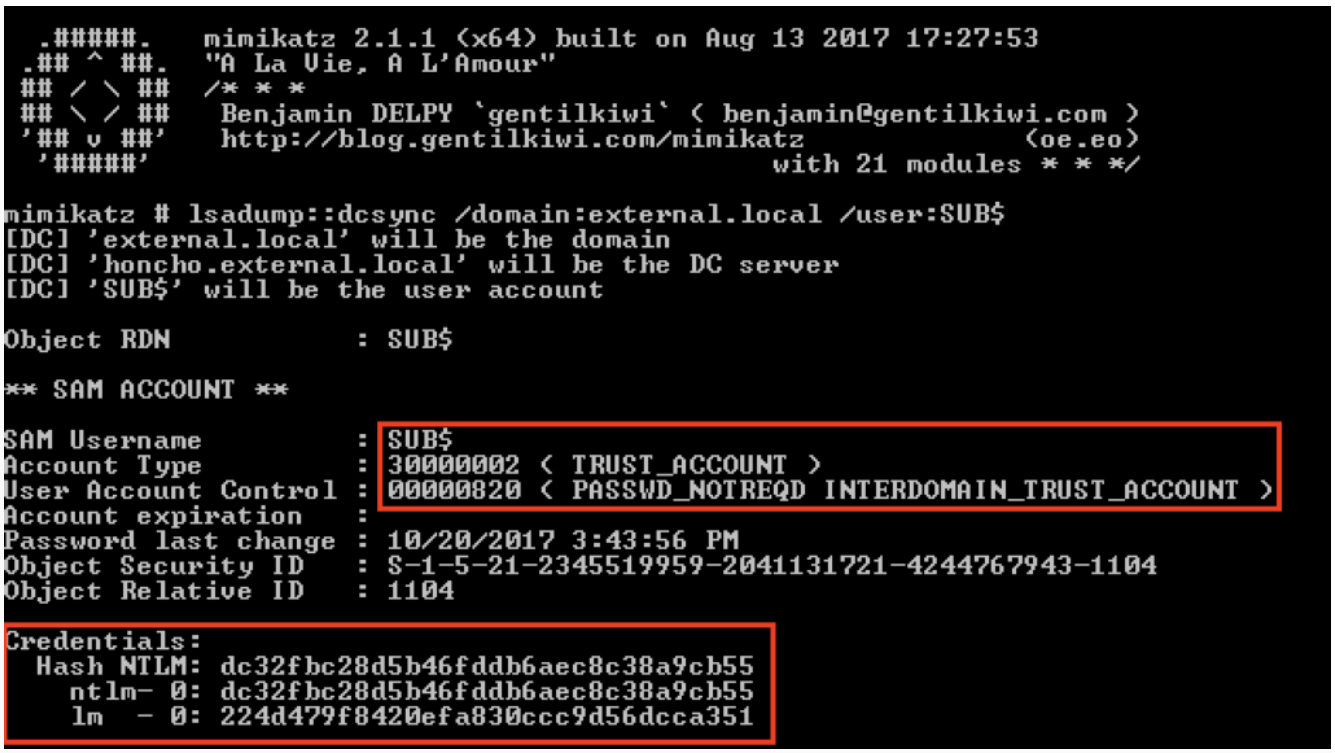
Вспомните схему, как Kerberos работает через трасты:



Итак, когда пользователь представляет этот реферальный запрос на получение межобластных билетов в чужой домен, заново подписанный ключом доверия между областями, TGT пользователя включается в него. И опять, поскольку чужой домен доверяет домену, который выдал реферальный билет, этот чужой домен доверяет TGT пользователя и всей включённой в него информации, чтобы быть точным.

Опять же, повторяю вам по-английски, когда чужой домен расшифровывает реферальный билет с помощью согласованного ключа доверия, он видит TGT пользователя и говорит: *«О'кей, другой домен уже установил подлинность этого пользователя и сказал, что он тот, о ком идёт речь/это группы, в которых пользователь состоит, поэтому я поверю этой информации, потому что я доверяю этому домену»*.

Таким образом, если у нас получится извлечь хэш ключа доверия между областями, то можно подделать реферальный билет (как описывает Шон), который позволит нам притвориться любым пользователем из первого домена при запросе доступа ко второму домену. Этот хэш-запрос можно выполнить через обычный дамп паролей или через DCSync, запросив учётную запись **FOREIGN_DOMAIN_SHORTNAME$**:



Впрочем, если мы сможем добыть ключ доверия между областями, то почти всегда можно извлечь хэш krbtgt ссылающегося домена. Если у нас он есть, мы можем создать билет для домена, ссылающегося на пользователя, притворяясь любым пользователем, каким хотим, на чужом домене. Вот почему у меня не было нужды подделывать ссылки межобластных трастов на местах. Но есть один конкретный пример, когда это становится интересным.

В 2015 году, после того как все начали по-настоящему осознавать значение золотых билетов, Microsoft выпустила сценарии, дающие организациям возможность изменять пароль учётной записи krbtgt. Чтобы это было эффективно для леса с одним доменом, пароль надо было менять дважды. Теперь, из-за осуществления атаки с взломом sidHistory, пароль для учётной записи krbtgt должен быть дважды изменён в КАЖДОМ домене леса.

Скажем, организация делает это и меняет пароли для *каждой* учётной записи с расширенными правами в *каждом* домене леса. Безопасны ли они? Нуу, хотя ключи доверия между областями автоматически изменяются каждые 30 дней в соответствии с разделом 6.1.6.9.6.1 Технической спецификации Active Directory, их не обновляют при изменении учётной записи krbtgt. Так что, если злоумышленник обладает ключами между областями, он всё равно может использовать подход sidHistory, чтобы взломать траст, как это подробно излагает Шон. Опять же, существует миллион и ещё один способ «сделать» Active Directory. ¯_(ツ)_/¯

Так что существует только одно решение, если вы хотите быть полной уверенности (спасибо @gentilkiwi :)

Ещё одна заметка на полях: Kerberoasting через доменные тресты

Мы любим Kerberoasting. Представленный Тимом Мединой в 2014 году, он помог нам «прожарить» уже такую гору целей! И если у нас есть домен, который нам доверяет, мы можем «прожечь» эти трастовые границы одним маленьким «щипком» в некоторых случаях.

При использовании класса .NET System.IdentityModel.Tokens.KerberosRequestorSecurityToken (а затем его метода .GetRequest()), мы указываем имя принципала службы (SPN), чтобы запросить TGS-REP, и затем используем GetRequest() для извлечения байтов для AP-REQ, предназначенного для отправки целевой службе. Этот AP-REQ содержит билет службы, который мы затем извлекаем и применяем для оффлайнового Kerberoasting'а/взлома пароля.

Документация для KerberosRequestorSecurityToken.ServicePrincipalName прекрасно описывает этот формат как «*host/<hostname>**@<domain>** или <hostname>, где hostname – это имя компьютера, на котором размещена целевая веб-служба, а domain – полностью квалифицированное доменное имя области Kerberos, в которой находится компьютер.*» Поэтому, если у вас есть какие-либо проблемы с Kerberoasting между трастами (особенно внешними и лесными), попробуйте использовать формат **SERVICE/host.domain.com@domain.com**, и вас ждёт больший успех. Это возможно с помощью **Get-DomainSPNTicket** от PowerView - функции, на которой построен **Invoke-Kerberoast**.

Эпилог: фильтрация SID

Итак, ранее мы говорили о том, как/почему лес является границей траста в Active Directory, а не доменом. Большую роль в этом играет защита безопасности, которую я уже упоминал несколько раз ранее – она называется фильтрацией SID. Лучшим справочником по фильтрации SID является документация [MS-PAC] «Структура данных сертификата привилегированного атрибута», а именно раздел 4.1.2.2 "Фильтрация SID и преобразование заявок". Я постараюсь как можно лучше объяснить несколько существенных моментов.

Когда TGT пользователя представляется новому домену через ссылку, этот TGT содержит сертификат привилегированного атрибута (PAC), в котором есть, помимо прочего, идентификатор безопасности пользователя (SID), идентификаторы безопасности групп, в которых он состоит, и всё, что есть в ранее обсуждавшемся поле sidHistory (то есть в части **ExtraSids** PAC, описанной в разделе **«Трастпокалипсис»**). Эта информация идентификации безопасности в PAC разбирается и анализируется *доверяющим* доменом, и применяются различные фильтры - в зависимости от типа траста.

SIDы, соответствующие определённым шаблонам, отклоняются доверяющим доменом при различных обстоятельствах - как мера защиты безопасности. Фильтрация SID придумана для того, чтобы не дать злоумышленникам с расширенными учётными данными в *доверенном* домене/лесу обрести контроль над *доверяющим* доменом/лесом. Это также описано в документации Microsoft «Вопросы безопасности для трастов».

Существует набор идентификаторов безопасности (SIDs), для которых установлено значение «AlwaysFilter» - они всегда отфильтровываются доверяющим доменом независимо от типа траста. Основной SID, интересный для нас - «Администраторы предприятия» (S-1-5-21-<Domain>-519). Тот, который позволяет нам выполнять атаку со взломом sidHistory. Он установлен на «ForestSpecific» для фильтрации. Как описывает Microsoft: «*Правило ForestSpecific предназначено для тех идентификаторов безопасности, которые никогда не допускаются в PAC, исходящий из леса или домена, помеченный как QuarantinedWithinForest, если только он не принадлежит этому домену*». Опять же, это объясняет, почему лес является границей траста, а не доменом, поскольку этот расширенный SID (наряду со многими другими) не может быть передан через границу траста, кроме случаев, когда целевой домен находится в том же самом лесу.

QuarantinedWithinForest, да? Случается, что домены *внутри* леса могут быть помечены как «помещённые в карантин», что осуществляет фильтрацию SID версии для домена, даже если он находится в лесу. Однако, как утверждается в документации, «*Единственными идентификаторами безопасности, которые разрешено передавать из такого домена, являются SID «Контроллеры домена предприятия» (S-1-5-9) и те, которые описаны объектом доверенного домена (TDO)*». Таким образом, поскольку SID «Контроллеры домена предприятия» НЕ отфильтровывается для доменов внутри леса, помещённых в карантин, всё равно существует способ «взломать» цепь лесного траста и скомпрометировать корень леса:

Вот то, что я пытался объяснить несколько лет назад, без понимания проблемы должным образом - но теперь она становится более понятной после прочтения кучи документации Microsoft :)

Итог

Трасты – тема непростая. Большинство пентестеров (и многих системных администраторов!) не имеют правильного понимания трастов и риска, связанного с их различными неправильными конфигурациями. К нашему сожалению, некоторые плохие парни действуют, опираясь на обширные знания в этой сфере, и с самого начала работы Active Directory они злоупотребляли трастами для использования доступа в один домен с переходом в другой.

Если архитектура траста вашего домена настроена неправильно, то каково решение? К сожалению, как со многими крупными архитектурными недостатками такого рода, здесь нет простого решения. Реорганизация основного развёртывания Active Directory может быть долгим, дорогостоящим и болезненным процессом, но всё же есть свет в конце тоннеля: административная среда расширенной безопасности (ESAE), обычно называемая «Red Forest». Она представляет собой

защищённую архитектуру Active Directory, которая убирает огромное количество уязвимостей/неправильных конфигураций Active Directory. Хотя Microsoft опубликовала аспекты этой архитектуры, единственный известный мне в настоящее время способ получения ESAE/Red Forest, - это заплатить Microsoft за её внедрение для вас. ¯ \ _ (ツ) _ / ¯

Я хотел бы со временем опубликовать руководство для организаций о том, как внедрять среду «red-forest-esque», которая, хоть и соответствует эталонной архитектуре всего лишь «неофициально», но всё же она лучше, чем нынешние реализации многих предприятий. Если у кого-то есть документация или руководство, как сделать это практически - пожалуйста, свяжитесь со мной: @harmj0y. Я буду счастлив получить такого рода информацию!