Идём по киберследу

Дмитрий Неверов

Анализ защищенности Active Directory с помощью утилиты BloodHound

PRO

Москва
2025

УДК 004.9
ББК 16.8
Н44

Редактор Евгения Якимова

Неверов Д.

Н44 Идём по киберследу : Анализ защищенности Active Directory с помощью утилиты BloodHound / Дмитрий Неверов. — М. : Альпина ПРО, 2025. — 298 с.

ISBN 978-5-206-00398-7

Представьте, что вы можете видеть невидимые связи в вашей инфраструктуре Active Directory, выявлять сложные последовательности атак и устранять их до того, как они приведут к инцидентам. Утилита BloodHound делает это реальностью! В этой книге вы познакомитесь с мощным инструментом, который использует графовую базу данных neo4j и язык запросов Cypher, чтобы дать вам полный контроль над вашей системой безопасности. С помощью этой книги вы сможете освоить интерфейсы BloodHound и расширять его функционал для решения специфических задач вашей организации, научитесь писать эффективные запросы на языке Cypher для выявления скрытой опасности, визуализировать все опасные связи между объектами Active Directory и планировать действия по их устранению.

Не важно, специалист вы по безопасности, аудитор или участник Red Team, эта книга даст вам все необходимые знания для проведения глубокого анализа защищенности Active Directory и выявления потенциальных атак.

УДК 004.9
ББК 16.8

Все права защищены. Никакая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами, включая размещение в интернете и в корпоративных сетях, а также запись в память ЭВМ для частного или публичного использования, без письменного разрешения владельца авторских прав. По вопросу организации доступа к электронной библиотеке издательства обращайтесь по адресу mylib@alpina.ru.

ISBN 978-5-206-00398-7

© Неверов Д., 2024
© Оформление. ООО «Альпина ПРО», 2025

СОДЕРЖАНИЕ

ВСТУПЛЕНИЕ

Утилита BloodHound — популярный инструмент для проведения оценки защищенности Active Directory. BloodHound использует графовую базу данных neo4j и язык запросов Cypher, что позволяет увидеть небезопасные связи между объектами, которые не очевидны при обычном линейном рассмотрении. В книге приводятся интерфейсы BloodHound и базы данных neo4j. Также мы знакомимся с языком запросов Cypher на реальных примерах, а в завершение рассматриваем, как можно расширить функционал BloodHound, чтобы повысить эффективность утилиты.

1

ГЛАВА

ОБЩАЯ ИНФОРМАЦИЯ И НАСТРОЙКА ЛАБОРАТОРИИ

===a){while(c=
unction(a,b){if("undefined"!=typeof b.ge-
=[],(c.qsa=Y.test(n.querySelectorAll))&&
:t id='"+u+"-\r\\' msallowcap
aptur
'+K+"*(?:value|"+J+")"),a.querySelectorAl
d").length||q.pu
innerHTML="<a hr
ement("input");b.setAttribute("type","hid
push("name"+K+"*[*
oled",":disabled"),o.appendChild(a).disa
ed",":disabled"),a.querySelectorAll("*,:x
itMatchesSelector||o.mozMatchesSelector||o
:tedMatch=s.call(a,"*"),s.call(a,"[s!='']
&&new RegExp(r.join("|")),b=Y.test(o.com-
)===a.nodeType?a.documentElement:a,d=b&&b
ains(d):a.compareDocumentPosition&&16&a.
ode)if(b===a)return!0;return!1},B=b?func-
!b.compareDocumentPosition;return d?d:(d=
(b):1,1&d||!c.sortDetached&&b.compareDocu
.ownerDocument===v&&t(v,b)?1:k?I(k,a)-I(k,b
rentNode,f=b.parentNode,g=[a],h=[b];
turn"input"===c&&b.type===a}}function na(
=c||"button"===c)&&b.type===a}}function
==!1?"label"in b?"label"in b.parentNode?b
ed!==!a&&ea(b)===a:b.disabled===a:"la-
rn b=+b,ia(function(c,d){var e,f=a([],c.
unction qa(a){return a&&"undefined"!=-
n(a){var b=a&&(a.ownerDocument||a).docu-
on(a){var b,e,g=a?a.ownerDocument||a:v;re
ent,p=!f(n),v!==n&&(e=n.defaultView)&&e.
ttachEvent&&e.attachEvent("onunload",da))
lassName")}),c.getElementsByTagNam
sByTagName("*").length}),c.getElementsBy-
{return o.appendChild(a).id=u,!n.getEle-
=function(a){var b=a.replace(_,aa);return
){if("undefined"!=typeof b.getElementBy-
ction(a){var b=a.replace(_,aa);return
buteNode("id");return c&&c.value===b}},d
c,d,e,f=b.getElementById(a);if(f){if-
sByName(a),d=0;while(f=e[d++])if(c=f

Любой проект начинается со сбора и анализа информации, и проекты по наступательной безопасности не исключение. Можно сказать, что это один из самых важных этапов проекта: качество собранной информации позволит эффективно выполнить поставленные задачи и уменьшить количество потраченного времени.

В результате сбора информации мы получаем большой объем данных, который необходимо изучить и извлечь важное содержание. В линейных строковых данных не всегда можно эффективно определить связи между двумя объектами. Визуализация данных в виде графов помогает определить связь между двумя объектами и показать причину возникновения этой связи. Также графы помогают определить дальнейшие шаги при выполнении работ.

Графы можно рисовать на бумаге или в приложениях (например, visio), но при использовании этого метода могут быть упущены некоторые связи. Существуют инструменты, которые на основе полученных данных могут показать связи между объектами, даже если эти связи на первый взгляд неочевидны. Среди них *Adelante*[1], *Ping Castle*[2] и *BloodHound*[3]. Именно о BloodHound эта книга.

[1] https://github.com/Seyaji/adelante.

[2] https://github.com/vletoux/pingcastle.

[3] https://github.com/BloodHoundAD/BloodHound.

Что такое BloodHound

BloodHound состоит из трех элементов:

1. *BloodHound* — это одностраничное веб-приложение, написанное на Java Script; при создании приложения используется *Linkurious*. Для компиляции используется Electron.
2. *Neo4j* — база данных для хранения информации, в которой используется язык запросов Cypher.
3. *SharpHound* — сборщик информации из Active Directory.

BloodHound использует теорию графов, чтобы показать скрытые и часто непреднамеренные связи в среде Active Directory или Azure.

Область применения

Наступательная безопасность: специалисты по информационной безопасности могут использовать BloodHound для обнаружения очень сложных последовательностей атак, которые обычным способом невозможно быстро обнаружить.

Кроме наступательной безопасности этот инструмент может использоваться и в других областях для обеспечения безопасности, например:

- Защитники могут использовать BloodHound для выявления и устранения тех же последовательностей атак.
- Специалисты по реагированию на инциденты могут использовать BloodHound для проведения расследований и выявления причин инцидента.
- Аудиторы могут проводить проверки на соответствие стандартам безопасности.
- Утилита BloodHound будет полезна во время стратегических и тактических игр, когда любой сценарий можно визуализировать и пошагово разобрать.

Настройка лаборатории

Материал в книге подготовлен с использованием среды Windows. Для сбора дополнительной информации используются скрипты, написанные на Powershell, который установлен по умолчанию на Windows и имеет достаточный набор функций для работы с доменом, файловой системой и т. д.

Для изучения материала потребуется тестовый стенд с Active Directory, а также машина, на которой будут анализироваться данные и добавляться функционал к самому BloodHound. В книге домен называется `DOMAIN.LOCAL`, но это не имеет большого значения, самое главное — менять имя домена на свое при выполнении запросов.

Минимальные требования к стенду — это контроллер домена и компьютер для аналитики и разработки. Для удобства сбора информации и анализа данных машину для BloodHound можно ввести в домен. Наименование машин будет следующим:

- `DC` — контроллер домена, Windows Server 2019;
- `COMP` — рабочая станция для BloodHound, Windows 10/11.

Если мощности позволят, необходимо создать еще одну виртуальную машину и добавить ее в домен. Если мощностей нет — тогда просто создать еще один объект, компьютер:

- `SERVER` — просто сервер, Windows Server 2019.

Установка Active Directory

В первую очередь для установки Active Directory на сервере, который будет являться контроллером домена, необходимо поменять имя на `DC` и установить статический адрес.

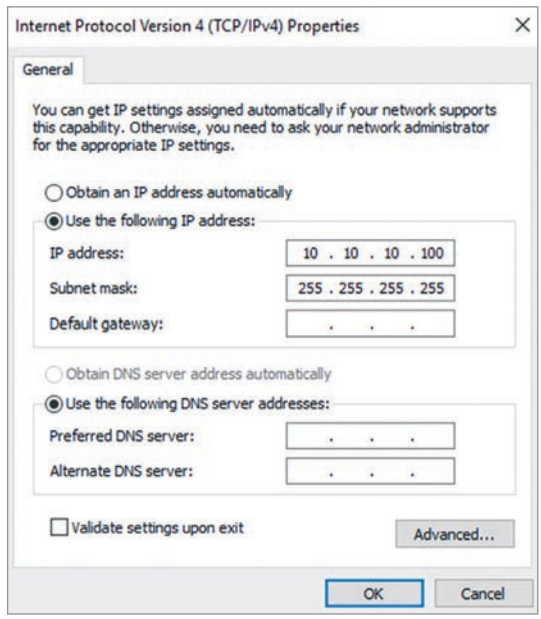


Рис. 1.1. Установка статического адреса для контроллера домена

Запускаем *Server Manager* и выбираем *Add roles and features*. Следуем за мастером добавления новой роли. Нажимаем кнопку *Next*. Предложенные по умолчанию настройки нас будут устраивать, поэтому нажимаем кнопку *Next* до тех пор, пока не появится окно *Select server roles*.

Выбираем следующие роли:

- *Active Directory Domain Service;*
- *DNS Server.*

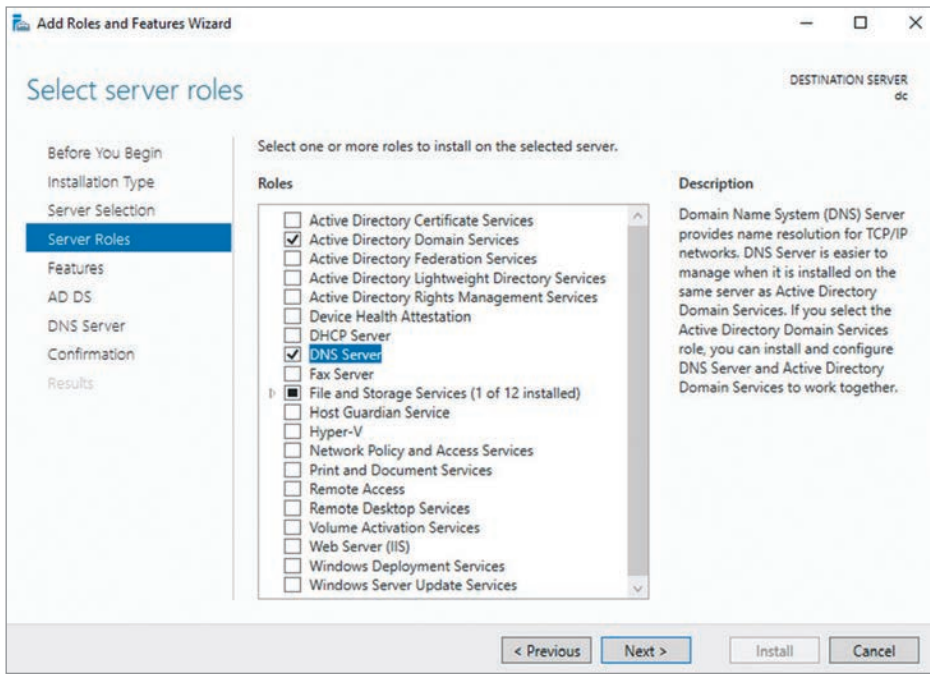


Рис. 1.2. Выбор ролей

Далее нажимаем кнопку *Next* до самого конца, пока кнопка *Install* не станет активной, и нажимаем на нее (рис. 1.3)

После установки нажимаем на кнопку *Close*. В верхнем правом углу появился желтый восклицательный знак, который указывает нам, что роли требуют завершения настройки (рис. 1.4).

Нажимаем на ссылку *Promote this server to a domain controller*. Появляется окно с выбором, куда добавить контроллер домена. Так как у нас ничего нет, выбираем *Add a new forest* и вводим имя домена `domain.local` (рис. 1.5).

В следующем окне оставляем все по умолчанию и вводим пароль для восстановления (рис. 1.6).

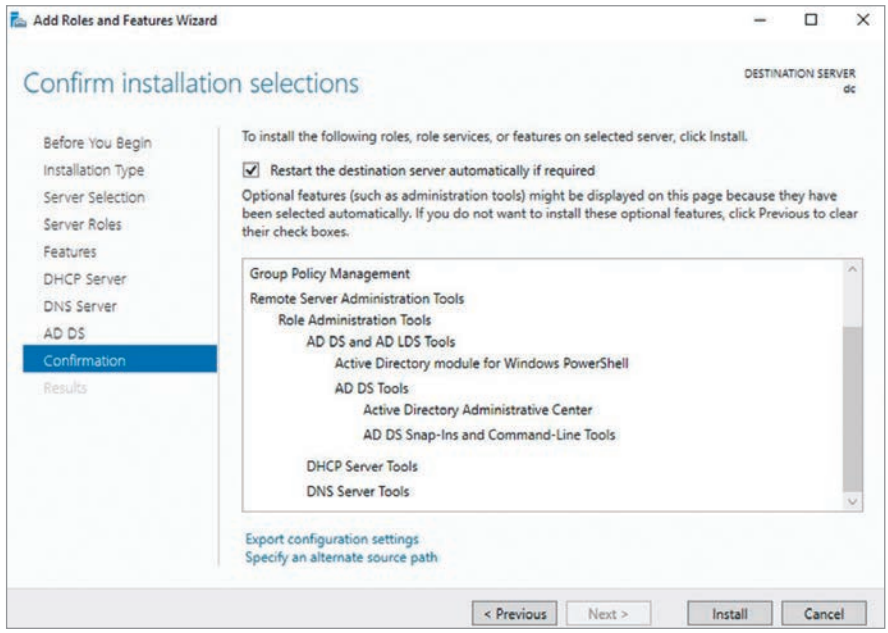


Рис. 1.3. Подтверждение установки Active Directory

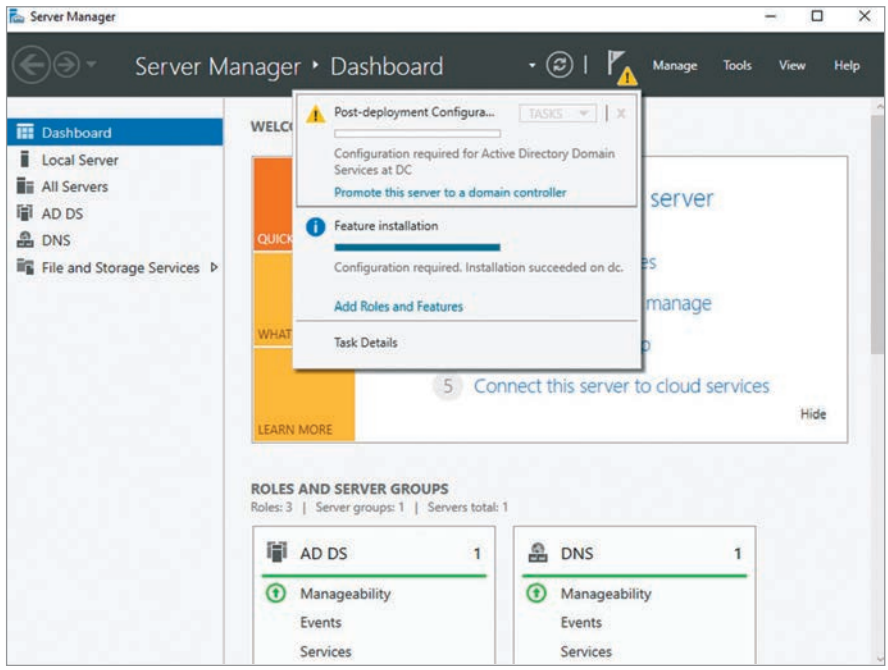


Рис. 1.4. Завершение установки Active Directory

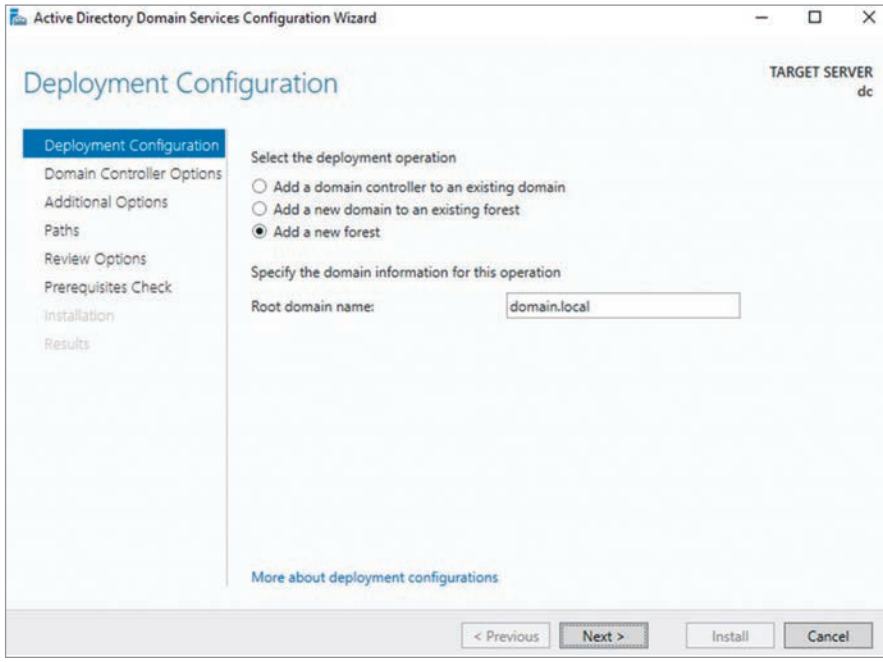


Рис. 1.5. Установка имени домена

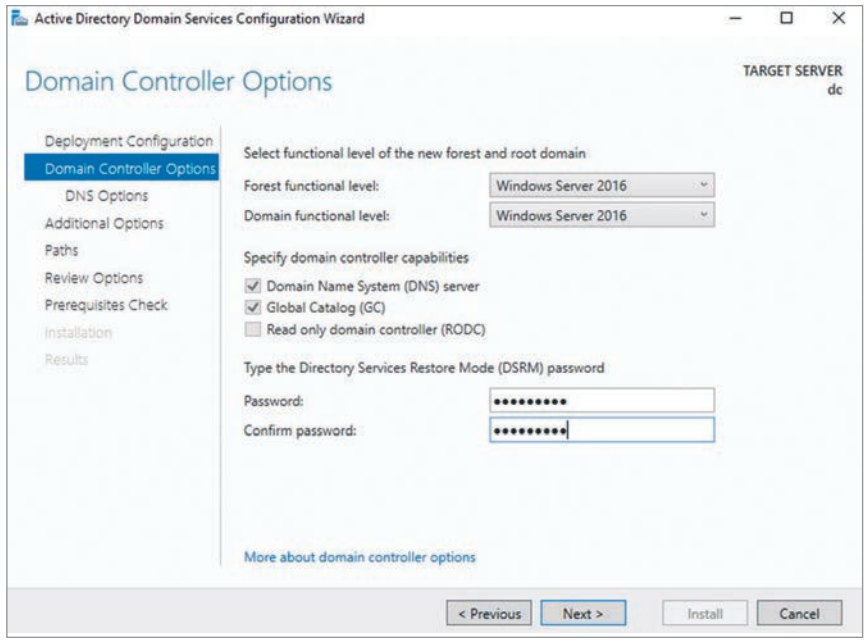


Рис. 1.6. Выбор уровня домена

Все остальные настройки оставляем без изменений и нажимаем кнопку *Next* до того момента, как кнопка *Install* станет активной. Нажимаем на нее, дожидаемся установки и перезагрузки сервера.

Теперь необходимо ввести машины `COMP` и `SERVER` в домен. Но перед этим их нужно переименовать и установить статические IP-адреса.

Совет

Рекомендуется на всех хостах отключить межсетевой экран и антивирус, чтобы они нам не мешали во время экспериментов.

Создание объектов домена

После создания домена необходимо создать несколько учетных записей:

- `admin` — пароль Qwerty123, установить бесконечный срок действия пароля, после создания добавить пользователя в группу доменных администраторов;
- `user` — пароль Qwerty123, установить бесконечный срок действия пароля;
- `victim` — пароль Qwerty123, установить бесконечный срок действия пароля.

Создать пользователей можно с помощью ADUC или Active Directory Module. В этом случае команды будут следующими:

```
# Создать пользователя admin
New-ADUser -Name "admin" -SamAccountName "admin"
-UserPrincipalName "admin@domain.local" -DisplayName
"admin" -GivenName "admin" -AccountPassword (ConvertTo-
SecureString "Qwerty123" -AsPlainText -force) -Enabled
$true -PasswordNeverExpires $true

# Создать пользователя user
New-ADUser -Name "user" -SamAccountName "user"
-UserPrincipalName "user@domain.local" -DisplayName
"user" -GivenName "user" -AccountPassword (ConvertTo-
SecureString "Qwerty123" -AsPlainText -force) -Enabled
$true -PasswordNeverExpires $true

# Создать пользователя victim
New-ADUser -Name "victim" -SamAccountName "victim"
-UserPrincipalName "victim@domain.local" -DisplayName
"victim" -GivenName "victim" -AccountPassword
```

```
(ConvertTo-SecureString "Qwerty123" -AsPlainText -force)
-Enabled $true -PasswordNeverExpires $true

# Добавить пользователя admin в группу доменных администраторов
Add-ADGroupMember -Identity "Domain Admins" -Members
admin
```

Совет

Для наполнения домена можно воспользоваться скриптом BadBlood[1], но не стоит для первых экспериментов создавать большое количество объектов. Работать с хостом COMP будем от имени учетной записи admin, она входит в группу локальных администраторов как член группы доменных администраторов. В некоторых ситуациях можно использовать другие учетные записи для тестирования различных запросов.

Выполним еще несколько действий, которые позволят сделать нашу лабораторию более интересной с точки зрения запросов:

- добавим пользователя user в группу локальных администраторов на хосте COMP;
- авторизуемся на хосте SERVER от имени пользователя victim.

Совет

Добавьте пользователя user в группу локальных администраторов на хосте COMP для будущих запросов.

Установка neo4j

Перед использованием BloodHound необходимо подготовить рабочую станцию (в нашем случае это будет компьютер с именем COMP): установить OpenJDK и neo4j.

Установка OpenJDK

База данных neo4j написана на языке Java, и для ее работы требуется OpenJDK. Для версии neo4j 4.4.11, которая будет использоваться

[1] https://github.com/davidprowe/BadBlood.

на протяжении всей книги, необходимо установить OpenJDK 11. Существует два варианта установки: вручную, где потребуется прописывать все пути самостоятельно, и с помощью *winget*.

Внимание

Если использовать другие версии OpenJDK, при запуске neo4j возникнет ошибка с сообщением, что данная версия не поддерживается, и рекомендациями по поддерживаемым версиям.

Установка OpenJDK вручную

Для начала нужно скачать скомпилированный дистрибутив OpenJDK 11 с официального сайта[1]. Распакуем архив в директорию `C:\Program Files\openjdk\`. Теперь необходимо прописать пути в переменных окружения. В командной строке с правами локального администратора нужно выполнить `sysdm.cpl`, перейти во вкладку «*Дополнительно*» и нажать на переменные среды.

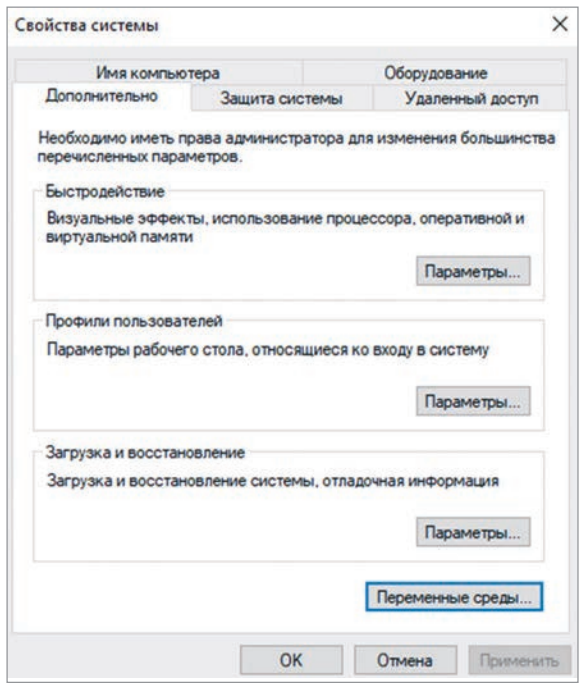


Рис. 1.7. Свойства системы

[1] https://jdk.java.net/java-se-ri/11-MR2.

Теперь нужно создать новую системную переменную с именем `JAVA_HOME` и указать путь до распакованного архива (в нашем случае это `C:\Program Files\openjdk\jdk-11.0.0.1`).

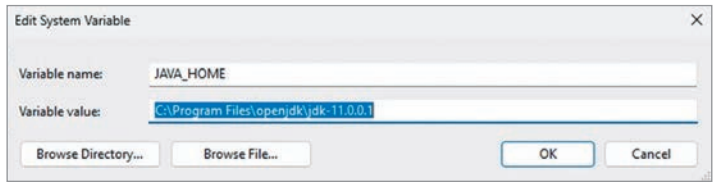


Рис. 1.8. Переменная окружения JAVA_HOME

Также необходимо добавить созданную переменную окружения в PATH.

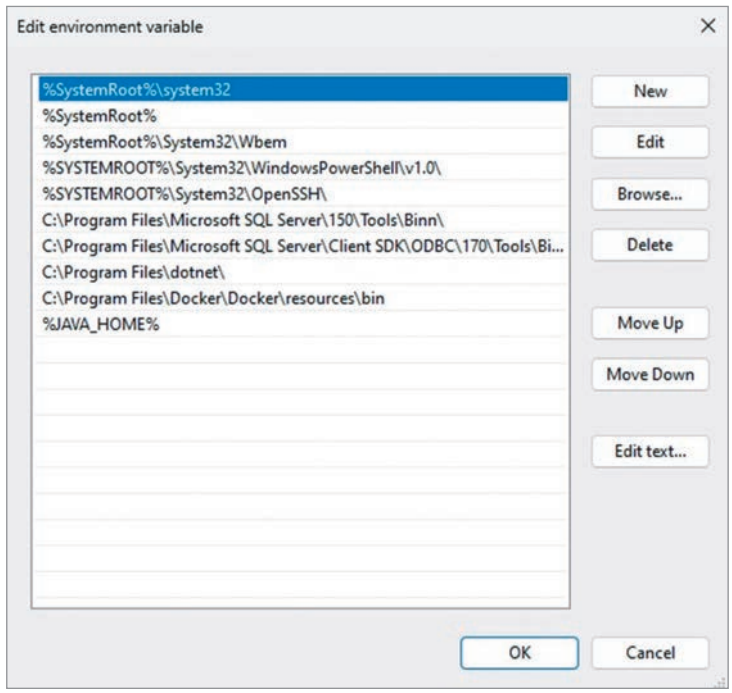


Рис. 1.9. Добавление в PATH

Установка OpenJDK с помощью winget

Для Windows 10 необходимо установить *App Installer* из магазина, а в Windows 11 *winget* установлен по умолчанию. Запустите консоль с правами администратора и выполните команду:

```
winget install ojdkbuild.openjdk.11.jdk
```

После установки все пути будут добавлены автоматически.

Установка neo4j

Скачать neo4j можно с официального сайта разработчика[1], для изучения будем использовать бесплатную версию Community Edition. На момент подготовки книги к печати использовалась версия 4.4.11.

Внимание

Бесплатная версия neo4j позволяет создавать только одну базу. Для работы над проектом этого достаточно.

Распаковываем архив с neo4j в удобную директорию, я обычно использую `C:\Tools\Neo4j\` (для удобства будем называть эту директорию `$NEO4J_HOME`), затем запускаем командную строку или powershell с правами администратора (потребуется для установки службы) и выполняем следующую команду:

```
c:\Tools\Neo4j\bin\neo4j.bat console
```

Если никаких ошибок не возникнет, то можно увидеть информацию об успешном запуске, как показано ниже.

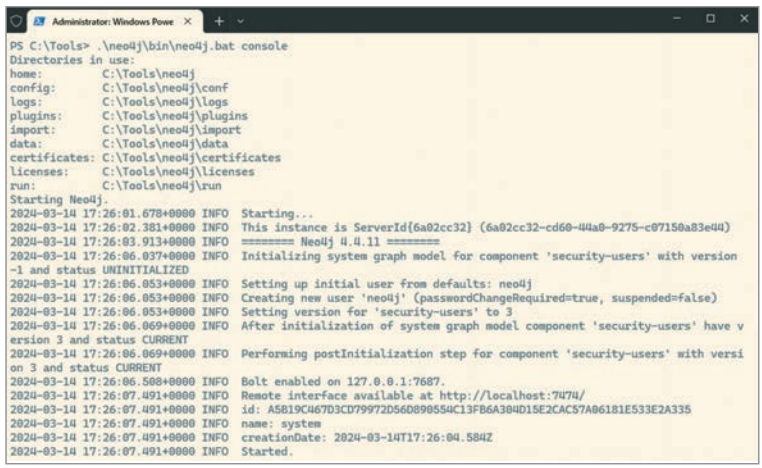


Рис. 1.10. Первый запуск neo4j

[1] https://neo4j.com/.

Остановить neo4j можно сочетанием клавиш *CTRL + z*.

Лучше создать службу, которая будет автоматически запускать neo4j после перезагрузки хоста. Для этого необходимо выполнить следующие команды с правами локального администратора:

```
C:\Tools\Neo4j\bin\neo4j.bat install-service
C:\Tools\Neo4j\bin\neo4j.bat start
```

Первая команда установит службу, а вторая ее запустит.

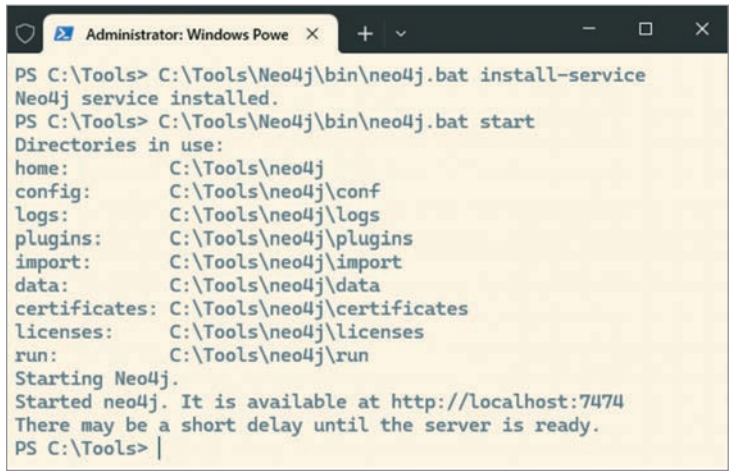


Рис. 1.11. Создание и запуск службы

Проверить статус службы можно с помощью команды:

```
c:\Tools\Neo4j\bin\neo4j.bat status
```

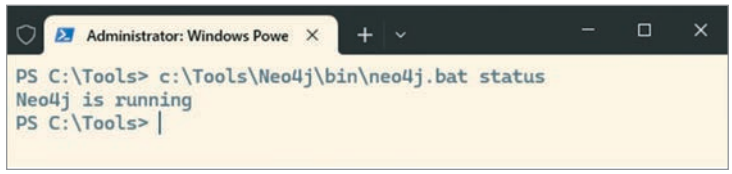


Рис. 1.12. Проверка статуса

Смена пароля

После успешного запуска neo4j требует сменить пароль, установленный по умолчанию. Для этого запускаем браузер, переходим по адресу `http://localhost:7474` и вводим логин `neo4j` и пароль `neo4j`.

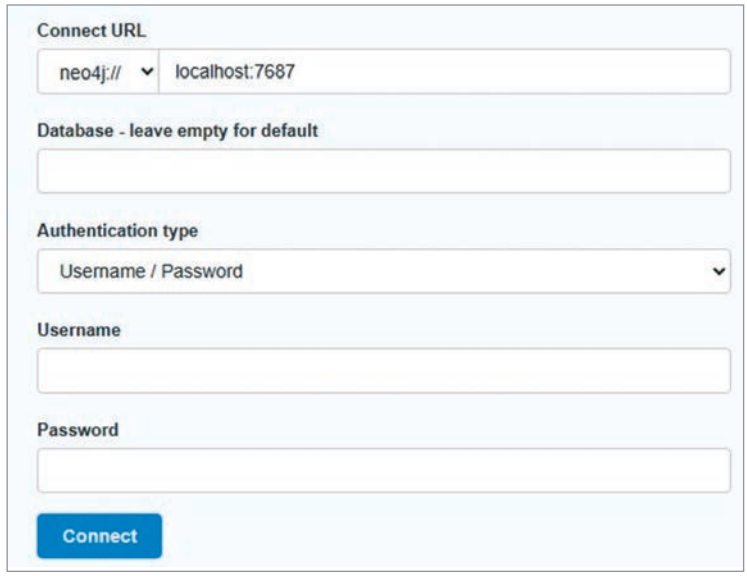


Рис. 1.13. Первый запуск браузера neo4j

Выполнив первую аутентификацию, neo4j попросит сменить пароль для пользователя `neo4j`.

Рис. 1.14. Форма смены пароля

На данном этапе больше никаких действий не потребуется, и мы переходим к заключительной части настройки лаборатории.

Установка BloodHound

Для установки BloodHound не требуется сложных действий. Скачаем необходимую версию (на момент подготовки книги к печати 4.3.1) с официального GitHub[1].

Внимание

|| Более новая версия BloodHound может потребовать новую версию neo4j.

Разархивируем загруженный архив, перейдем в директорию с полученными из архива файлами и запустим `bloodhound.exe`. После запуска приложения появится приглашение для ввода пароля.

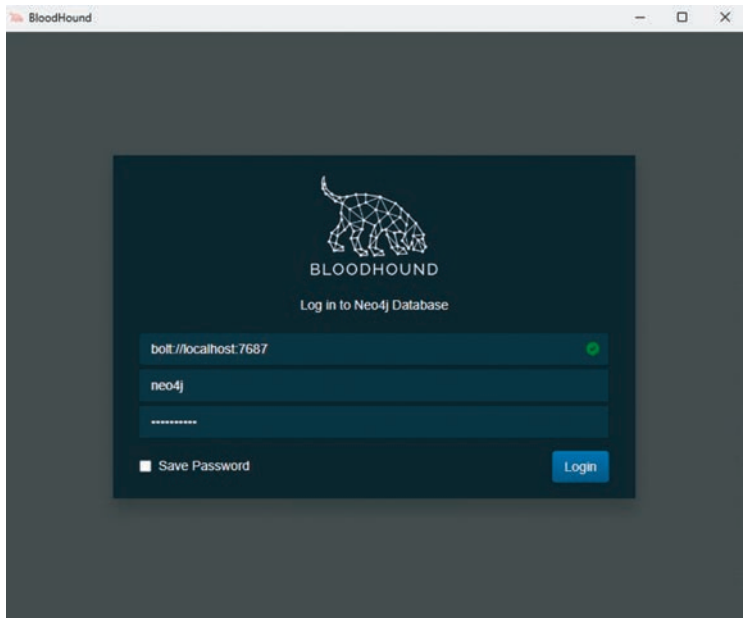


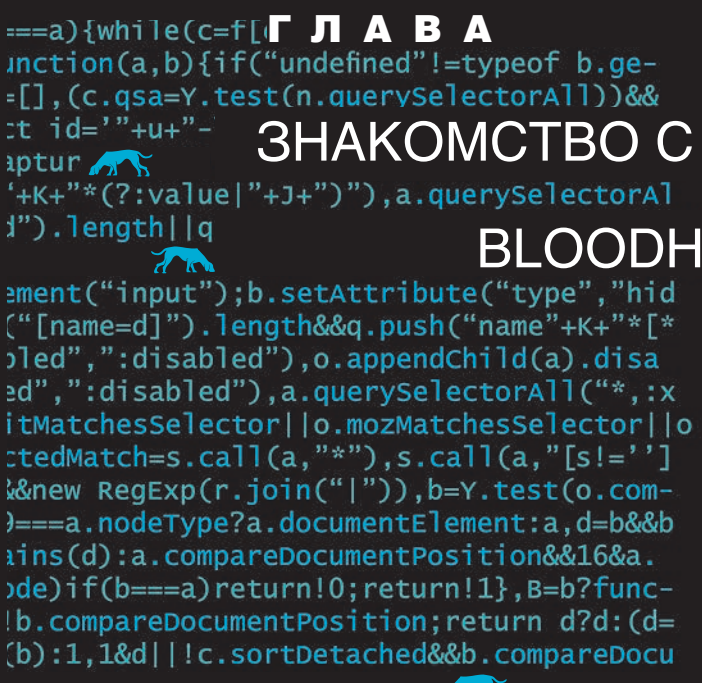
Рис. 1.15. Форма аутентификации BloodHound

Можно установить флаг *Save Password*, чтобы не вводить пароль каждый раз.

[1] https://github.com/BloodHoundAD/BloodHound/releases.

ГЛАВА 2

ЗНАКОМСТВО С SHARPHOUND, BLOODHOUND И NEO4J



Как говорилось ранее, утилита BloodHound состоит из трех частей: непосредственно сама BloodHound, сборщик данных SharpHound и база данных neo4j. В этой части книги мы рассмотрим интерфейсы этих приложений.

SharpHound

SharpHound — это консольное приложение, написанное на C#. SharpHound собирает информацию об объектах домена через запросы LDAP, а также информацию с хостов, такую как членство в локальных группах и сессиях.

Скачать SharpHound можно на официальном GitHub[1]. Разные версии SharpHound генерируют разные форматы данных в JSON, которые BloodHound не всегда принимает. Для BloodHound версии 4.3.1 подойдет SharpHound 1.1.0 или 1.1.1.

Информация

В исходных кодах BloodHound тоже есть SharpHound, он находится в директории `Collectors`.

Внимание

Стоит упомянуть, что антивирусные решения считают SharpHound вредоносной утилитой.

SharpHound имеет большое количество настроек, которые помогают более гибко собирать информацию с домена и рабочих станций. Не будем останавливаться на всех параметрах запуска утилиты — у нее есть понятная справка, которую можно получить, выполнив команду:

```
SharpHound.exe -h
```

В интернете можно найти различные подсказки по методам запуска и некоторую дополнительную информацию по запросам Cypher. Одна из таких подсказок представлена на рисунке 2.1[2].

Для последующего изучения нам потребуются данные из домена. Поэтому в нашей лаборатории запустим SharpHound на хосте `COMP` от имени доменной учетной записи `admin`, которая входит в группу доменных администраторов. Это позволит собрать всю полезную информацию из домена и хостов (рис. 2.2).

```
SharpHound -c All
```

[1] https://github.com/BloodHoundAD/SharpHound/releases.

[2] https://github.com/SadProcessor/HandsOnBloodHound/blob/master/BH21/BH4_SharpHound_Cheat.pdf.

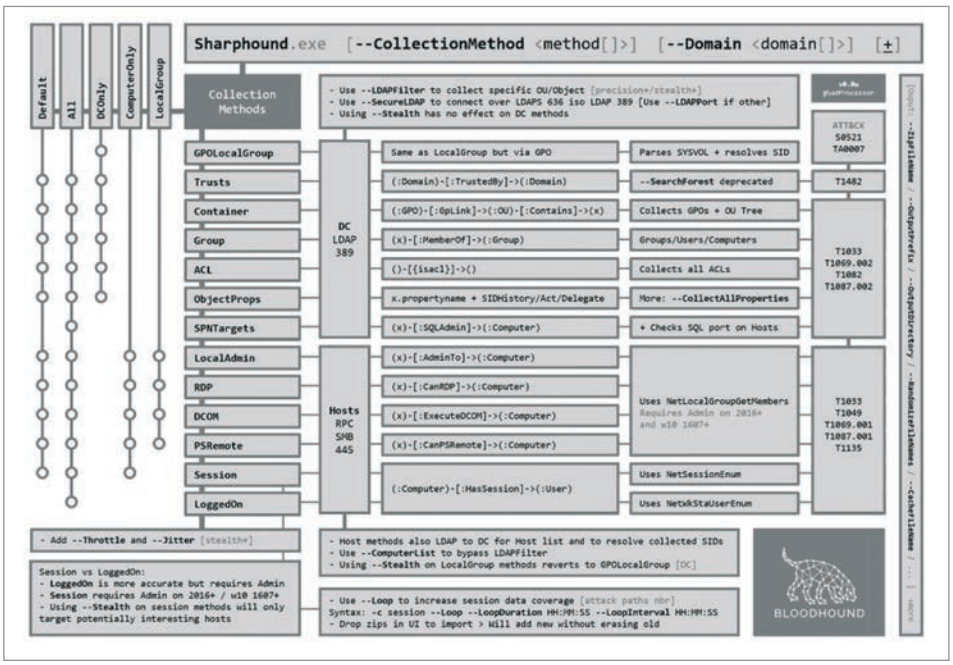


Рис. 2.1. Подсказка по SharpHound

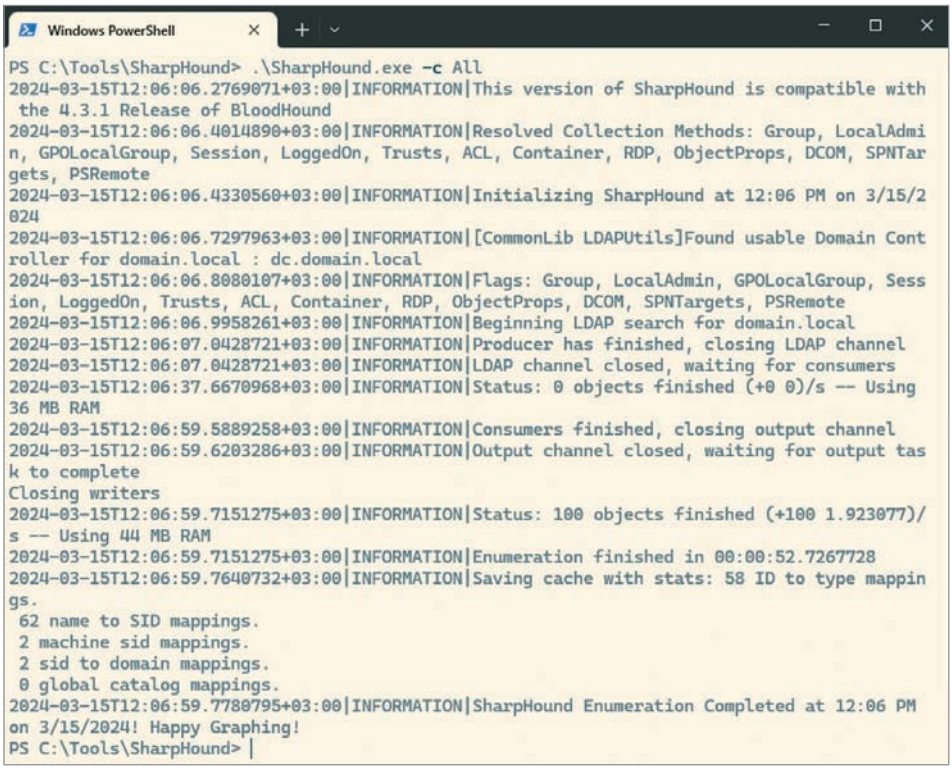


Рис. 2.2. Результат сбора информации

Внимание

Обновленные операционные системы не позволяют получить информацию о сессиях пользователей (*HasSession*) и членстве в локальных группах (*CanRDP* или *AdminTo*) без прав локального администратора.

К собранной информации вернемся позже, а сейчас рассмотрим интерфейс BloodHound.

Интерфейс BloodHound

После прохождения аутентификации открывается основное окно. Пока данных нет, оно пустое; после загрузки данных BloodHound выполняет запрос, который показывает, какие пользователи входят в группу доменных администраторов.

Интерфейс BloodHound интуитивно понятен. Весь его функционал представлен в одном окне.

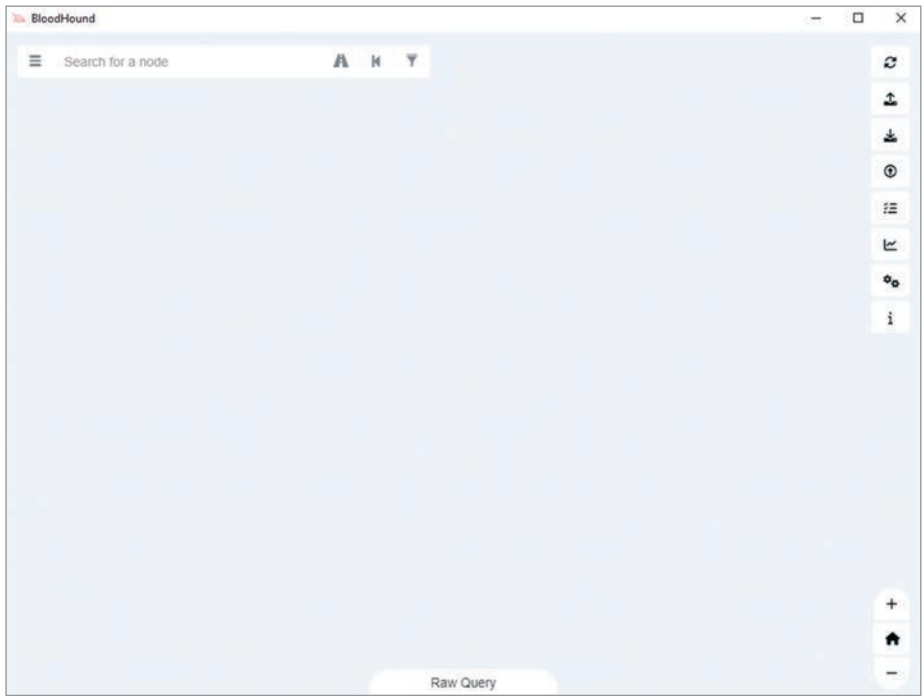


Рис. 2.3. Основное окно BloodHound

В левом верхнем углу располагаются форма поиска и информационные вкладки, в правом верхнем углу — меню для настройки интерфейса, загрузки и выгрузки данных. В правом нижнем углу — работа с масштабом, внизу по центру окна располагается форма *Raw Query* для Cypher-запросов.

Основное поле

В основном поле отображается или узел, или граф на основании запроса Cypher. Узлы можно перемещать по полю. Само поле, узлы и связи имеют контекстное меню, которое можно вызвать правой клавишей мыши.

Форма поиска

Рис. 2.4. Форма поиска

Форма поиска состоит из следующих элементов:

- Дополнительная информация (More Info)
- Поле поиска узлов
- Поиск путей (Pathfinding)
- Возврат (Back)
- Фильтрация типов связей (Filter Edge Types)

Дополнительная информация (More Info)

Этот объект является основным полем для получения информации о свойствах и некоторых связях узлов. При нажатии на кнопку *More Info* выпадает поле, состоящее из трех элементов.

Рис. 2.5. Вкладка «Дополнительная информация»

Database Info

Вкладка содержит статистические данные по количеству объектов в базе данных, а также некоторые инструменты для работы с базой данных (рис. 2.6).

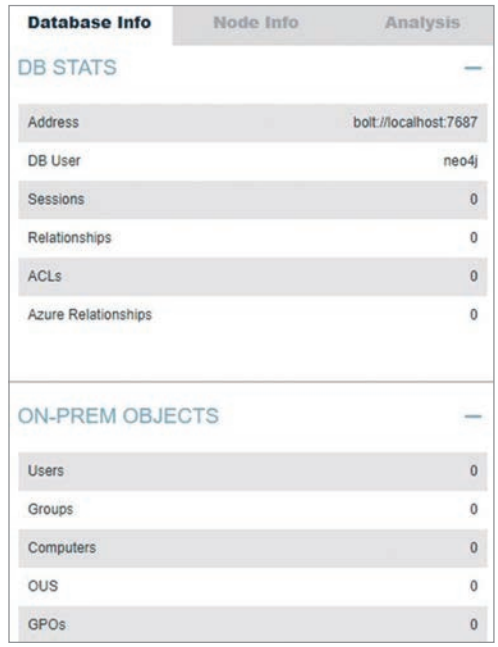


Рис. 2.6. Статистика по узлам и связям

Ниже статистики находятся элементы для управления данными.

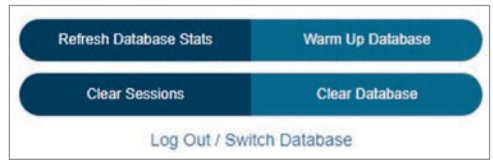


Рис. 2.7. Управление данными

Кратко рассмотрим эти элементы и их функционал:

- *Refresh Database Stats* — после загрузки данных или добавления информации через запросы Cypher статистика может быть неверной, эта кнопка обновляет статистические данные.
- *Warm Up Database* — по описанию от разработчиков, при нажатии этой кнопки данные из базы переносятся в оперативную память, что позволяет увеличить скорость работы с ними.
- *Clear Sessions* — при нажатии этой кнопки удаляются все связи *HasSession*. Эта функция бывает полезной перед загрузкой новых данных о сессиях пользователей.

- *Clear Database* — при нажатии этой кнопки удаляются все узлы и связи между ними.

Замечание

Интересно, что при очистке базы данных в браузере neo4j остаются ссылки на свойства объектов и названия связей.

Информация об узле (Node Info)

В этой вкладке отображаются свойства узла. Кроме того, в ней выполняются некоторые Cypher-запросы, которые предоставляют статистические данные, например, для пользователей и компьютеров выдается информация о сессиях или коротких путях до узлов высокой ценности.

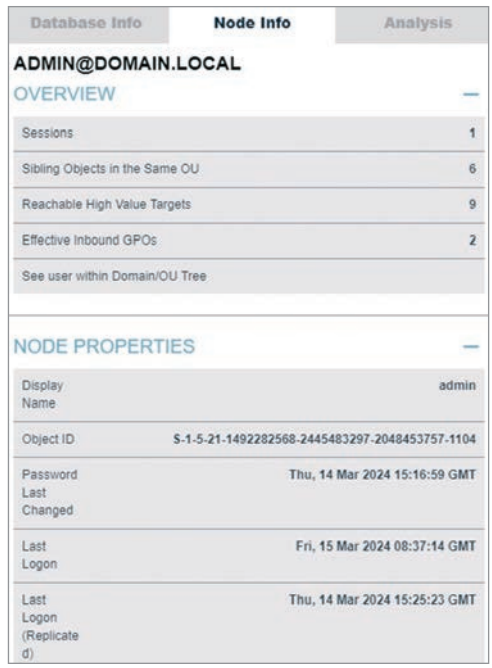


Рис. 2.8. Информация об узле

Разные типы узлов содержат разную информацию. Так, например, групповые политики содержат информацию, к каким пользователям или компьютерам они применяются, а у пользователей и компьютеров отображается информация о правах на другие объекты или правах, связанных с боковым перемещением (рис. 2.9).

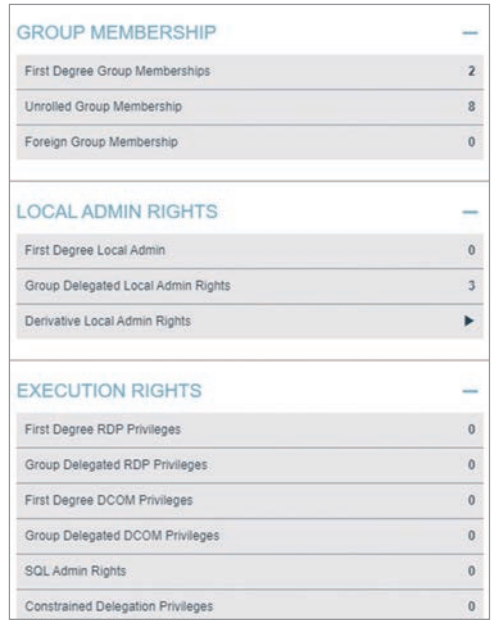


Рис. 2.9. Информация о группах и правах

Очень полезна информация о входящих и исходящих правах (ACL) на другие объекты, которые могут быть использованы во время работ.

Рис. 2.10. Входящие и исходящие ACL

Анализ (Analysis)

Вкладка *Анализ* (*Analysis*) содержит встроенные в BloodHound полезные запросы, с которых можно начать исследовать инфраструктуру Active Directory (рис. 2.11).

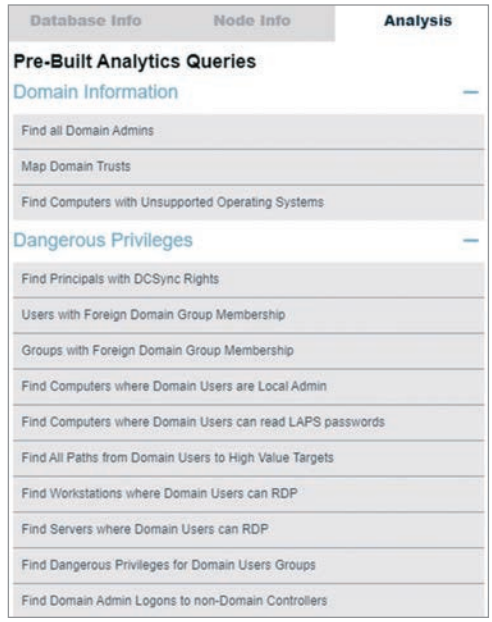


Рис. 2.11. Список встроенных запросов

Информация

Встроенные запросы находятся в файле `PrebuildQueries.json` в директории `src\components\SearchContainer\Tabs.`

Ниже находится раздел для добавления собственных Cypher-запросов, который содержит форму для их создания, а также список созданных запросов, разделенный по категориям.

Рис. 2.12. Раздел создания собственных запросов

При нажатии на кнопку в виде карандаша появляется форма для добавления запросов (рис. 2.13).

При переходе к полю выбора категории запроса можно создать собственную категорию или выбрать из существующих.

Информация

Файл `customqueries.json` с собственными запросами находится в домашней директории пользователя, запустившего BloodHound, с путем `\AppData\Roaming\bloodhound\`.