

## Inhalt

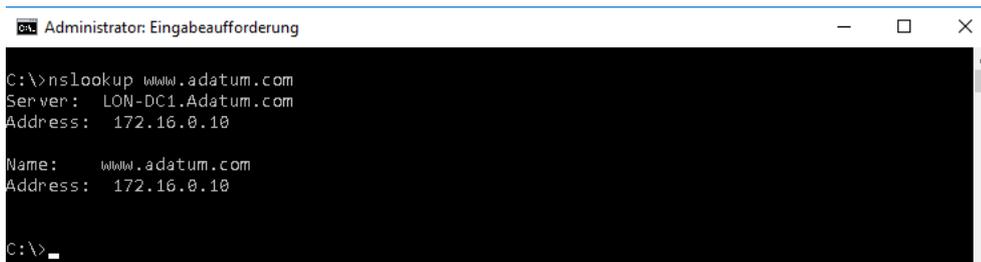
Der Angriff – eine DNS Amplification Attacke .....	1
DNS als gutgläubiger Netzwerkdienst .....	1
Schema einer DNS Amplification Attacke .....	2
Beispielszenario eines Angriffes .....	4
Ein Schutz - Response Rate Limiting? .....	6
Was ist RRL und wie funktioniert es? .....	6
Beispielszenario eines Angriffes mit aktiviertem RRL .....	6
Katz und Maus .....	8

## Der Angriff – eine DNS Amplification Attacke

### DNS als gutgläubiger Netzwerkdienst

Die Namensauflösung im Netzwerk ist zentraler Bestandteil aller Infrastrukturen (was geht ohne?). Und DNS ist als Service aktuell die Komponente, mit der die Namen in IP-Adressen (und umgekehrt) aufgelöst werden.

Dennoch ist DNS schon einige Tage alt und verwendet einige Mechanismen, die durch Angreifer ausgenutzt werden können. Ein DNS prüft nicht, welches System ihm eine Anfrage stellt. Er sucht nach einer Antwort und sendet diese an das System zurück:



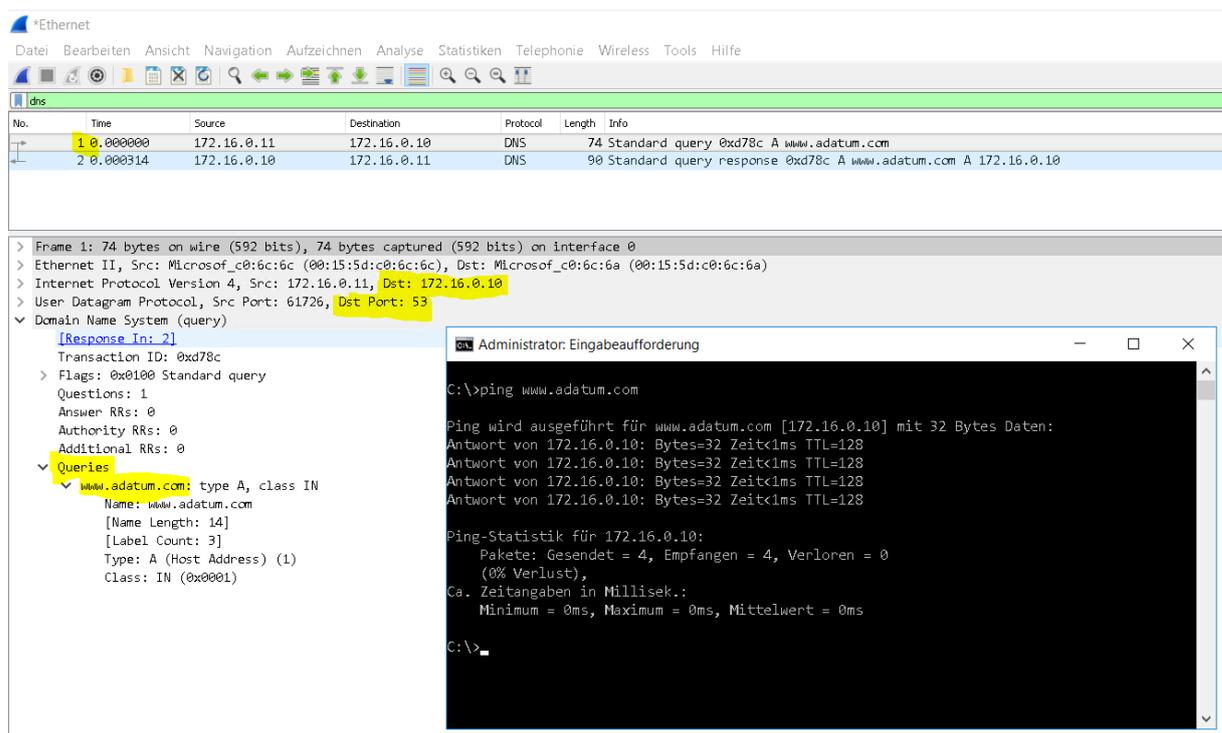
```

Administrator: Eingabeaufforderung
C:\>nslookup www.adatum.com
Server: LON-DC1.Adatum.com
Address: 172.16.0.10

Name: www.adatum.com
Address: 172.16.0.10

C:\>
    
```

Über das Netzwerk werden dafür Pakete zwischen dem Client und dem DNS-Server ausgetauscht. Der DNS-Client sendet an seinen DNS-Server via UDP an dessen Port 53 eine Anfrage (hier ausgelöst durch einen Ping auf www.adatum.com):



The screenshot shows a network capture on the 'Ethernet' interface. The packet list pane shows two DNS packets:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.16.0.11	172.16.0.10	DNS	74	Standard query 0xd78c A www.adatum.com
2	0.000314	172.16.0.10	172.16.0.11	DNS	90	Standard query response 0xd78c A www.adatum.com A 172.16.0.10

The packet details pane for the second packet (Response In: 2) shows:

- Transaction ID: 0xd78c
- Flags: 0x0100 Standard query
- Questions: 1
- Answer RRs: 0
- Authority RRs: 0
- Additional RRs: 0
- Queries:
  - www.adatum.com: type A, class IN
    - Name: www.adatum.com
    - [Name Length: 14]
    - [Label Count: 3]
    - Type: A (Host Address) (1)
    - Class: IN (0x0001)

An inset screenshot shows a Windows command prompt with the output of a ping command:

```

Administrator: Eingabeaufforderung
C:\>ping www.adatum.com

Ping wird ausgeführt für www.adatum.com [172.16.0.10] mit 32 Bytes Daten:
Antwort von 172.16.0.10: Bytes=32 Zeit<1ms TTL=128

Ping-Statistik für 172.16.0.10:
    Pakete: Gesendet = 4, Empfangen = 4, Verloren = 0
            (0% Verlust),
    Ca. Zeitangaben in Millisek.:
        Minimum = 0ms, Maximum = 0ms, Mittelwert = 0ms

C:\>
    
```

Der DNS-Server antwortet auf die Anfrage:

The screenshot shows a network capture in Wireshark. The top pane displays a list of packets:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.16.0.11	172.16.0.10	DNS	74	Standard query 0xd78c A www.adatum.com
2	0.000314	172.16.0.10	172.16.0.11	DNS	90	Standard query response 0xd78c A www.adatum.com A 172.16.0.10

The bottom pane shows the details of the selected packet (Frame 2):

- Ethernet II, Src: Microsof\_c0:6c:6a (00:15:5d:c0:6c:6a), Dst: Microsof\_c0:6c:6c (00:15:5d:c0:6c:6c)
- Internet Protocol Version 4, Src: 172.16.0.10, Dst: 172.16.0.11
- User Datagram Protocol, Src Port: 53, Dst Port: 61726
- Domain Name System (response)
  - Request In: 1
    - [Time: 0.000314000 seconds]
    - Transaction ID: 0xd78c
    - Flags: 0x8500 Standard query response, No error
    - Questions: 1
    - Answer RRs: 1
    - Authority RRs: 0
    - Additional RRs: 0
    - Queries
      - www.adatum.com: type A, class IN
        - Name: www.adatum.com
        - [Name Length: 14]
        - [Label Count: 3]
        - Type: A (Host Address) (1)
        - Class: IN (0x0001)
      - Answers
        - www.adatum.com: type A, class IN, addr 172.16.0.10
          - Name: www.adatum.com
          - Type: A (Host Address) (1)
          - Class: IN (0x0001)
          - Time to live: 3600
          - Data length: 4
          - Address: 172.16.0.10

The terminal window shows the command `C:\>ping www.adatum.com` and its output:

```

Ping wird ausgeführt für www.adatum.com [172.16.0.10] mit 32 Bytes Daten:
Antwort von 172.16.0.10: Bytes=32 Zeit<1ms TTL=128

Ping-Statistik für 172.16.0.10:
    Pakete: Gesendet = 4, Empfangen = 4, Verloren = 0
    (0% Verlust),
    Ca. Zeitangaben in Millisek.:
        Minimum = 0ms, Maximum = 0ms, Mittelwert = 0ms
C:\>
    
```

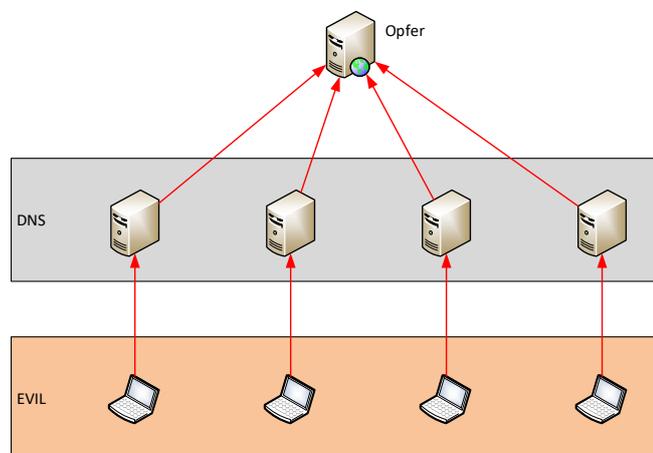
Bei der Anfrage verwendet der Server die IP-Adresse des DNS-Clients und dessen QuellPort. Und natürlich wird im lokalen Netzwerk auch die MAC-Adresse des DNS-Clients verwendet.

Der DNS-Server arbeitet auf diese Weise alle Anfragen brav ab. Er prüft nicht, WER ihn etwas fragt...

### Schema einer DNS Amplification Attacke

Und das kann ein möglicher Angriffsvektor sein. Angenommen, ein Angreifer sendet ein UDP-Paket mit einer Anfrage an einen DNS und fälscht darin seine IP-Adresse. Wer wird wohl die Antwort auf die Frage erhalten? Richtig: der aktuelle Eigentümer der gefälschten IP!

OK, der Client, der keine Anfrage gestellt hat wird die Antwort des DNS-Servers ignorieren. Was aber, wenn der Angreifer nicht nur eine Anfrage sendet, sondern tausende pro Sekunde. Und was wäre, wenn der Angreifer ein Botnetz verwendet, um JEWEILS tausende Anfragen pro Sekunde an verschiedene DNS-Server zu senden – jedes mal mit der gefälschten IP-Adresse seines Opfers? Richtig: das wird ein Denial of Service...



Das ist dann aber nur ein Distributed Denial of Service. Ein Angreifer kann noch weiter gehen und durch die richtigen Abfragen sehr große Antworten von den DNS-Servern provozieren. Er verstärkt den Angriff, was als Amplification bezeichnet wird. Diese Anfrage „kostet“ den Angreifer 74 Bytes.

Das Opfer erhält ein 90 Bytes Antwortpaket:

**\*Ethernet**

Datei Bearbeiten Ansicht Navigation Aufzeichnen Analyse Statistiken Telephonie Wireless Tools Hilfe

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	74	172.16.0.10	DNS	74	Standard query 0xd78c A www.adatum.com
2	0.000314	90	172.16.0.11	DNS	90	Standard query response 0xd78c A www.adatum.com A 172.16.0.10

Frame 2: 90 bytes on wire (720 bits), 90 bytes captured (720 bits) on interface 0  
 Ethernet II, Src: Microsof\_c0:6c:6a (00:15:5d:c0:6c:6a), Dst: Microsof\_c0:6c:6c (00:15:5d:c0:6c:6c)  
 Internet Protocol Version 4, Src: 172.16.0.10, Dst: 172.16.0.11  
 User Datagram Protocol, Src Port: 53, Dst Port: 61726  
 Domain Name System (response)

Findet der Angreifer einen großen Record und fragt diesen ab, dann ist seine Anfrage immer noch recht klein. Aber das Opfer wird geflutet. Die Frage kostet den Angreifer 96 Bytes. Die Antwort ist aber schon 1283 Bytes groß...

**Aufzeichnen von Ethernet**

Datei Bearbeiten Ansicht Navigation Aufzeichnen Analyse Statistiken Telephonie Wireless Tools Hilfe

No.	Time	Source	Destination	Protocol	Length	Info
2	6.649101	82	172.16.0.10	DNS	82	Standard query 0xc5a TXT largerecord.adatum.com
3	6.649665	137	172.16.0.11	DNS	137	Standard query response 0xc5a TXT largerecord.adatum.com SOA lon-dc1.adatum.com
7	6.650034	96	172.16.0.10	DNS	96	Standard query 0xc5a TXT largerecord.adatum.com
9	6.650264	1283	172.16.0.11	DNS	1283	Standard query response 0xc5a TXT largerecord.adatum.com TXT
30	6.048283	75	172.16.0.10	DNS	75	Standard query 0xa3ad A wpad.adatum.com
31	6.049006	140	172.16.0.11	DNS	140	Standard query response 0xa3ad No such name A wpad.adatum.com SOA lon-dc1.adatum.com

Frame 9: 1283 bytes on wire (10264 bits), 1283 bytes captured (10264 bits) on interface 0  
 Ethernet II, Src: Microsof\_c0:6c:6a (00:15:5d:c0:6c:6a), Dst: Microsof\_c0:6c:6c (00:15:5d:c0:6c:6c)  
 Internet Protocol Version 4, Src: 172.16.0.10, Dst: 172.16.0.11  
 Transmission Control Protocol, Src Port: 53, Dst Port: 55895, Seq: 1461, Ack: 43, Len: 1229  
 [2 Reassembled TCP Segments (2689 bytes): #8(1460), #9(1229)]  
 Domain Name System (response)  
 [Request In: 7]  
 [Time: 0.000230000 seconds]  
 Length: 2687  
 Transaction ID: 0xc5a  
 Flags: 0x8580 Standard query response, No error  
 Questions: 1  
 Answer RRs: 1  
 Authority RRs: 0  
 Additional RRs: 0  
 Queries  
 largerecord.adatum.com: type TXT, class IN  
 Name: largerecord.adatum.com  
 [Name Length: 22]  
 [Label Count: 3]  
 Type: TXT (Text strings) (16)  
 Class: IN (0x0001)  
 Answers  
 largerecord.adatum.com: type TXT, class IN  
 Name: largerecord.adatum.com  
 Type: TXT (Text strings) (16)  
 Class: IN (0x0001)  
 Time to live: 3600  
 Data length: 2635  
 TXT Length: 127  
 TXT: 1419369874,2105893006,1894821258,1601230856,1812450941,1109715087,1351295589,394418670,392086563,1856931651,1797791260,84866663  
 TXT Length: 127

```
Administrator: Eingabeaufforderung - powershell
PS C:\> Resolve-DnsName -Name largerecord.adatum.com -Type txt

Name                                     Type    TTL    Section  Strings
----
largerecord.adatum.com                  TXT     3600   Answer   {1419369874,2105893006,1894821258,1601230856,1812450941,1109715087,1351295589,394418670,392086563,1856931651,1797791260,84866663,2,256495276,500520365,745881821,908984769,1152516645,74682107,1279247894,1018190952,390283486,449664132,784250368,115652174,153,8578358,616343602,1905554209,456653227,1491065473,1656909655,1761932134,1901628775,1265554882,609278301,277266618,439085720,1
```

DNSSEC verschärft das Szenario noch zusätzlich. Die nächste Abfrage ist absolut identisch mit der im vorherigen Bild. Zusätzlich habe ich auf dem DNS-Server DNSSEC konfiguriert:

**DNS-Manager**

Datei Aktion Ansicht ?

Name	Typ	Daten	Zeitstempel
ijlm36vmmobr8tp0n32enn9i...	RR-Signatur (RRSIG)	[NSEC3][Inception(UTC): 0...	Static
ijlm36vmmobr8tp0n32enn9i...	Next Secure 3 (NSEC3)	[SHA-1][NO Opt-Out][50]...	Static
kh9pgbhajqf1c1kq4lvjaaha...	RR-Signatur (RRSIG)	[NSEC3][Inception(UTC): 0...	Static
kh9pgbhajqf1c1kq4lvjaaha...	Next Secure 3 (NSEC3)	[SHA-1][NO Opt-Out][50]...	Static
kq3c867iuokbnh9rp9v114j5o...	RR-Signatur (RRSIG)	[NSEC3][Inception(UTC): 0...	Static
kq3c867iuokbnh9rp9v114j5o...	Next Secure 3 (NSEC3)	[SHA-1][NO Opt-Out][50]...	Static
LargeRecord	Text (TXT)	1419369874,2105893006,18...	Static
LargeRecord	RR-Signatur (RRSIG)	[TXT][Inception(UTC): 07...	Static
LargeRecord	RR-Signatur (RRSIG)	[NSEC3][Inception(UTC): 0...	Static

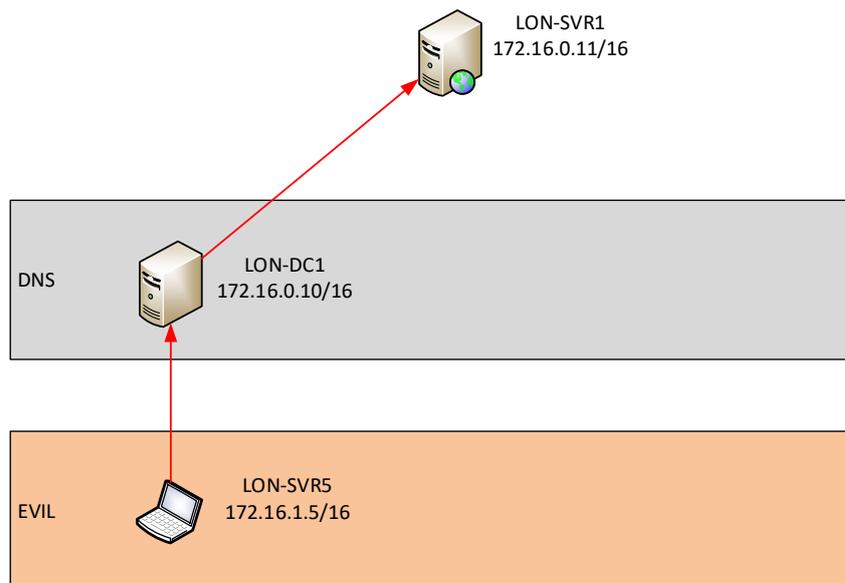
Und hier kommt das Ergebnis beim Client. Nun ist die Antwort schon 1430 Bytes groß:

The screenshot shows a Wireshark capture of a DNS query and response. The response packet (No. 6) has a length of 1430 bytes. A red arrow points to this value. Below the capture, a PowerShell terminal window shows the output of the command `Resolve-DnsName -Name largerecord.adatum.com -Type txt -DnssecOk`. The output lists 16 TXT records for the domain largerecord.adatum.com, each with a TTL of 3600. The records are: {1419369874, 894821258, 12450941, 11095589, 394413, 185693165, 84866663, 20520365, 745769, 1152516, 1279247894, 0283486, 448368, 1156521, 58, 61634366, 456653227, 16909655, 17628775, 1265501, 27726661, 16, 4705769, 0460176, 673072, 1380524, 2099741248, 6603226, 21275147, 38101}.

### Beispielszenario eines Angriffes

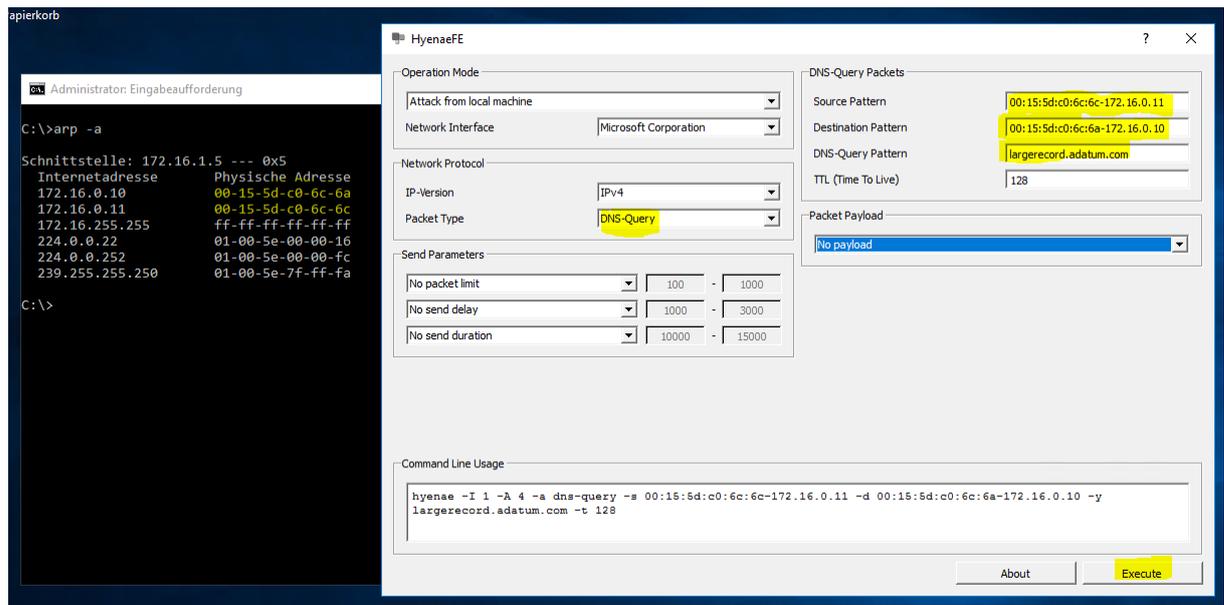
Ein kleines LAB soll das Problem sichtbar machen. Ich verwende 3 Server:

- einen DNS-Server (LON-DC1 mit der IPv4 172.16.0.10),
- ein Opfer (LON-SVR1 mit der IPv4 172.16.0.11)
- und einen Angreifer-Server (LON-SVR5 mit der IP 172.16.1.5)



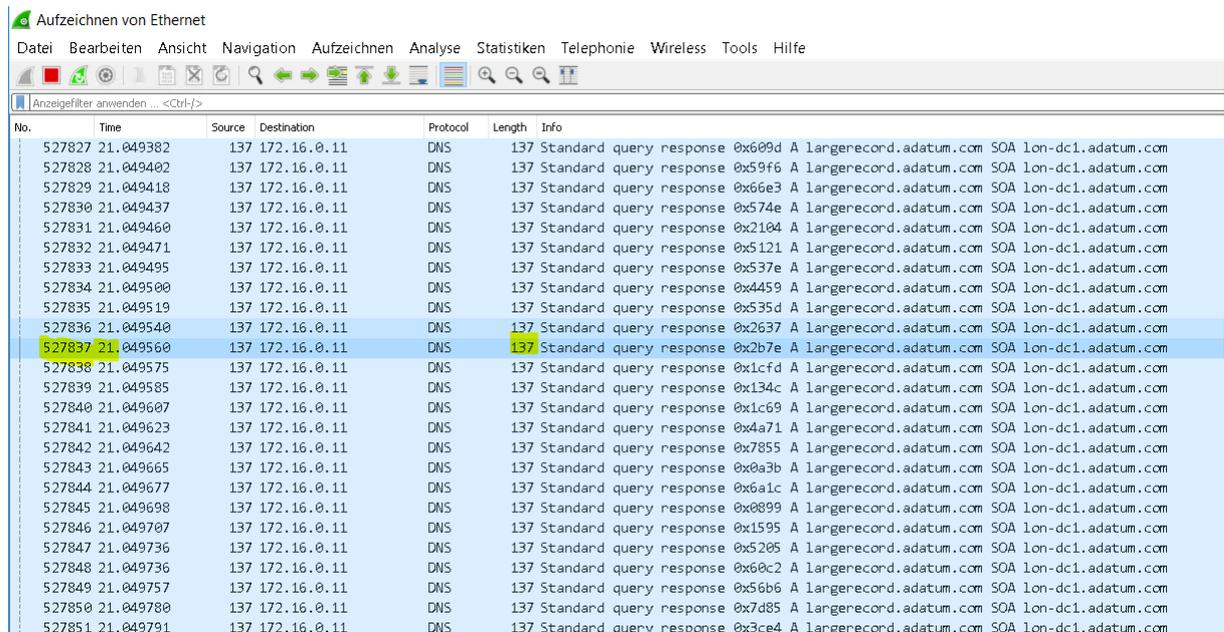
Auf dem Angreifer-Server verwende ich das Tool Hyenae, um die DDOS-Attacke zu starten. Diese sendet extrem viele Anfragen „im Auftrag von Server LON-SVR1“ an den LON-DC1. Und dieser wird LON-SVR1 auch brav antworten.

Und so geht's. Auf LON-SVR5 bringe ich Hyenae in Stellung. Als Source verwende ich die MAC und die IPv4 von LON-SVR1 (mein Opfer). Als Destination verwende ich die MAC und die IPv4 vom DNS-Server:



Der Record ist mein LargeRecord – ein dicker TXT-Eintrag mit DNSSEC bei etwa 1400 Bytes gröÙe – pro Antwort!!!

Auf LON-SVR1 verwende ich WireShark, um die Antworten vom DNS-Server zu sehen. Und los geht's:



Mein Opfer hat Glück, dass die einfache Version nur Host-A-Records abfragen kann. Da sind die über 500.000 Antworten in 21 Sekunden nur etwa 70 MB statt 700 MB groß. Und es war nur 1 DNS!

Ich denke, das Problem ist nun klar...

## Ein Schutz - Response Rate Limiting?

### Was ist RRL und wie funktioniert es?

Die Antwort hat Microsoft in den DNS-Service ab Windows Server 2016 eingebaut: Response Rate Limiting. Dabei wird der Server prüfen, welcher Client wie oft eine Anfrage stellt. Wenn das Anfrageaufkommen einen Schwellwert überschreitet, dann werden die Antworten für einen Zeitraum gedrosselt. Es wird also nicht mehr jede Anfrage beantwortet.

ACHTUNG: Falsch konfiguriert wird so ein Denial Of Service für alle Clients und Server vorprogrammiert sein!!!

Leider wurde die Option nicht in die grafische Oberfläche eingebaut. Die Steuerung ist nur über die PowerShell möglich:

```
PS C:\> Get-Command -Name *response*

CommandType      Name                                     Version      Source
-----
Function         Add-DnsServerResponseRateLimitingExceptionlist 2.0.0.0     DnsServer
Function         Get-DnsServerResponseRateLimiting              2.0.0.0     DnsServer
Function         Get-DnsServerResponseRateLimitingExceptionlist 2.0.0.0     DnsServer
Function         Remove-DnsServerResponseRateLimitingExceptionlist 2.0.0.0     DnsServer
Function         Set-DnsServerResponseRateLimiting              2.0.0.0     DnsServer
Function         Set-DnsServerResponseRateLimitingExceptionlist 2.0.0.0     DnsServer

PS C:\>
```

Es sind bereits einige Einstellungen per default gesetzt. Die Aktivierung von RRL fehlt aber noch:

```
PS C:\> Get-DnsServerResponseRateLimiting

ResponsesPerSec      : 5
ErrorsPerSec         : 5
WindowInSec          : 5
IPv4PrefixLength     : 24
IPv6PrefixLength     : 56
LeakRate              : 3
TruncateRate         : 2
MaximumResponsesPerWindow : 1024
Mode                  : Disable

PS C:\>
```

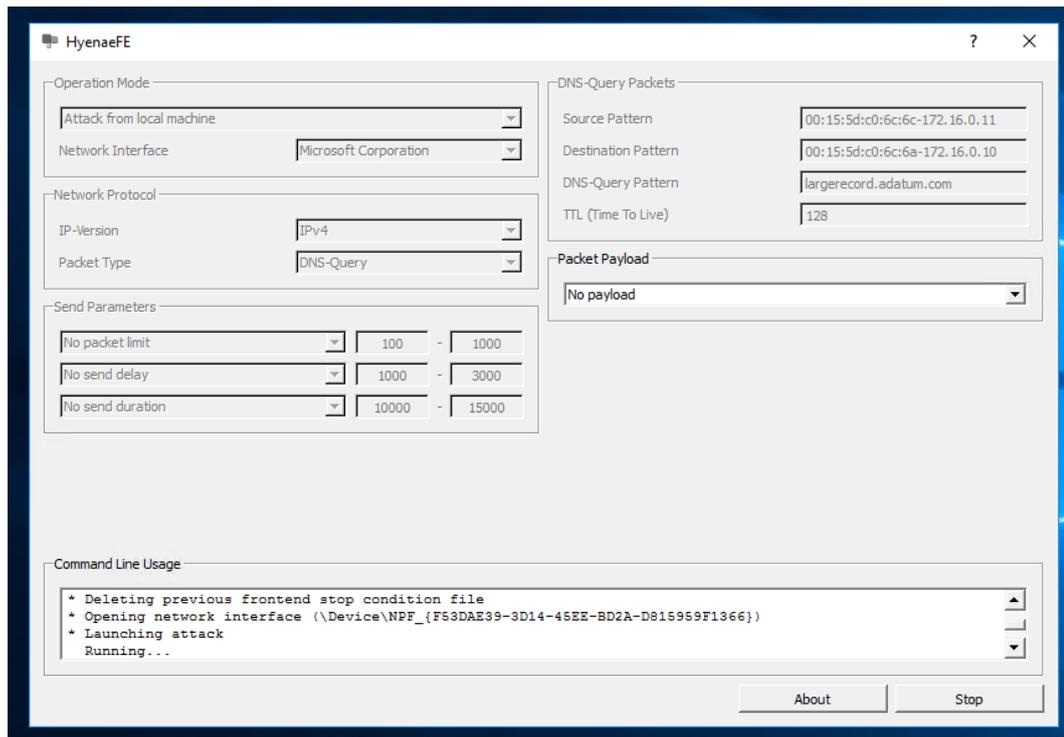
Dann aktivieren wir doch RRL einmal. Nach einer Bestätigung und einer Warnung, das RRL falsch konfiguriert selbst ein Denial of Service für DNS bewirken kann, ist das neue Feature aktiviert:

```
PS C:\> Set-DnsServerResponseRateLimiting -Mode Enable
WARNUNG: Response Rate Limiting kann zu Denial-of-Service-Angriffen auf Clients führen, wenn eine Vielzahl ähnlicher Anforderungen festgestellt wird, die aus einer gemeinsamen Quelle stammen.

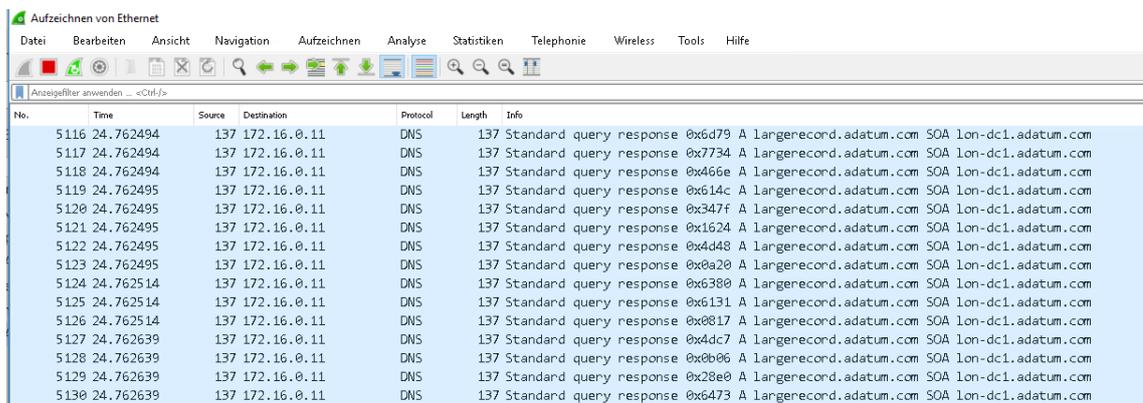
PS C:\>
```

### Beispielszenario eines Angriffes mit aktiviertem RRL

Dann bleibt nur noch ein ausstehender Test von RRL. Das Szenario ist das gleiche wie im ersten Kapitel: LON-SVR5 startet mit Hyenae eine DOS-Attacke und sendet dabei sehr viele DNS-Anfragen „im Auftrag von LON-SVR1“ an den DNS-Server:



Auf LON-SVR1 läuft derweil wieder Wireshark und schneidet für uns die Antworten des DNS-Servers mit. Der Test läuft auch etwas mehr als 20 Sekunden. In dieser Zeit hatte der DNS-Server ohne RRL über 500.000 Antworten gesendet. Und mit RRL?



No.	Time	Source	Destination	Protocol	Length	Info
5116	24.762494	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x6d79 A largerecord.adatum.com SOA lon-dc1.adatum.com
5117	24.762494	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x7734 A largerecord.adatum.com SOA lon-dc1.adatum.com
5118	24.762494	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x466e A largerecord.adatum.com SOA lon-dc1.adatum.com
5119	24.762495	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x614c A largerecord.adatum.com SOA lon-dc1.adatum.com
5120	24.762495	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x347f A largerecord.adatum.com SOA lon-dc1.adatum.com
5121	24.762495	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x1624 A largerecord.adatum.com SOA lon-dc1.adatum.com
5122	24.762495	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x4d48 A largerecord.adatum.com SOA lon-dc1.adatum.com
5123	24.762495	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x0a20 A largerecord.adatum.com SOA lon-dc1.adatum.com
5124	24.762514	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x6380 A largerecord.adatum.com SOA lon-dc1.adatum.com
5125	24.762514	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x6131 A largerecord.adatum.com SOA lon-dc1.adatum.com
5126	24.762514	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x8817 A largerecord.adatum.com SOA lon-dc1.adatum.com
5127	24.762639	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x4dc7 A largerecord.adatum.com SOA lon-dc1.adatum.com
5128	24.762639	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x0b06 A largerecord.adatum.com SOA lon-dc1.adatum.com
5129	24.762639	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x28e0 A largerecord.adatum.com SOA lon-dc1.adatum.com
5130	24.762639	137.172.16.0.11	137.172.16.0.11	DNS	137	Standard query response 0x6473 A largerecord.adatum.com SOA lon-dc1.adatum.com

Jetzt sind es noch etwas über 5000 Antworten von DNS-Server an das Opfersystem! 😊

**Und so funktioniert es:** der DNS-Server sendet per Default mit aktiviertem RRL nur 1024 identische Antworten in einem Zeitraum von 5 Sekunden an einen Client, der identische Fragen stellt:

```
PS C:\> Get-DnsServerResponseRateLimiting

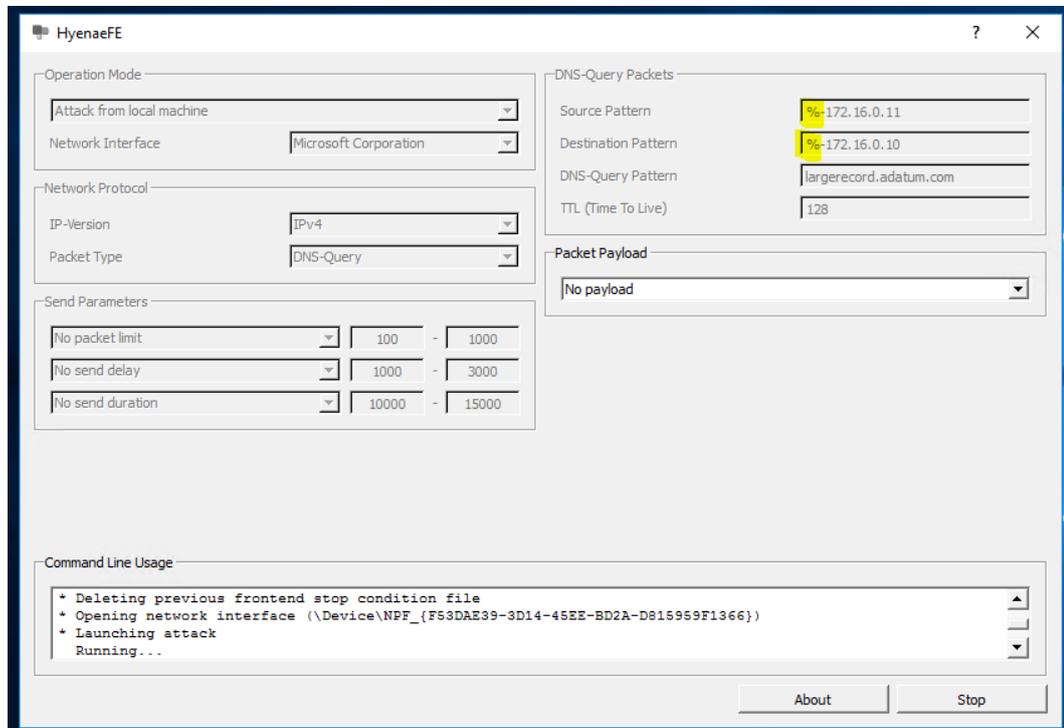
ResponsesPerSec      : 5
ErrorsPerSec         : 5
WindowInSec          : 5
IPv4PrefixLength     : 24
IPv6PrefixLength     : 56
LeakRate              : 3
TruncateRate         : 2
MaximumResponsesPerWindow : 1024
Mode                  : Enable
```

Und das sind in 25 Sekunden eben etwas mehr als 5000 Antwortpakete. 😊

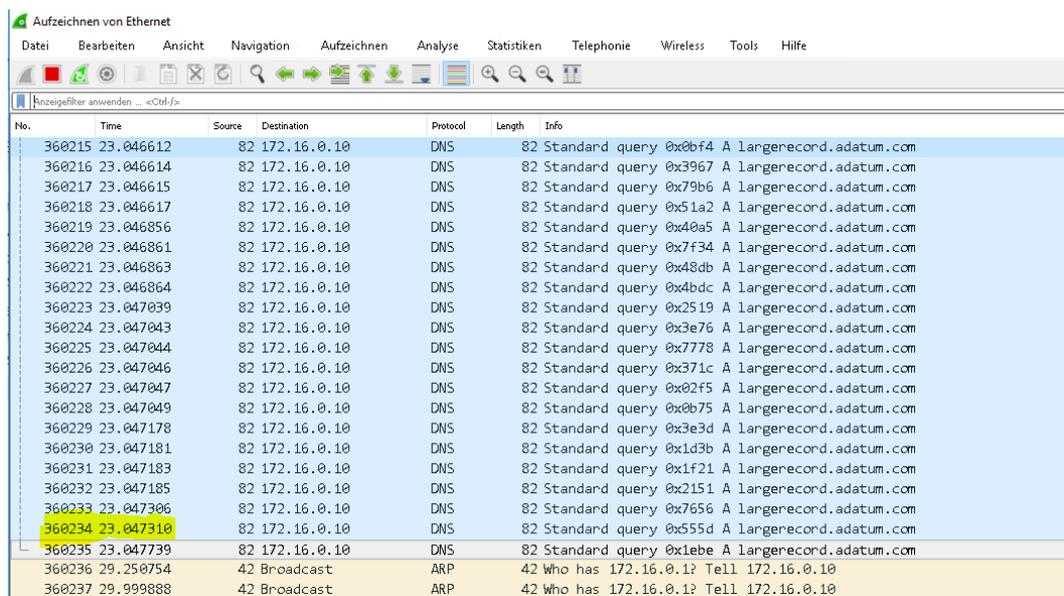
### Katz und Maus

Leider genügen die Mechanismen von RRL nur bestimmten Rahmenbedingungen. Und diese setzen neben gleichen Absender-IP-Adressen auch gleiche MAC-Adressen voraus. Nur in der Kombination wird ein Client eindeutig gekennzeichnet.

Wenn ich in Hyenae nun einen Parameter der Attacke ändere (gefälschte MAC-Adressen)...



... dann versagt RRL, da der DNS-Server nicht mehr einen Client sondern ganz viele Clients erkennt. Und jeder bekommt in einem Zeitfenster die 1024 Standardantworten:



No.	Time	Source	Destination	Protocol	Length	Info
360215	23.046612	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x0bf4 A largerecord.adatum.com
360216	23.046614	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x3967 A largerecord.adatum.com
360217	23.046615	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x79b6 A largerecord.adatum.com
360218	23.046617	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x51a2 A largerecord.adatum.com
360219	23.046856	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x40a5 A largerecord.adatum.com
360220	23.046861	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x7f34 A largerecord.adatum.com
360221	23.046863	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x48db A largerecord.adatum.com
360222	23.046864	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x4bdc A largerecord.adatum.com
360223	23.047039	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x2519 A largerecord.adatum.com
360224	23.047043	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x3e76 A largerecord.adatum.com
360225	23.047044	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x7778 A largerecord.adatum.com
360226	23.047046	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x371c A largerecord.adatum.com
360227	23.047047	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x02f5 A largerecord.adatum.com
360228	23.047049	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x0b75 A largerecord.adatum.com
360229	23.047178	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x3e3d A largerecord.adatum.com
360230	23.047181	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x1d3b A largerecord.adatum.com
360231	23.047183	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x1f21 A largerecord.adatum.com
360232	23.047185	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x2151 A largerecord.adatum.com
360233	23.047306	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x7656 A largerecord.adatum.com
360234	23.047310	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x555d A largerecord.adatum.com
360235	23.047739	82.172.16.0.10	172.16.0.10	DNS	82	Standard query 0x1ebe A largerecord.adatum.com
360236	29.250754	42	Broadcast	ARP	42	Who has 172.16.0.1? Tell 172.16.0.10
360237	29.999888	42	Broadcast	ARP	42	Who has 172.16.0.1? Tell 172.16.0.10

Dennoch kann RRL einen Teil des Angriffsvektors vereiteln. Es lohnt sich also, diese Funktion zu evaluieren!