DNS-based data exfiltration and infiltration technology and detection

By: Yanhui Jia, Rongbo Shao, Yi Ren, Matt Tennis, Xin Ouyang, John Harrison, Jens Egger, Ashwin Dewan

In this paper, we introduce the types, methods, and usage of DNS-based data infiltration and exfiltration, and the mitigation strategies provided by Palo Alto Networks. DNS-based evasion techniques are on the rise, with popular toolkits such as iodine, dns2tcp, OzymanDNS, SplitBrain, tcp-over-dns, and DNS Messenger frequently being deployed in the wild. When utilizing a DNS tunnel, attackers can discreetly exfiltrate data or create covert command and control channels, allowing for further exploitation and data loss.

We will discuss various real-world use cases of DNS tunneling tools, and how we can detect and prevent these kinds of cyber-attacks.

What is DNS Tunneling?

In order to discuss DNS tunneling, it is prudent to briefly describe DNS. The Domain Name System (DNS) is a distributed directory that resolves human-readable hostnames, such as www.paloaltonetworks.com, to machine-readable IP addresses like 23.41.44.222. DNS is also a directory of crucial information about domain names, such as email servers (MX records), sending verification (DKIM, SPF, DMARC), TXT record verification of domain ownership, and even SSH fingerprints (SSHFP).

DNS is widely used on nearly every system that requires general purpose use of the Internet, but it is trivial to subvert for malicious ends. DNS-based data exfiltration and infiltration are two significant misuse cases in the wild. Malicious actors can use DNS-based tunneling to steal user data or sensitive information, or drop a second stage Trojan on a compromised host. The impact against any organization that utilizes the Internet can be catastrophic, and the use of this technique is indifferent to industry. DNS tunneling is here to stay, and it is quickly becoming a significant universal threat.

DNS-Based Tunnel Data Exfiltration

DNS-based data exfiltration is unauthorized data transfer via the DNS protocol, despite the fact that it is not designed for tunneling. In the simplest attack case, as seen in the following diagram, the attacker sets up a name server (NS) for a controlled domain, e.g. dns.exf.com. Sensitive data can then be covertly exfiltrated in a DNS request to the attacker by prepending data to the domain. This case uses simple encoding to obfuscate detection, e.g. base64_encode(data).dns.exf.com.

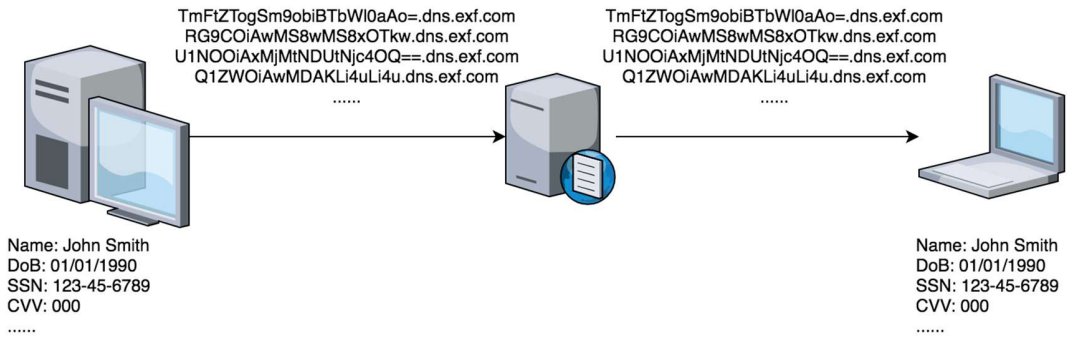

Figure 1 a simple example of data exfiltration

Generic Method for DNS-Base Data Exfiltration

The implementations of DNS tunneling tools vary, but the concept remains the same: exfiltrate encoded data hidden in the DNS query section. Some tools provide configurable parameters including encoding methods, data segment length, query type, etc., but many more use static settings. The following are brief demonstrations of common exfiltration methods.

```
Domain Name System (query)
    [Response In: 12]
    Transaction ID: 0x17db
  > Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ~ Queries
    ~ 646f776e6c6f61642e706e67.1.6c27f0.png.1567.start.scr.f61591a603[redacted] type A, class IN
        Name: 646f776e6c6f61642e706e67.1.6c27f0.png.1567.start.scr.f61591a603.[redacted]
        [Name Length: 73]
        [Label Count: 10]
        Type: A (Host Address) (1)
        Class: IN (0x0001)
```

Figure 2 Data Exfiltration via DNS A Record

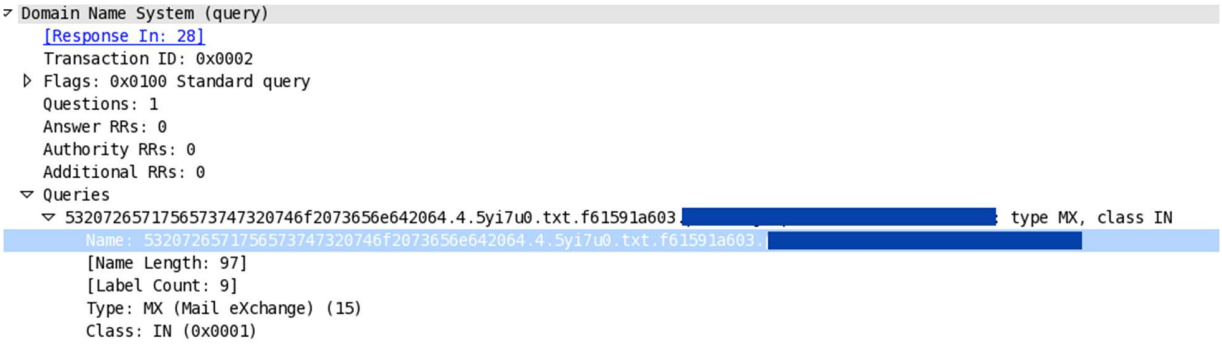

Figure 3 Data Exfiltration via DNS MX Record

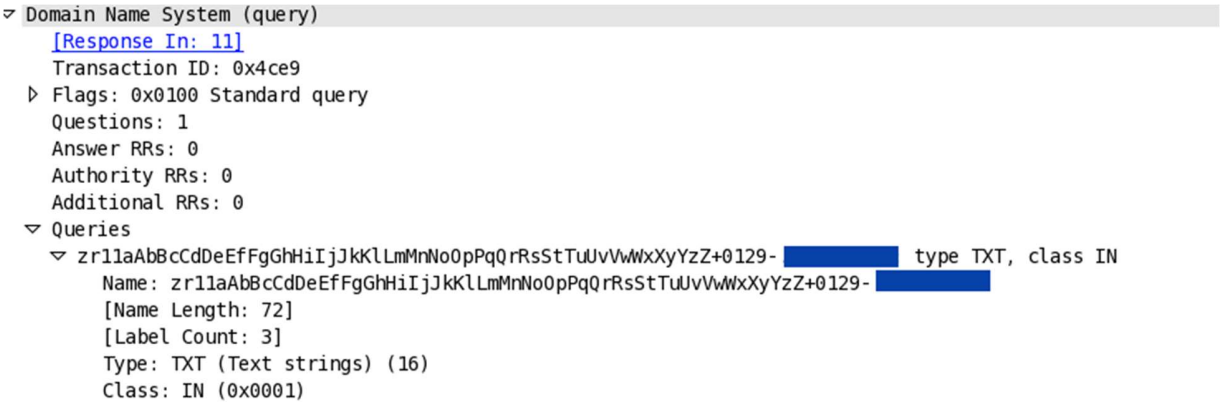

Figure 4 Data Exfiltration via DNS TXT Record

Figure 5 Data Exfiltration via DNS CNAME Record

Typical Exfiltration Toolkits

Here is an example from SplitBrain. As an enhancement of OzymanDNS, SplitBrain is able to work as SSH proxy, which enables its users to transfer data through SSH-over-DNS channel.

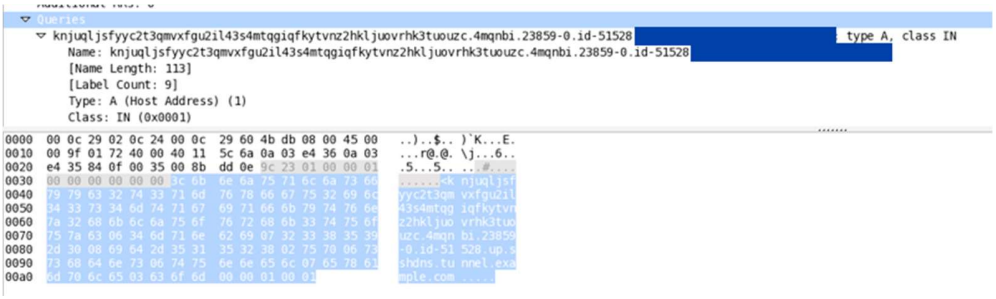

Figure 6 SplitBrain DNS tunneling data in DNS A record

```
>>> a = "knjuqljsfyyc2t3qmvxfgu2il43s4mtqgiqfkytvnz2hkljuovrhk3tuouzc4mqn"
>>> b = a.upper()
>>> c = base64.b32decode(b)
>>> c
'SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.2\r'
```

Figure 7 Base32 decoded data

DNS-Based Tunnel Data Infiltration

DNS data infiltration is the reversal of the exfiltration method. Either plain text or encoded text is added into DNS queries or replies to avoid detection. There are three prominent use cases for DNS infiltration. First, this technique is used to inject second-stage payloads into the target network. Second, a covert tunnel can be used in conjunction with data exfiltration tactics to form a reliable command and control (C2) channel. Third, it can be simply used as a data channel.

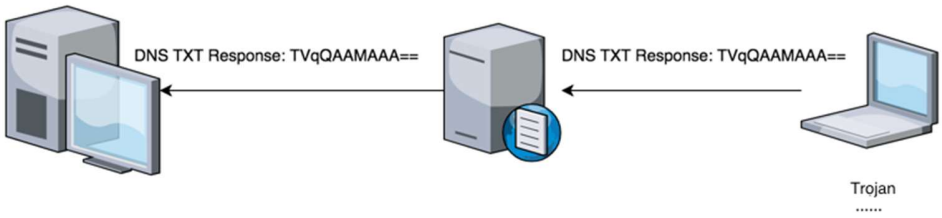

Figure 8 a simple example of data infiltration

Generic Method for DNS-Based Data Infiltration

Infiltration and exfiltration both exploit the same functionality of the DNS protocol, with the position and encoding of data within a DNS packet remaining mostly the same. The primary difference is that the destination of infiltration is a host under most circumstances. Infiltration data is usually in the answer section of DNS replies, and the rest of the reply is normal and does not increase the risk of detection.

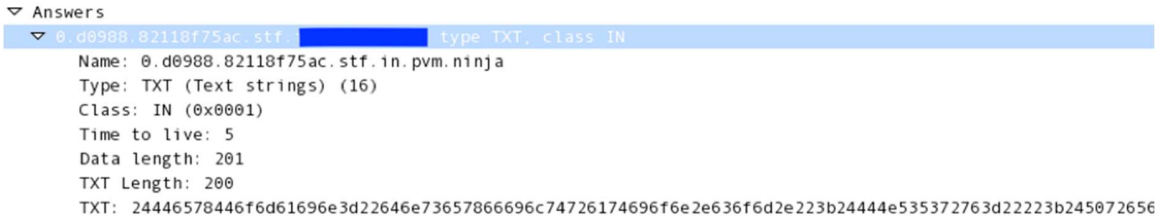

Figure 9 Infiltration within TXT record

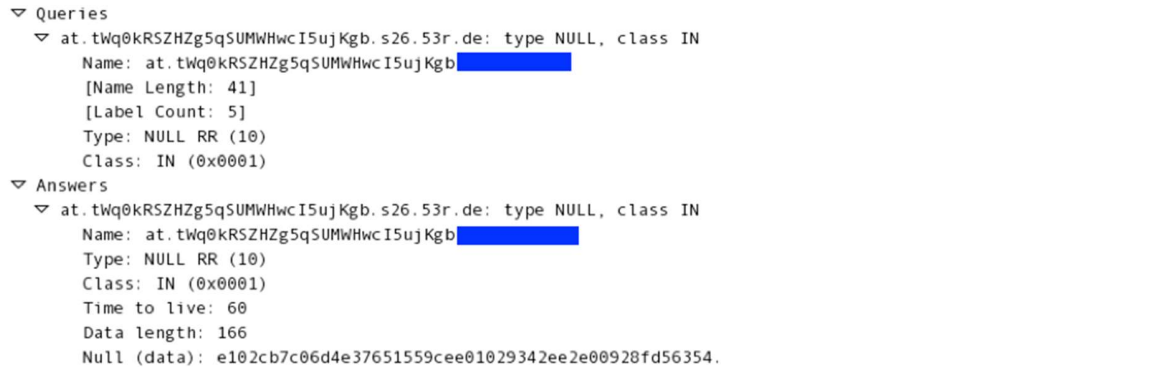

Figure 10 Infiltration within Null record as seen in YourFreedom tunnel tool

Typical Infiltration Malware

DNS Messenger is a Remote Access Trojan (RAT). Upon infecting a system, it utilizes a DNS tunnel to drop a Windows Web shell and execute it without any code written to the filesystem. After execution it then establishes a covert C2 channel through DNS tunneling.

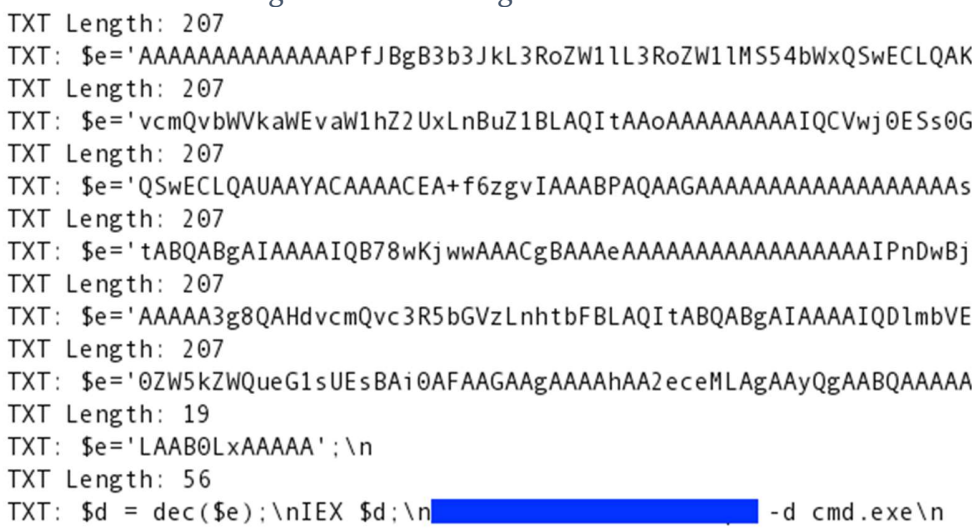

Figure 11 PowerShell Binaries and Command

Figure 11 shows the delivery of the web shell binaries and the command to execute it.

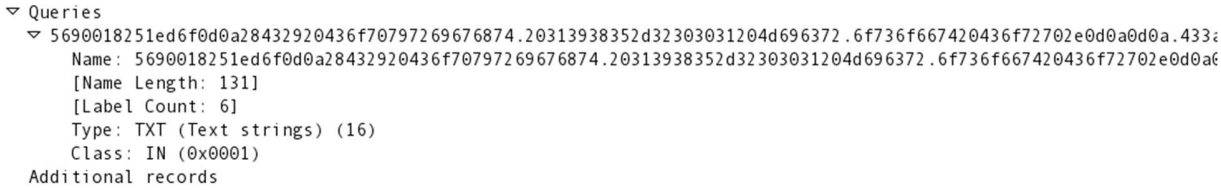

Figure 12 C2 Channel with Windows Command Line

Figure 12 shows the C2 channel what was returned to C2 server which decode to "Copyright 1985-2001 Microsoft Copt.\r\n\r\nC:\\" as the Windows command line prompt.

Detecting DNS-Based Tunneling

1. Pattern-based

   Currently, most modern spyware still uses constant ASCII or hexadecimal strings to initialize a C2 handshake, heartbeat, or tunnel. As a consequence, pattern-based detection methods are still very reliable for a significant amount of attacks. Open-source technologies such as Snort and Suricata employ pattern-based signatures to prevent a variety of attacks including network intrusions, malware and spyware, and static DNS exploitation. Palo Alto Networks also provides robust signature-based detection methods, as well as detection logic for advanced malware and toolkits. Pattern-based detection is still effective against some instances of DNS tunneling, and a majority of the millions of older threats that are still active in the wild.

   As an example, we demonstrate captured DNScapy traffic, which creates an SSH tunnel over DNS. In addition to an SSH connection, SCP file transfer and SOCKS proxying (ssh -D) are also supported. CNAME or TXT records can be used for the tunnel. The default mode is RAND, which randomly uses both CNAME and TXT records.

   From the figures below, we can extract the SSH tunnel response message "CNAME:a.8.0.U1NILTIuMC1PcGVuU1NIXzcuMnAyIFVidW50dS00dWJ1bnR1Mi4yDQo=", which is encoded within the DNS tunnel. This base64 string decodes to "SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.2". We can use "U1NIL" as a pattern, along with other identifiers, to uniquely identify and detect this kind of DNS tunneling.

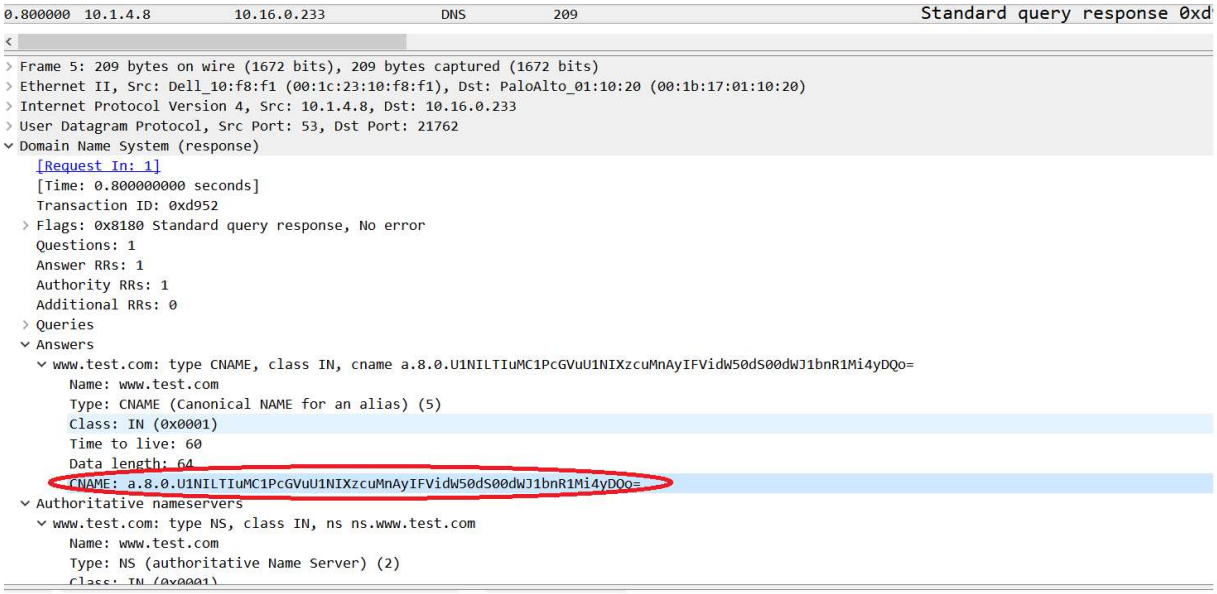

Figure 13 DNScapy SSH tunnel connects through DNS

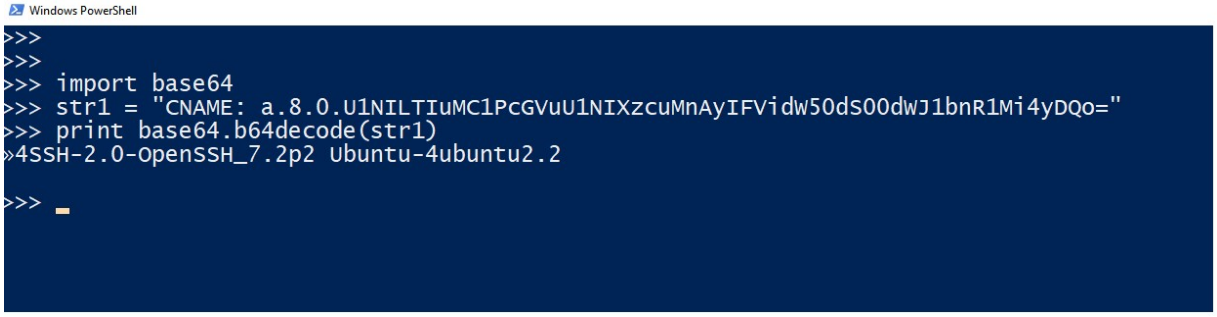

Figure 14 DNS tunnel data decoding

2. Behavior-based
   Despite the efficacy of pattern-based detection, malware family variants frequently have modified or obfuscated C2 signals which can evade detection signatures. In order to solve these cases, we implement behavior-based detection models. Many attributes of a DNS query or response can be analyzed to detect malicious traffic.

Behavior	Description
Number of subdomains	How many subdomains are present in the domain request
Packet number in domain	What's the number of the packet in data transmission

Encoding in domain or TXT record	Whether the domain or TXT record includes evidence of base64/32 encoding.
Domain and subdomain length	Suspicious domain/subdomain length
Counts in different session	Data statistics in different session

As an example, we examine an excerpt from the OilRig malware traffic listed below. The malware is utilizing a DNS tunnel as a covert data exfiltration and C2 channel. The first subdomain is encoded and has a constant length of 13 bytes, the second subdomain is a sequence number that is one byte long, and the third subdomain is another one byte constant character followed by the domain. We observed multiple sessions with these features, which enabled us to create reliable behavior-based detection for OilRig's C2 channel. This can also be modified to catch variants of this pattern, especially if a pattern-based solution is not sufficient.

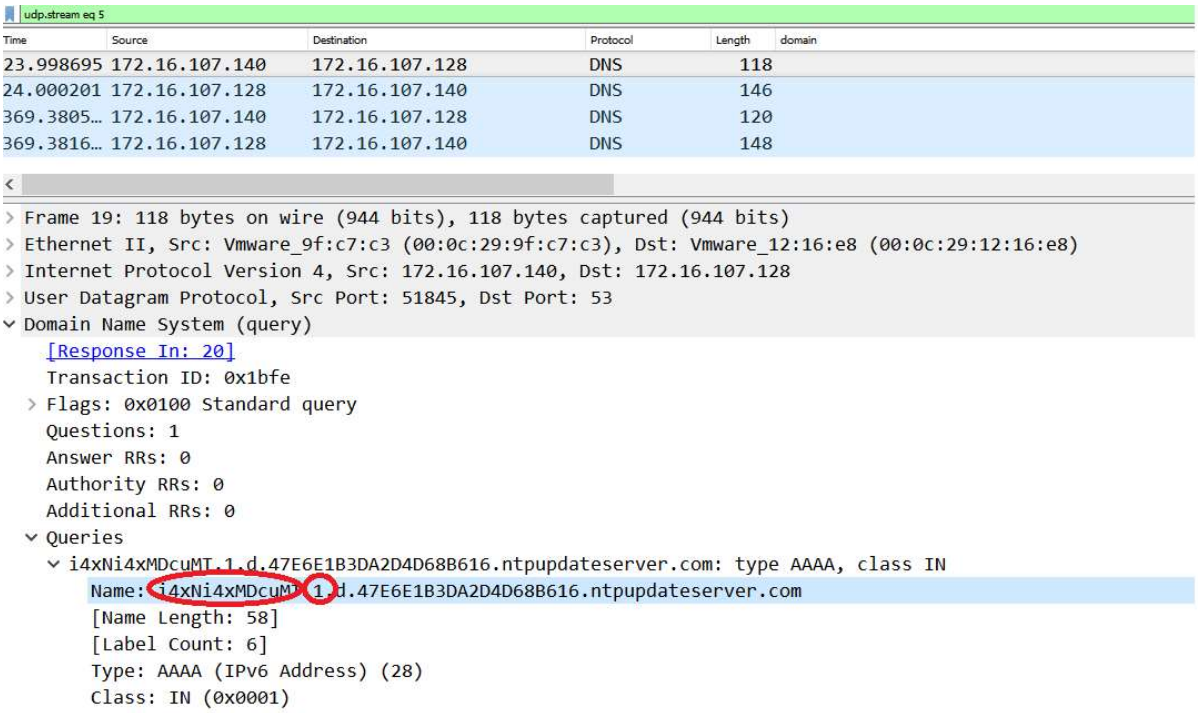

Figure 15 Oilrig DNS-based tunnel traffic

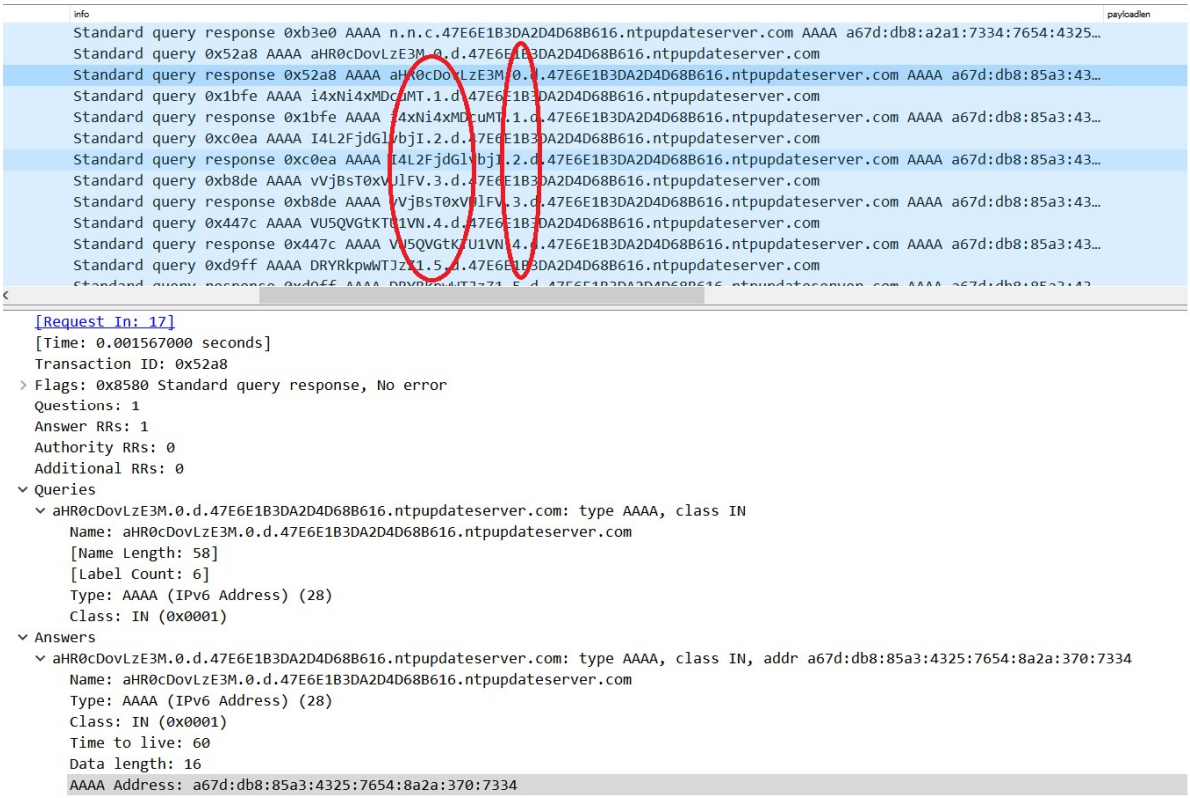

Figure 16 OilRig DNS-based tunnel - multiple sessions

3. Mixed Pattern- and Behavior-Based Detection Method

   False positives (FPs) and false negatives (FNs) are the most consequential indicators for threat detection. If benign traffic is blocked because of a FP, the result can be disastrous for customers. Likewise, allowing malicious traffic to pass undetected defeats the purpose of threat detection and prevention. As a result, FP and FN rates must be kept as low (> 0.1%) as possible. In order to achieve this goal, we often integrate both of these detection models in order to prevent FPs and FNs and improve our detection rate.

4. Machine Learning

   Palo Alto Network's Magnifier Behavioral Analytics platform employs machine learning to empower our customers to quickly detect network threats, including DNS tunneling. Below are real scenarios of Magnifier in action.

   - A host is periodically accessing an external domain that is rarely used by the host, or by other hosts in its peer group. The access to this domain has occurred repeatedly, over many days. Analysis of the connection pattern shows that it is consistent with malware connecting to its command and control server for updates and operating instructions. Per host and per peer group: **Every day Magnifier examines the previous 14 days worth of log data.**
   - Magnifier flagged that behavior as abnormal because no other machine communicated with that domain, and automated investigation and threat

intelligence showed that the domain was marked as suspicious by a third party. The customer cleaned up the machine and mitigated their risk.

- There was also another example during an evaluation - an infected machine was making many DNS requests that failed to resolve, which is a behavioral indicator of infected machines. Traps was not installed because the machine is an IoT-type device with 1GB of memory. Magnifier flagged it as Random-Looking Domain Names and Failed DNS since its unusual behavior compared to other devices. After investigating the alerts and the destinations, the results indicated that the DNS requests are known malicious domains belonging to the Locky ransomware campaign. The customer cleaned up the infected machines and mitigated their risk.

5. Toolkit and Malware List

Tool Name	Description
dns2tcp	dns2tcp was written by Olivier Dembour and Nicolas Collignon. It is written in C and runs on Linux. The client can run on Windows. It supports KEY and TXT request types. [1]
tcp-over-dns	tcp-over-dns (TCP-over-DNS) was released in 2008. It has a Java based server and a Java based client. It runs on Windows, Linux and Solaris. It supports LZMA compression and both TCP and UDP traffic tunneling. [1]
OzymanDNS	OzymanDNS is written in Perl by Dan Kaminsky in 2004. It is used to setup an SSH tunnel over DNS or for file transfer. Requests are base32 encoded and responses are base64 encoded TXT records. [1]
iodine	iodine is a DNS tunneling program first released in 2006 with updates as recently as 2010. It was developed by Bjorn Andersson and Erik Ekman. Iodine is written in C and it runs on Linux, Mac OS X, Windows and others. Iodine has been ported to Android. It uses a TUN or TAP interface on the endpoint. [1]
SplitBrain	SplitBrain is a variant of OzymanDNS.
DNScat-P / dnscat2	DNScat (DNScat-P) was originally released in 2004 and the most recent version was released in 2005. It was written by Tadeusz Pietraszek. DNScat is presented as a “Swiss-Army knife” tool with many uses involving bi-directional

	communication through DNS. DNScat is Java based and runs on Unix like systems. DNScat supports A record and CNAME record requests (Pietraszek, 2004). Since there are two utilities named DNScat, this one will be referred to as DNScat-P in this paper to distinguish it from the other one. [1]
DNScapy	DNScapy was developed by Pierre Bienaime. It uses Scapy for packet generation. DNScapy supports SSH tunneling over DNS including a Socks proxy. It can be configured to use CNAME or TXT records or both randomly. [1]
TUNS	TUNS, an IP over DNS tunnel, was developed by Lucas Nussbaum and written in Ruby. It does not use any experimental or seldom used record types. It uses only CNAME records. It adjusts the MTU used to 140 characters to match the data in a DNS request. TUNS may be harder to detect, but it comes at a performance cost.
PSUDP	PSUDP was developed by Kenton Born. It injects data into existing DNS requests by modifying the IP/UDP lengths. It requires all hosts participating in the covert network to send their DNS requests to a Broker service which can hold messages for a specific host until a DNS request comes from that host. The message can then be sent in the response.
Your Freedom	HTTPS/UDP/FTP/DNS/ECHO VPN & tunneling solution for Windows, Mac OSX, Linux and Android. Bypass proxies and access the Internet anonymously
Hexify	A tool is developed by Infoblox to do the penetrating test for DNS tunneling.

Malware Name	Description
DNS_TXT_Pwnage	A backdoor capable of receiving commands and PowerShell scripts from DNS TXT queries.
DNSMessenger	DNSMessenger is Remote Access Trojan (RAT) that opens a backdoor so that hackers can control the compromised machine remotely
OilRig - BONDUPDATER	Trojan against a Middle Eastern government can use A records and TXT records within its DNS tunneling protocol for its C2 communications

Summary

DNS tunneling is a rapidly growing evasion technique that is becoming more popular with a variety of malware campaigns. Attackers deploy this technology in order to bypass network security appliances. Malicious actors can covertly infiltrate and exfiltrate sensitive data, issue command and control signals, and deploy multi-stage attacks, including advanced persistent threat (APT) campaigns.

DNS tunneling represents a significant threat to our customers and is becoming more effective. Evasion techniques can be fingerprinted into a network signature, but more cases require advanced behavioral and machine learning technology. Palo Alto Network's Next Generation Firewall and Magnifier service provides protection from DNS tunneling channels for our customers.

- DNS-based tunneling tools, toolkits, and malware can be mitigated with IPS-Content. Please be sure to update your content to the latest version.
- All identified C2 domains that we have discovered have been covered and released by PAN-DB.
- The Magnifier service can also empower our customers to detect and analyze unknown and APT attacks with a robust machine leaning model that is kept up-to-date with the evolving threat landscape.

References

[1] https://www.sans.org/reading-room/whitepapers/dns/detecting-dns-tunneling-34152