7-2012

A Covert Channel in TTL Field of DNS Packets

Christopher Hoffman
Rochester Institute of Technology

Daryl Johnson
Rochester Institute of Technology

Bo Yuan
Rochester Institute of Technology

Peter Lutz
Rochester Institute of Technology

A Covert Channel in TTL Field of DNS Packets

Christopher Hoffman, Daryl Johnson, Bo Yuan, Peter Lutz
Rochester Institute of Technology
{cwh4129,daryl.johnson,bo.yuan,peter.lutz}@rit.edu

***Abstract*—Covert channels are used as a means of secretly transferring information when there is a need to hide the fact that communication is taking place. With the vast amount of traffic on the internet, network protocols have become a common vehicle for covert channels, typically hiding information in the header fields of packets. Domain name service (DNS) packets contain a 32-bit time to live (TTL) fields for each response record. This is the number of seconds the entry is valid for before caching servers remove the entry. There is no prescribed value for this field making it an ideal covert carrier.**

I. Introduction

The most common technique for transferring secure information is cryptography, using algorithms to mask what is being communicated [1]. Although this makes the information unreadable by third-parties, it does not hide the fact that communication is taking place. Covert channels are utilized to attempt to hide the existence of a communication channel. The original application of covert channels was to solve the Prisoners Problem, where two parties wanted to communicate but communication was mediated by a warden who was able to read the messages and determine if they were allowed. They had to devise a way to hide their secret conversation in a way that would not look suspicious.

Covert channels can be categorized into three categories: storage, timing, and behavior[2], [3]. Storage channels use a shared storage medium between two parties where one party writes and one reads. These include networking protocols or disk storage. Timing channels use relative timing of events to convey information using patterns. This is normally done by purposely altering resources such as CPU or other resource utilization. Behavior channels function more at the application layer and information is conveyed by selecting certain actions. An example of this is selecting certain moves in a board game.

With the vast amount of traffic on the internet, network protocols have become a common vehicle for covert channels, typically hiding information in the header fields of packets. DNS has a time-to-live(TTL) field, a 32-bit unsigned integer, which denotes how long the (domain name, address) pair is valid for, measured in seconds. This is different from the IP TTL field which is the number of hops a packet can make before being removed from the network. This keeps a packet from traveling in a continuous loop.

The DNS protocol does not outline what the TTL value should be so it is possible to assign it any desired value. Although most authoritative servers use discrete increments such as hours and days, due to caching, it is not uncommon to see values with minutes and seconds. This amount of uncertainty makes the TTL field ideal for a covert channel. Different methods for encoding information can be used to alter the bandwidth and covertness of the channel.

II. Related Work

There has not been much work using DNS to transfer information other than encoding information in the domain name. Typically it is more common to use DNS as a device for tunneling TCP to restricted areas [4]. This is due to the robustness of DNS. In systems that require payment or authentication, DNS traffic is typically still allowed to ensure users do not cache incorrect information.

Even though multiple answers can be returned for each question, a long packet length can appear suspicious. Omar, Ahmedy, and Ngadi examined this issue while they were working with a DNS covert channel for tunneling IP [5]. Their work involved using an alphabet of domain names, each domain name corresponding to a set of characters. To increase bandwidth, more question/answer segments could be included per packet but this increased the size of the packet. The packet would become suspicious if it was either much shorter or longer than normal DNS packets. By keeping the packets near a normal size, the channel would be less suspicious.

Zander, Armitage, and Branch devised a covert channel using the IP TTL field [6]. The effectiveness of this field is dependent on the natural variation in the carrier network and the ability to encode information in this natural carrier. Since the hop distance between two hosts can change over time, it is not possible to assign a static value to their distance. They proposed using an alphabet of low and high values to signal 0 and 1 respectfully. To allow the channel to be more robust, natural distance between the two hosts was reacquired as it changed to provide a channel with less noise.

Zander, Armitage, and Branch went on to evaluate IP covert channels in [7]. They observed three basic techniques: direct encoding, mapped encoding, and differential encoding. These techniques fall short due to the problem of not being able to add this covert channel to a preexisting carrier. Instead of a channel where the endpoints are the covert sender and receiver, a preexisting channel can be utilized and the covert operators sit in the middle and either actively alter traffic or passively listen for the channel passing by. Zander *et al* developed a method for transmitting a covert channel over IP TTL where the covert sender and receiver are not the overt sender and receiver. The first method they proposed assumes that the distance between the covert parties is already known. The sender generates a TTL related to the message, the receiver

then adds the known distance before decoding the message. They also proposed another scheme where the sender alters the TTL value, sending bits by repeating or changing the TTL value of subsequent packets.

Qu *et al* examined another TTL covert channel in the IP header protocol [8]. They used this carrier because it is common to see variance in the values it sends. This channel is also persistent from IPv4 to IPv6, even though the field name changes to Hop-Limit. In IP, the TTL field will decrement as it passes through each device on its path and remove itself from circulation when it hits zero. Since the hop distance between two hosts can change dynamically, care must be taken so the message doesnt deteriorate along its path. They devised a method utilizing Galois Fields to decrease the error rate in the channel.

III. TTL IN DNS

A. Background

DNS is used for translating qualified domain names to addresses that can be used for communicating to remote servers. The DNS request message consists of a list of one or more server names it wants to communicate with. The DNS server generates resource records for the questions, returning address information. The reply can also contain additional resource records with information the server thinks the client may want to know. Generally authoritative name servers for the domain are also provided.

Part of the DNS reply is a time to live (TTL) field attached to each resource record. This is used to communicate how long that DNS entry is valid for caching purposes. When the entry reaches its life span it is removed from the caching table. Before the lifespan is reached, the caching server can generate a reply as opposed to forwarding the request to a higher level.

B. Covert Channel

For this covert channel, communication will appear to take place normally. It will require a DNS server either be operated by the sender or a server that has been compromised by the sender. Clients require no special utilities to use the channel. When a client sends a request to the defined domain, the server will generate a valid DNS reply, answering all the questions. An analysis of the packet will show an answer for every question as to not raise suspicion of an observer or IDS system. It does not include additional answers to protect the channels presence.

For each question, a small portion of the message is inserted into the TTL field of the answer. To provide a higher level of covertness, messages can be encrypted instead of plain text. The proof of concept server and client are configured so that an encryption scheme can be placed into each of them easily to provide more security or covertness as needed.

A DNS request can contain more than one question statement and each question can have multiple resource records for clustered services such as the Google search engine. This can be leveraged to increase bandwidth by having more than one TTL field in the reply.

To examine TTL fields of large data sets, packet captures were obtained from the Cooperative Association for Internet Data Analysis(CAIDA) [9]. These packet captures were analyzed with Wireshark and scripting tools to examine the TTL fields. TTL values varied between a few minutes and a couple days. A large percentage of values were a discrete number of hours or days but values of minutes and seconds were not uncommon. Observing the DNS traffic, the average number of answer records, authority records, and additional records per response averaged a little above one with the high end of the range well above twenty, even reaching above 100. For testing, one question was used in the request and one answer resource record was returned along with one of each authoritative and additional resource records, a situation that would not look suspicious. Since some DNS servers strip authoritative and additional resource records, these TTLs will not be used for data transfer.

While testing, the TTL values were also examined to find their behavior. The data was graphed to observe the distribution. There was a constant distribution of all TTL values with the most prevalent spikes at 1 and 2 days. There was also more prevalent values in the low range. Statistical tests were also carried out. The mean TTL was found to be 98742 seconds, 27.4 hours, and a standard deviation of 81766, 22.7 hours. This gives a range of 4.7 to 50 hours that most traffic will fall into. Removing the 10 most used TTL values (5, 10, 15 minutes, 1, 2, 3, 4, 12 hours, and 1, 2 days), the mean and standard deviation become 38.44 hours and 42.86 hours respectively for a range of 0 to 81.30 hours. By removing the extreme outliers, more information can be gained about the behavior of traffic that isn't a highly common value. To be covert, an alphabet should be chosen that falls into this range to blend into the rest of the non-covert DNS traffic. The graph of TTL values with the top 10 removed is shown in Figure 1.

If the message is placed into the TTL field as the ASCII value, the packet capture will show the message. To increase the covertness of the channel, a layer of encryption can be used to mask the value. Many of the encryption schemes used for IP TTL field need to handle the decremental nature of the field. Since DNS TTL doesnt change between endpoints, except during caching, any reversible obfuscation algorithm can be used. This could incorporate encryption such as AES or it can be a simple addition based cipher. The client and server are configured to be able to replace the obfuscation algorithm.

Since the TTL field is 4 bytes long, it would be possible to encode 4 bytes directly into the TTL but this would create a TTL over 27 weeks which would be suspicious. Instead, one of the current implementation uses a cipher that takes the value of 2 ASCII bytes and multiplies it by 2 to generate larger TTL values. Calculation is shown in Equation 1

$$TTL = (byte_1 * 256 + byte_2) * 2. \quad (1)$$

This is sufficient to mask the hidden value from a packet capture. This scheme has a one standard deviation range of range of 13.04-17.02 hours so there is a high probability that

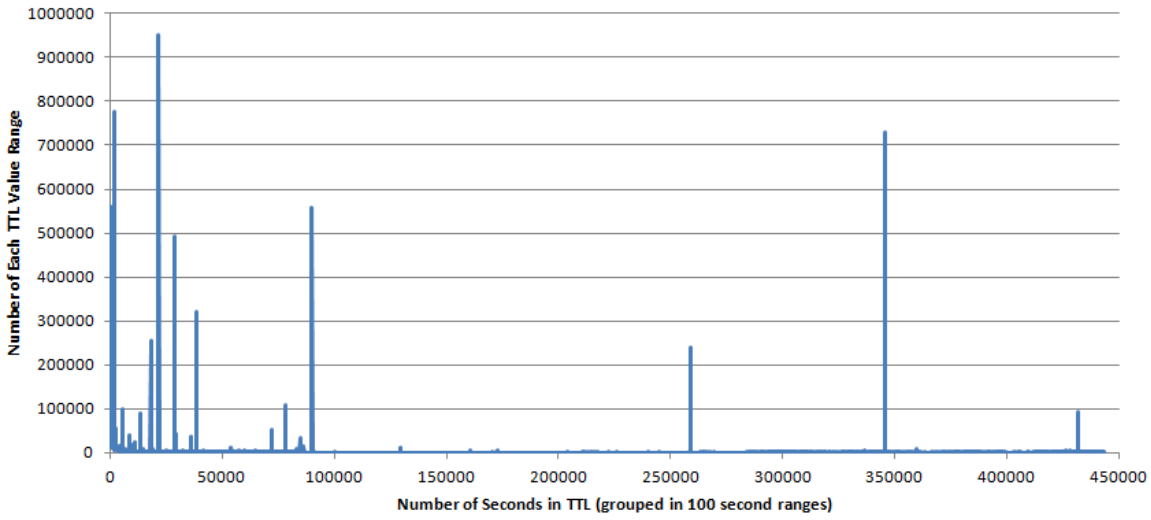

Fig. 1. Plotting the number of occurrences of TTL values from CAIDA datasets

the values will fall within the desired range. Packet captures of this scheme were taken and displayed in Figures 2 and 3 which together form the word "come".

```
z0.covertdns.com: type A, class IN, addr 23.23.183.221
  Name: z0.covertdns.com
  Type: A (Host address)
  Class: IN (0x0001)
  Time to live: 14 hours, 8 minutes, 30 seconds
  Data length: 4
  Addr: 23.23.183.221 (23.23.183.221)
```

Fig. 2. Packet capture of covert traffic with the message "co".

```
z1.covertdns.com: type A, class IN, addr 23.23.183.221
  Name: z1.covertdns.com
  Type: A (Host address)
  Class: IN (0x0001)
  Time to live: 15 hours, 33 minutes, 30 seconds
  Data length: 4
  Addr: 23.23.183.221 (23.23.183.221)
```

Fig. 3. Packet capture of covert traffic with the message "me".

To increase bandwidth, a text only scheme was implemented. A text only message can be sent using a mapping of a=0, b=1, etc with 26=EOF. The EOF character is used at the end of stream to notify the receiver that the message is complete. If the message is an odd length the EOF will appear directly after the message, otherwise it will be sent in the next TTL value. EOF also helps keep a larger TTL as to avoid suspicion as opposed to padding with null values. Using this scheme, a TTL of one day can transmit 3 characters per resource record. Four characters cause the TTL value to move towards a week which is a long time out if a caching server is in the middle. To increase the TTL to fit within the desired range, the value is multiplied by 5 before it is encoded into the packet. This calculation is shown in Equation 2.

$$TTL = (char_1 * 27^2 + char_2 * 27 + char_3) * 5 \quad (2)$$

This scheme has a one standard deviation range of 0.1-21.05 hours so it falls below the desired range but is acceptable due to the density of low values in the CAIDA dataset.

IV. Results and Evaluation

Based on a proof of concept server and client, the TTL field of DNS successfully carried a channel between two parties. The experiment was constructed using Amazon AWS to host the DNS server to have the traffic travel across the internet. The client was run on a local host using Wireshark to monitor the traffic between the two locations. To conduct testing through DNS servers, a domain was attached to the server. This ensured that the channel would not be broken during normal communication.

Testing this channel for bandwidth, the client was configured to query 100 new hostnames. Using Wireshark, the duration of all 100 queries were measured and used to find the number of queries per minute. 100 queries took 2.6 seconds which calculates to a bandwidth of 2305 packets per minute.

Using the alphabetic encoding scheme with one resource record, the capacity of the channel is 3 characters per packet. Averaging 2305 DNS interactions a minute, this gives a 6915 character/minute bandwidth. Using the byte encoding alphabet, 2 bytes can be transmitted per packet. This yields 4610 bytes/minute bandwidth.

Both of the proposed alphabets fall within the desired range gained from the statistical analysis of CAIDA data. If the covert traffic is viewed on its own it may look suspicious but with overt traffic it will be sufficiently masked.

Investigating rules from Snort [10], a popular IDS, rules do not exist for specifically limiting DNS traffic based on TTL. There exist a set of rules that do block a DNS packet with TTL as a parameter of the condition, it looks for a TTL less than or equal to one minute. However, it also has a condition that the reply is coming from a non-authoritative server. This rule is to alert to the presence of a DNS spoofing attack and so

the DNS covert channel would not trigger that alarm. To test, a snort IDS was configured on the network where it could see the covert channel traffic. The IDS did not detect the presence of the covert channel.

To detect the presence of this channel, a stateful system would need to be employed that would compare the TTL of similar resource records. If the values appeared to fluctuate in a non-uniform manner, there could be suspicion that covert communication is taking place. Such a system would generate a number of false negatives that it may be deemed unusable.

Bromberger alerts to the possibility of covert channels using DNS and gives an outline on how to protect against them [11]. A few of the characteristics that could indicate a DNS channel are lookups that are composed of hexadecimal strings, lookups with long 3rd and higher level names, multiple queries to non-obvious or foreign domains, responses with loopback addresses or other non-routable network addresses, queries to DDNS providers, and requests that are not followed by a requested connection to the requested addresses. As long as the parties communicating are within the same country and limit the number of interactions, the only characteristic my covert channel would fail is the client not requesting a connection to the requested addresses although this can be solved by having the client generate a connection to the addresses it received.

Once the channel is discovered, the channel can be linked to the sender. The receiver is safer from being detected in that the communication stream would need to be tracked through multiple DNS servers to get back to the receiver. Anonymity can be attempted by each party using the follow techniques. For the sender, a rogue DNS can be configured inside a network for sending information. Using spoofed IP and MAC addresses that keep changing, the server can avoid being located. However, the network could be configured as to only allow DNS replies from certain internal machines. The user could also obtain a public addressable DNS server under fraudulent credentials to avoid being tracked if the channel is found.

The receiver can also attempt to remain anonymous by spoofing their address on a request and passively listening for a reply. The caveat here is that the receiver might not see the return traffic if they are on a switched network that will send the reply directly to the address of the spoofed request. If the receiver is on a hubbed network and is able to see the traffic, it could only be increasing the chance of the channel being detected. If the spoofed address does not exist, it would be suspicious that communication is taking place with a non-existent entity. If the address does exist, a host-based IDS might trigger an alert that it received DNS traffic it did not request.

The channel is robust when the reply is from an authoritative server as opposed to a caching server. DNS traffic is necessary for network communication, even if only allowing DNS queries to an internal server. During normal traffic, DNS TTL does not get altered except during caching. When caching occurs, the TTL value gets decremented which will break the channel. During normal request/reply interaction, the TTL doesnt change.

V. Future Work

DNS uses caching along the path to reduce the amount of queries needed and speed up resolutions. When an answer is received from a caching server, the TTL value will be altered from the value given by the authoritative server. This change can break the channel since the time until it is requeried is unpredictable. Creating a channel that would be robust over caching would allow multiple clients to access the covert channel after a single client resolved the domain name. Although this would not be robust for the full life of the entry, it would allow some flexibility to who could access the channel.

This channel can also be used from an intermediate station [1], so instead of the channel endpoint generating traffic, the response can be encoded into existing traffic. This would require a router in the data flow stream to intercept and alter packets. This could use a legitimate DNS server and limit the linkability of the communicating parties.

VI. Conclusion

The TTL field of DNS was created to give a lifespan to a domain name, address pair. However, this value can be altered by the user to create a covert channel. This works well because the remainder of the packet is legitimate traffic and TTL does not have an expected value. This creates a robust channel as DNS traffic is largely unobstructed. At the current time, the channel is not robust enough to survive caching.

Acknowledgements

Support for the IPv4 Routed /24 Topology Dataset is provided by the National Science Foundation, the US Department of Homeland Security, the WIDE Project, Cisco Systems, and CAIDA Members.

References

[1] S. Zander, G. Armitage, and P. Branch, "A Survey of Covert Channels and Countermeasures in Computer Network Protocols," *Communications Surveys Tutorials, IEEE*, vol. 9, no. 3, pp. 44 –57, quarter 2007.

[2] D. Johnson, B. Yuan, and P. Lutz, "Behavior-Based Covert Channel in Cyberspace," *Intelligent Systems and Knowledge Engineering*, 2009.

[3] R. C. Newman, "Covert Computer and Network Communications," in *Proceedings of the 4th Annual Conference on Information Security Curriculum Development*, ser. InfoSecCD '07. New York, NY, USA: ACM, 2007, pp. 12:1–12:8.

[4] L. Nussbaum, P. Neyron, and O. Richard, "On Robust Covert Channels Inside DNS," in *Emerging Challenges for Security, Privacy and Trust*, ser. IFIP Advances in Information and Communication Technology, D. Gritzalis and J. Lopez, Eds. Springer Boston, 2009, vol. 297, pp. 51–62.

[5] S. Omar, I. Ahmedy, and M. Ngadi, "Indirect DNS Covert Channel Based on Name Reference for Minima Length Distribution," in *Information Technology and Multimedia (ICIM), 2011 International Conference on*, Nov. 2011, pp. 1 –6.

[6] S. Zander, G. Armitage, and P. Branch, "Covert Channels in the IP Time To Live Field," in *Australian Telecommunication Networks and Applications Conference*, Dec. 2006.

[7] S. Zander, G. Armitage, and Branch, "An Empirical Evaluation of IP Time To Live Covert Channels," in *Networks, 2007. ICON 2007. 15th IEEE International Conference on*, Nov. 2007, pp. 42 –47.

[8] H. Qu, P. Su, and D. Feng, "A Typical Noisy Covert Channel in the IP Protocol," in *Security Technology, 2004. 38th Annual 2004 International Carnahan Conference on*, Oct. 2004, pp. 189 – 192.

[9] Y. Hyun, B. Huffaker, E. Aben, and M. Luckie, "The CAIDA IPv4 Routed /24 Topology Dataset - 3/26/2012-3/27/2012,4/7/2012-4/8/2012," http://www.caida.org/data/active/ipv4_routed_24_topology_dataset.xml.

[10] Snort IDS, http://www.snort.org/.

[11] S. Bromberger, "DNS as a Covert Channel Within Protected Networks," in *National Electric Sectory Cybersecurity Organization*, 2011.