Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: November 9, 2018

L. Song
Beijing Internet Institute

May 8, 2018

ATR: Additional Truncation Response for Large DNS Response
draft-song-atr-large-resp-01

Abstract

As the increasing use of DNSSEC and IPv6, there are more public evidence and concerns on IPv6 fragmentation issues due to larger DNS payloads over IPv6. This memo introduces an simple improvement on DNS server by replying an additional truncated response just after the normal fragmented response. It can be used to relieve users suffering on DNS latency and failures due to large DNS response. It also can be utilized as a measuring and troubleshooting tool to locate the issue and conquer.

REMOVE BEFORE PUBLICATION: The source of the document with test script is currently placed at GitHub [ATR-Github]. Comments and pull request are welcome.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on November 9, 2018.

Table of Contents

1. Introduction 2
2. The ATR mechanism 4
3. Indicating a ATR response 5
4. Operational considerations 6
4.1. ATR timer 6
4.2. ATR payload size 7
4.3. Less aggresiveness of ATR 8
5. Security Considerations 9
6. IANA considerations 9
7. Acknowledgments 9
8. References 9
Appendix A. How well does ATR actually work? 11
Appendix B. Considerations on Resolver awareness of ATR 12
Appendix C. Revision history of this document 13
C.1. draft-song-atr-large-resp-01 13
Author's Address 14

1. Introduction

Large DNS response is identified as a issue for a long time. There is an inherent mechanism defined in [RFC1035] to handle large DNS response (larger than 512 octets) by indicating (set TrunCation bit) the resolver to fall back to query via TCP. Due to the fear of cost of TCP, EDNS(0) [RFC6891] was proposed which encourages server to response larger response instead of falling back to TCP. However, as the increasing use of DNSSEC and IPv6, there are more public evidence and concerns on user's suffering due to packets dropping caused by IPv6 fragmentation in DNS due to large DNS response.

It is observed that some IPv6 network devices like firewalls intentionally choose to drop the IPv6 packets with fragmentation Headers[I-D.taylor-v6ops-fragdrop]. [RFC7872] reported more than 30% drop rates for sending fragmented packets. Regarding IPv6

fragmentation issue due to larger DNS payloads in response, one measurement [IPv6-frag-DNS] reported 35% of endpoints using IPv6-capable DNS resolver can not receive a fragmented IPv6 response over UDP. Moreover, most of the underlying issues with fragments are unrevealed due to good redundancy and resilience of DNS. It is hard

for DNS client and server operators to trace and locate the issue when fragments are blocked or dropped. The noticeable DNS failures and latency experienced by end users are just the tip of the iceberg.

Depending on retry model, the resolver's failing to receive fragmented response may experience long latency or failure due to timeout and reties. One typical case is that the resolver finally got the answer after several retires and it falls back to TCP after deceasing the payload size in EDNS0. To avoid that issue, some authoritative servers may adopt a policy ignoring the UDP payload size in EDNS0 extension and always truncating the response when the response size is large than a expected one. However one study [Not-speak-TCP] shows that about 17% of resolvers in the samples can not ask a query in TCP when they receive truncated response. It seems a dilemma to choose hurting either the users who can not receive fragments or the users without TCP fallback capacity. There is also some voice of "moving all DNS over TCP". But It is generally desired that DNS can keep the efficiency and high performance by using DNS UDP in most of time and fallback as soon as possible to TCP if necessary for some corner case.

To relieve the problem, this memo introduces an small improvement on DNS responding process by replying an Additional Truncated Response (ATR) just after a normal large response which is to be fragmented. Generally speaking ATR provides a way to decouple the EDNS0 and TCP fallback in which they can work independently according to the server operator's requirement. One goal of ATR is to relieve the hurt of users, both stub and recursive resolver, from the position of server, both authoritative and recursive server. It does not require any changes on resolver and has a deploy-and-gain feature to encourage operators to implement it to benefit their resolvers.

Another goal of ATR is to help troubleshooting for DNS operators where ATR can be deployed as a measurement tool to identify vulnerable servers and users. A flag bit is required in EDNS0 OPT header to distinguish ATR response from a ordinary truncated

response. A resolver (or troubleshooter) can tell if there is any fragment not received in a certain transaction with a name server by receiving only ATR response without ordinary UDP response. Another way of using ATR as measurement tool is to reply ATR response to specific group of resolvers and record the TCP connections it received during that period. It can help identify vulnerable users and adopt ATR to them selectively.

[REMOVE BEFORE PUBLICATION] Note that in Appendix A of this memo there is a brief introduction of the test done by APNIC on how well does ATR actually work. And comments are also attached by the author

of this memo. It may help people to understand what the benefit and tradeoff that ATR brings.

2. The ATR mechanism

The ATR mechanism is very simple that it involves a ATR module in the responding process of current DNS implementation . As show in the following diagram the ATR module is right after truncation loop if the packet is not going to be fragmented.

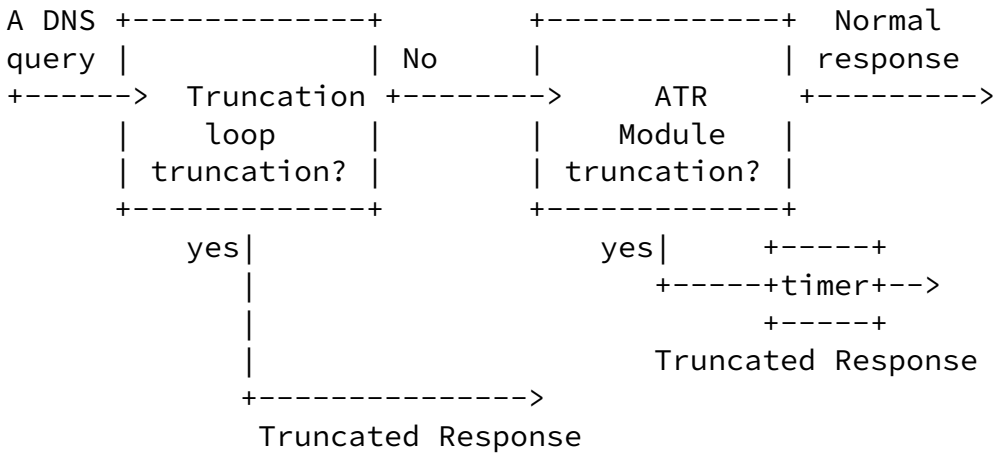

Figure 1: High-Level Testbed Components

The ATR responding process goes as follows:

- When an authoritative server receives a query and enters the responding process, it first go through the normal truncation loop to see whether the size of response surpasses the EDNS0 payload size. If yes, it ends up with responding a truncated packets. If no, it enters the ATR module.

- In ATR module, similar like truncation loop, the size of response is compared with a value called ATR payload size. If the response of a query is larger than ATR payload size, the server firstly sends the normal response and then coin a truncated response with the same ID of the query.

- The server can reply the coined truncated response in no time. But considering the possible impact of network reordering, it is suggested a timer to delay the second truncated response, for example 10~50 millisecond which can be configured by local operation.

Note that the choice of ATR payload size and timer SHOULD be configured locally. And the operational consideration and guidance is discussed in Section 4.2 and Section 4.1 respectively.

There are three typical cases of ATR-unaware resolver behavior when a resolver send query to an ATR server in which the server will generate a large response with fragments:

- Case 1: a resolver (or sub-resolver) will receive both the large response and a very small truncated response in sequence. It will happily accepts the first response and drop the second one because the transaction is over.

- Case 2: In case a fragment is dropped in the middle, the resolver will end up with only receiving the small truncated response. It will retry using TCP in no time.

- Case 3: For those (probably 30%*17% of them) who can not speak TCP and sitting behind a firewall stubbornly dropping fragments. Just say good luck to them!

In the case authoritative server truncated all response surpass

certain value , for example setting IPv6-edns-size to 1220 octets, ATR will helpful for resolver with TCP capacity, because the resolver still has a fair chance to receive the large response.

3. Indicating a ATR response

As introduced in ATR it is necessary to distinguish ATR response in a special way from a ordinary truncated response. It enables resolver operators to log cases where ATR responses is received without a (reassembled) UDP response to a query. Without an indicator that distinguishes ATR response, there would be no way to avoid false alarms from authoritative servers that always and only return truncated responses when the message exceeds some size. It is actually the use case where Google RDNS is considering. Google RDNS would like to use such indications to flag problematic name servers where RDNS should restrict maximum EDNS to a lower value than the default 4096 that currently used.

A simple way for that indicator of ATR response is to define a bit in the Z field on the EDNS0 OPT header in the response. This has the virtue of simplicity, and only a minimal risk of breaking existing implementations. This bit is referred to as the "ATR Response" (AT) bit. In the context of the EDNS0 OPT meta-RR, just following the DO bit, the AT bit is the second bit of the third and fourth bytes of the "extended RCODE and flags" portion (section 6.1.3 of RFC6891) of the EDNS0 OPT meta-RR, structured as follows:

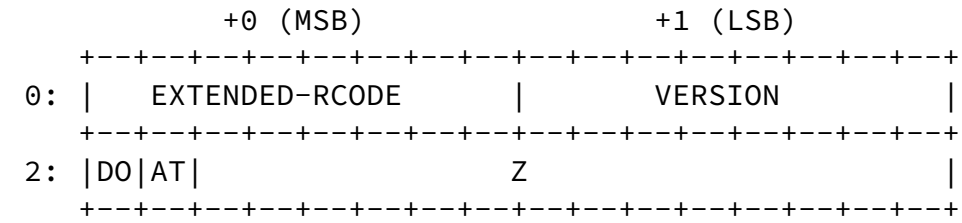

Figure 2: Wire format of extended RCODE and flags with AT bit

The logic of AT bit is simple that setting the AT bit to one in a response indicates to the resolver that the response is an ATR response. The AT bit cleared(set to zero) indicates the response is a ordinary response. Note that AT bit and TC bit SHOULD be set and

appear in the response as a pair. The response will be ignored if only AT bit is set.

The indication of ATR defined in this memo is for measurement and logging purpose. But it is possible used by resolver operator in other aspects, because it signals the resolver the large response is fragmented and dropped on the path. The resolver could act more actively if it is able to recognized that bit. More is discussed in Appendix B.

4. Operational considerations

In previous sections, only behavior of ATR server and AT bit are specified. There are lots of space for operational issues, such as the parameter of the ATR timer and ATR payload size, and policies on when ATR is triggered to avoid side-effect.

4.1. ATR timer

As introduced in Section 2 ATR timer is a way to avoid the impact of network reordering(RO). The value of the timer is critical, because if the delay is too short, the ATR response may be received earlier than the fragmented response (the first piece), the resolver will fall back to TCP bearing the cost which should have been avoided. If the delay is too long, the client may timeout and retry which negates the incremental benefit of ATR. Generally speaking, the delay of the timer should be "long enough, but not too long".

To the best knowledge of author, the nature of RO is characterized as follows hopefully helping ATR users understand RO and how to operate ATR appropriately in RO context.

- RO is mainly caused by the parallelism in Internet components and links other than network anomaly [Bennett]. It was observed that RO is highly related to the traffic load of Internet components. So RO will long exists as long as the traffic load continue increase and the parallelism is used to enhance network throughput.

- The probability of RO varies largely depending on the different

tests samples. Some work shown RO probability below 2% [Paxson] [Tinta] and another work was above 90% [Bennett]. But it is agreed that RO is site-dependent and path-dependent. It is observed in that when RO happens, it is mostly exhibited consistently in a small percentages of the paths. It is also observed that higher rates smaller packets were more prone to RO because the sending inter-spacing time was small.

- It was reported that the inter-arrival time of RO varies from a few milliseconds to multiple tens of milliseconds [Tinta]. And the larger the packet the larger the inter-arrival time, since larger packets will take longer to be transmitted.

Reasonably we can infer that firstly RO should be taken into account because it long exists due to middle Internet components which can not be avoided by end-to-end way. Secondly the mixture of larger and small packets in ATR case will increase the inter-arrival time of RO as well as the its probability. The good news is that the RO is highly site specific and path specific, and persistent which means the ATR operator is able to identify a few sites and paths, setup a tunable timer setting for them, or just put them into a blacklist without replying ATR response.

Based on the above analysis it is hard to provide a perfect value of ATR timer for all ATR users due to the diversity of networks. It seems OK to set the timer with a range from ten to hundreds ms, just below the timeout setting of typical resolver. Is suggested that a decision should be made as operator-specific according to the statistic of the RTT of their users. Some measurement shown [Brownlee][Liang] the mean of response time is below 50 ms for the sites with lots of anycast instance like L-root, .com and .net name servers. For that sites, delay less than 50 ms is appropriate.

4.2. ATR payload size

Regarding the operational choice for ATR payload size, there are some good input from APNIC study [scoring-dns-root]on how to react to large DNS payload for authoritative server. The difference in ATR is that ATR focuses on the second response after the ordinary response.

For IPv4 DNS server, it is suggested the study that do not truncate

and fragment IPv4 UDP response with a payload up to 1472 octets which is Ethernet MTU(1500) minus the sum of IPv4 header(20) and UDP header(8). The reason is to avoid gratuitously fragmenting outbound packets and TCP fallback at the source.

In the case of ATR, the first ordinary response is emitted without knowing it be to fragmented or not on the path. If a large value is set up to 1472 octets, payload size between 512 octets and the large value size will probably get fragmented by aggressive firewalls which leads losing the benefit of ATR. If ATR payload size set exactly 512 octets, in most of case ATR response and the single unfragmented packets are under a race at the risk of RO.

Given IPv4 fragmentation issue is not so serious compared to IPv6, it is suggested in this memo to set ATR payload size 1472 octets which means ATR only fit large DNS response larger than 1500 octets in IPv4.

For IPv6 DNS server, similar to IPv4, the APNIC study is suggested that do not truncate IPv6 UDP packets with a payload up to 1,452 octets which is Ethernet MTU(1500) minus the sum of IPv6 header(40) and UDP header(8). 1452 octets is chosen to avoid TCP fallback in the context that most TCP MSS in the root server is not set probably at that time.

In the case of ATR considering the second truncated response, a smaller size: 1232 octets, which is IPv6 MTU for most network devices（1280） minus the sum of IPv6 header(40) and UDP header(8), should be chosen as ATR payload size to trigger necessary TCP fallback. As a complementary requirement with ATR, the TCP MSS should be set 1220 octets to avoid Packet Too Big ICMP message as suggested in the APNIC study.

In short, it is recommended that in IPv4 ATR payload size SHOULD be 1472 octets, and in IPv6 the value SHOULD be 1232 octets.

4.3. Less aggresiveness of ATR

There is a concern ATR sends TC=1 response too aggressively especially in the beginning of adoption. One of the idea to mitigate this aggressiveness, ATR may respond TC=1 responses at a low possibility, such as 10%.

Another way to mitigating is to reply ATR response selectively. It is observed that RO and IPv6 fragmentation issues are path specific and persistent due to the Internet components and middle box. So it is reasonable to keep a ATR "whitelist" by counting the retries and

recording the IP destination address of that large response causing many retires. ATR only acts to those queries from the IP address in the white list.

5. Security Considerations

There may be concerns on DDoS attack problem due to the fact that the ATR introduces multiple responses from authoritative server. DNS cookies [RFC7873] and RRL on authoritative may be possible solutions

6. IANA considerations

EDNS(0) [RFC6891] defines 16 bits as extended flags in the OPT record, these bits are encoded into the TTL field of the OPT record. IETF Standards Action is required for assignments of new EDNS(0) flags.

This document reserves one of these bits as the AT bit. It is requested that the second bit (left most) be allocated. Thus the USE of the OPT record TTL field would look like:

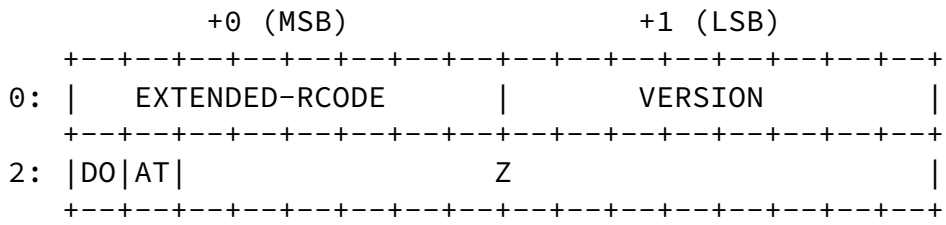

Figure 3: the USE of the OPT record TTL field for AT bit

7. Acknowledgments

Many thanks to reviewers and their comments. Geoff Huston and Joao Damas did a testing on the question "How well does ATR actually work?". Alexander Dupuy proposed the idea to distinguish ATR responses from normal ones. Akira Kato contributed ideas on operational consideration.

8. References

[ATR-Github]
"XML source file and test script of DNS ATR", September 2017, <https://github.com/songlinjian/DNS_ATR>.

[Bennett] "Packet Reordering is Not Pathological Network Behavior", December 1999, <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.461.7629&rep=rep1&type=pdf>.

[Brownlee] "Response time distributions for global name servers", 2002, <http://www.caida.org/publications/papers/2002/nsrtd/nsrtd.pdf>.

[How-ATR-Work] APNIC, "How well does ATR actually work?", April 2018, <https://blog.apnic.net/2018/04/16/how-well-does-atr-actually-work/>.

[I-D.taylor-v6ops-fragdrop] Jaeggli, J., Colitti, L., Kumari, W., Vyncke, E., Kaeo, M., and T. Taylor, "Why Operators Filter Fragments and What It Implies", draft-taylor-v6ops-fragdrop-02 (work in progress), December 2013.

[IPv6-frag-DNS] "Dealing with IPv6 fragmentation in the DNS", August 2017, <https://blog.apnic.net/2017/08/22/dealing-ipv6-fragmentation-dns>.

[Liang] Tsinghua University, "Measuring Query Latency of Top Level DNS Servers", February 2013, <https://netsec.ccert.edu.cn/duanhx/files/2013/02/latency.pdf>.

[Not-speak-TCP] "A Question of DNS Protocols", August 2013, <https://labs.ripe.net/Members/gih/a-question-of-dns-protocols>.

[Paxson] "End-to-End Internet Packet Dynamics", August 1999, <https://cseweb.ucsd.edu/classes/fa01/cse222/papers/paxson-e2e-packets-sigcomm97.pdf>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <https://www.rfc-editor.org/info/rfc6891>.

[RFC7872] Gont, F., Linkova, J., Chown, T., and W. Liu, "Observations on the Dropping of Packets with IPv6 Extension Headers in the Real World", RFC 7872, DOI 10.17487/RFC7872, June 2016, <https://www.rfc-editor.org/info/rfc7872>.

[RFC7873] Eastlake 3rd, D. and M. Andrews, "Domain Name System (DNS) Cookies", RFC 7873, DOI 10.17487/RFC7873, May 2016, <https://www.rfc-editor.org/info/rfc7873>.

[scoring-dns-root] APNIC, "Scoring the DNS Root Server System", November 2016, <https://blog.apnic.net/2016/11/15/scoring-dns-root-server-system/>.

[Tinta] "Characterizing End-to-End Packet Reordering with UDP Traffic", August 2009, <https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/35247.pdf>.

Appendix A. How well does ATR actually work?

It is worth of mentioning APNIC report[How-ATR-Work] on "How well does ATR actually work?" done by Geoff Huston and Joao Damas after 00 version of ATR draft. It was reported firstly in IEPG meeting before IETF 101 and then posted in APNIC Blog later.

It is said the test was performed over 55 million endpoints, using an online ad distribution network to deliver the test script across the Internet. The result is positive that ATR works! From the end users' perspective, in some 9% of IPv4 cases the use of ATR by the server will improve the speed of resolution of a fragmented UDP response by signaling to the client an immediate switch to TCP to

perform a re-query. The IPv6 behavior would improve the resolution times in 15% of cases.

It also analyzed the pros and cons of ATR. On one hand, It is said that ATR certainly looks attractive if the objective is to improve the speed of DNS resolution when passing large DNS responses. And ATR is incrementally deployable in favor of decision made by each server operator. On another hand, ATR also has some negative factors. One issue is adding another DNS DDoS attack vector due to the additional packet sent by ATR, (author's note : very small adding actually.) Another issue is risk of RO by the choice of the delay timer which is discussed fully in Section 4.1.

As a conclusion, it is said that "ATR does not completely fix the large response issue. If a resolver cannot receive fragmented UDP responses and cannot use TCP to perform DNS queries, then ATR is not

going to help. But where there are issues with IP fragment filtering, ATR can make the inevitable shift of the query to TCP a lot faster than it is today. But it does so at a cost of additional packets and additional DNS functionality". "If a faster DNS service is your highest priority, then ATR is worth considering", said at the end of this report

This test and report definitely made ATR more shiny attracting attention in the community. But it is found that there are still something unknown on "How well does ATR actually work". Firstly the test only counts the "success" case ("failure" otherwise) in which the "success" for large UDP case can be achieved by normal retries. The latency of that retries may reduced by ATR is not taken into account.

Secondly more analysis could be done in future to compare ATR with TCP. From the failure rate of users, we see ATR and TCP have similar performance (same for IPv4, and only 2% difference in IPv6). But there are resolvers who are able to receive the fragments more sooner in ATR case , but they fall back to TCP in TCP case. If not misunderstood, the two unknown parts underestimates the benefit of ATR in terms of speed of response.

The third unknown part is about the ATR timer and RO impact. In APNIC test, 10 ms was adopted as the delay of ATR timer according to

00 version of this draft. Different delay of ATR timer may not change the key result on gains of ATR (9% for IPv4 and 15% for IPv6). But the cost of RO is not measured. In the majority cases ATR is not needed, say 87% in IPv4, and 79% in IPv6. So it overestimate ATR in this regards if RO cost is not taken into account.

Appendix B. Considerations on Resolver awareness of ATR

ATR proposed in this memo is a server-side function which requires no change in resolver, so it is not required that resolver should recognized or use AT bit. But it may helpful for some special cases where a resolver is able to recognized or use AT bit.

One case is that when receiving a ATR response a ATR-aware resolver can adopt a "happyeyeballs" strategy by opening a separate transaction sending the query via TCP instead of falling back to TCP and closing the original UDP transaction. Listen to port 53 on both TCP and UDP port 53 will enhance the availability and reduce the latency. It will add more tolerance to network reordering issue as well. However, it should be taken into account about the balance of resolver's resource. Less priority should be given to that function when the resolver is "busy".

Another case is that a ATR-aware resolver is able to indicate its support for ATR to prudent servers who do not reply ATR response to every query and resolver. If that requirement is valid, it is possible for resolver to re-use the AT bit defined in this memo as a indication asking ATR response from the server.

However the two cases are currently outside of the scope of server-ATR specification. It needs further discussion.

Appendix C. Revision history of this document

C.1. draft-song-atr-large-resp-01

After receiving reviews and comments, changes of 01 version are shown as belows:

- Rewrite introduction and add another goal of ATR as a measuring tool;

- Add section 3 indicating a ATR response. An bit in the EDNS0 OPT header is defined as a indicator of ATR response. The flag bit is called "ATR Response" (AT) bit;

- Add Section 4 Operation considerations, which discuss ATR timer , ATR payload size, and less aggressiveness of ATR;

- Add IANA consideration to register the AT bit;

- Add section 7 Acknowledgments;

- Append a list of references regarding Network reordering, and APNIC's study on IPv6 and DNS;

- Add Appendix A, An introduce of APNIC testing work and author's comments;

- Appendix B. Considerations on Resolver awareness of ATR;

- Change the category="std" . It is said in RFC6891 IETF Standards Action is required for assignments of new EDNS(0) flags. So the draft should be categorized as standard track if registering AT bit is desired in this document.

Change history is also available in the public GitHub repository where this document is maintained: <https://github.com/songlinjian/DNS_ATR>.

Author's Address

Linjian Song
Beijing Internet Institute

Email: songlinjian@gmail.com
URI: http://www.biigroup.com/