DNS Extensions
INTERNET-DRAFT
Updates RFC 2845 (if approved)
Intended Status: Standards Track
Expires: January 8, 2014

H. Rafiee
M. v. Loewis
C. Meinel
Hasso Plattner Institute

July 8, 2013

Transaction SIGnature (TSIG) using CGA Algorithm in IPv6
draft-rafiee-intarea-cga-tsig-03.txt

Abstract

The first step in the Transaction SIGnature (TSIG) (RFC 2845) process is the generation of a shared secret to be used between a DNS server and a host. The second step consists of modifying the DNS configuration so that the DNS server will know what key to use with which host, because this shared secret is only valid between a pair of hosts. This document, CGA-TSIG, proposes a possible way to eliminate the human intervention needed for the generation and exchange of keys between a DNS server and a host when SEcure Neighbor Discovery (SEND) (RFC 3971) is used. CGA-TSIG will facilitate the authentication process of a host with a DNS server and will reduce the time needed to accomplish DNS Updates. It will also provide a means for securing the authentication process between resolvers and clients. CGA-TSIG will be added, as an extension, to TSIG in order to provide data integrity and proof of IP address ownership. The current signature generation and verification process used in TSIG will be substituted with the use of the same parameters as are used in generating a secure address in IPv6 networks, i.e., Cryptographically Generated Addresses (CGA) (RFC 3972).

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 http://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 January 8, 2014.

Table of Contents

1. Introduction

Transaction SIGnature (TSIG) [RFC2845] is a protocol that provides endpoint authentication and data integrity by the use of one-way hashing and shared secret keys in order to establish a trust relationship between two hosts which can be either a client and a server, or two servers. The TSIG keys, which are manually exchanged between these two hosts, need to be maintained in a secure manner. This protocol is today mostly used to secure a Dynamic Update, or to give assurance to the slave name server, that the zone transfer is from the original master name server and that it has not been spoofed by hackers. It does this by verifying the signature using a cryptographic key that is shared with the receiver.

It is possible to extend the TSIG protocol with the use of newly defined algorithms. This document proposes to use Cryptographically Generated Addresses (CGA) [RFC3972] as a new algorithm in the TSIG Resource Record (RR). CGA is an important option available in SEcure Neighbor Discovery (SEND) [RFC3971] which provides nodes with the necessary proof of IP address ownership by providing a cryptographic binding between a host and its IP address without the need for the introduction of a new infrastructure. CGA is a one-way hashing algorithm used to generate Interface IDs for IPv6 addresses in a secure manner. An interface ID consists of the rightmost 64 bits of the 128 bit IPv6 address. CGA verifies the ownership of the sender's IP address by finding a relationship between the sender's IP address and his public key [1,2].

Figure 1 IPv6 addresses

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying RFC 2119 significance.

3. Problem Statement

This document addresses the authentication problems associated with the need for hosts to change their IP addresses frequently in order to maintain privacy. This problem presents itself in three different scenarios: the authentication of a resolver with a client (stub), the authentication of two hosts (a client and a DNS server) during the DNS Update process, and the authentication of two DNS servers during the zone transfer. The focus of this document is on the first two problems, but it can also offer a possible solution to the third problem.

The DNS Update process is vulnerable to several types of spoofing attacks -- man in the middle, reflector , source IP spoofing, etc. TSIG secures this process by providing the transaction level authentication necessary by use of a shared secret. The current problem with using TSIG is the need for the manual processing that is required to generate and exchange the shared secrets. For each paired host there needs to be one shared secret and the administrator needs to manually add it to the DNS configuration file for each of these hosts. So, whenever these two hosts change their IP addresses, because of privacy issues as explained in RFC 4941 [RFC4941] or when moving to another subnet within the same network, this manual process will need to be invoked. The purpose of CGA-TSIG [7] is to minimize the amount of human intervention required to accomplish this exchange and, as a byproduct, to reduce the process's vulnerability to attacks introduced by human errors when SEcure Neighbor Discovery (SEND) is used for addressing purposes.

This same problem exists between a client and a DNS resolver. When a client sends a DNS query to a resolver, an attacker can send a response to this client containing the spoofed source IP address of this resolver. The client checks the resolver's source IP address for authentication. If the attacker spoofed the resolver's IP address, and if the attacker responds faster than the legitimate resolver, then the client's cache will be updated with the attacker's response. The client does not have any way to authenticate the resolver. In the above scenario, the resolver could add the TSIG Resource Record (RR) to the DNS query response and use the CGA-TSIG algorithm in order to permit a useful authentication of the result. CGA-TSIG assures the client that the query response comes from the true originator and not from an attacker. Currently there is little deployment of TSIG for resolver authentication with clients. One reason is that resolvers respond to anonymous queries and can be located in any part of the network. A second reason is that the manual TSIG process makes it difficult to configure each new client with the shared secret of the resolver.

There are several types of attack that CGA-TSIG can prevent. Here we will evaluate some of them. The use of CGA-TSIG will also reduce the number of messages needed in exchange between a client and a server

in order to establish a secure channel. To exchange the shared secret between a DNS resolver and a client, when TSIG is used, a minimum of four messages are required for the establishment of a secure channel. Modifying RFC 2845 to use CGA-TSIG will decrease the number of

messages needed in the exchange. The messages used in RFC 2930 (TKEY RR) are not needed when CGA-TSIG is used.

3.1. IP Spoofing

During the DNS Update process it is important that both communicating parties know that the one that they are communicating with is the actual owner of that IP address and that the messages are not being sent from a spoofed IP address. This can be accomplished by the use of the CGA algorithm which utilizes the node for IP address verification of other nodes.

3.2. DNS Dynamic Update Spoofing

Dynamic Update Spoofing is eliminated because the signature contains both the CGA parameters and the DNS update message. This will offer proof of the sender's IP address ownership (CGA parameters) and the validity of the update message.

3.3. Resolver Configuration Attack

When using CGA-TSIG, the DNS server, or the client, would not need further configuration. This would reduce the possibility of human errors being introduced into the DNS configuration file. Since this type of attack is predicated on human error, the chances of it occurring, when this extension is used, are minimized.

3.4. Exposing Shared Secret (key pairs)

In order to decrease the chances of attackers gaining unauthorized access to private keys on a node, it is recommended that key pairs be generated "on-the-fly".

3.5. Replay attack

Using the Time Signed value in the signature modifies the content of the signature each time the node generates and sends it to the DNS server. If the attacker tries to spoof this value with another timestamp, to show that the update message is current, the DNS server checks this message by verifying the signature. In this case, the verification process will fail thus also preventing the replay

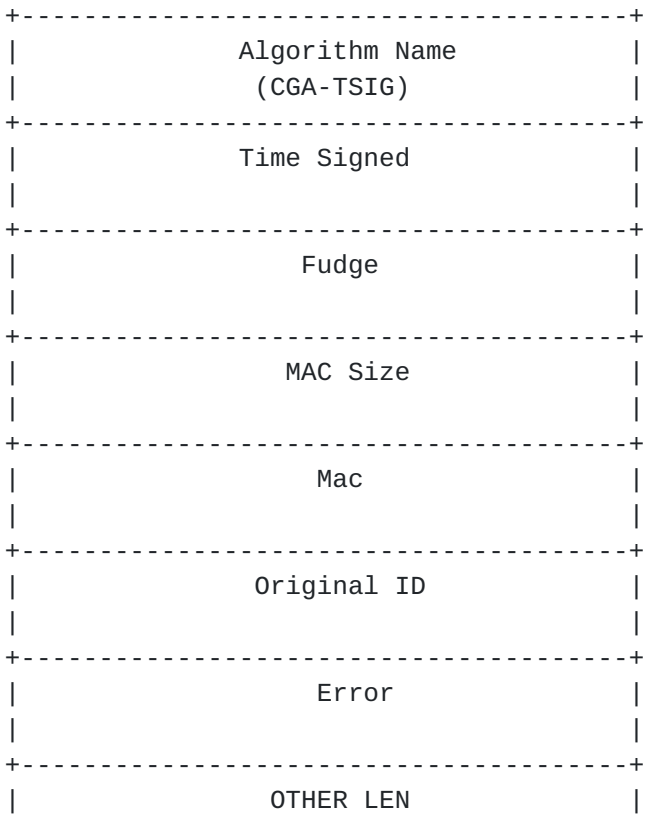
attack.

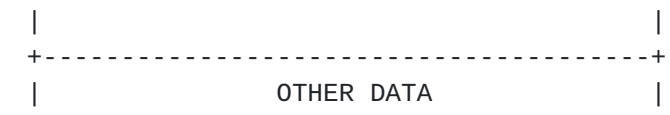
4. Algorithm Overview

The following sections explain the use of CGA for securing the DNS process by adding a CGA-TSIG data structure to the TSIG Resource Record (RR).

4.1. The CGA-TSIG DATA structure

The CGA-TSIG data structure SHOULD be added to the Other DATA section of the RDATA field in the TSIG Resource Record (RR) (see figures 2 and 3). The DNS RRTYPE must be set to TSIG [RFC2845]. The RDATA Algorithm Name MUST be set to CGA-TSIG. A detailed explanation of the standard RDATA fields can be found in section 2.3 RFC 2845. This document focuses only on the new structure added to the Other DATA section. These new fields are CGA-TSIG Len and CGA-TSIG DATA. The TSIG RR is added to an additional section of the DNS message. If another algorithm is used in place of CGA for SeND, such as SSAS [4 , 5], then the CGA-TSIG Len will be the length for the parameters of this algorithm and CGA-TSIG DATA will consist of the parameters required for verification of that algorithm, like signature, public key, etc.




OTHER DATA

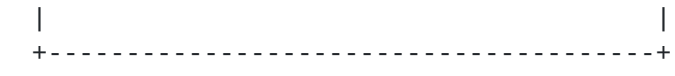


Figure 2 Modified TSIG RDATA

The CGA-TSIG DATA Field and the CGA-TSIG Len will occupy the first two slots of Other DATA. Figure 3 shows the layout.

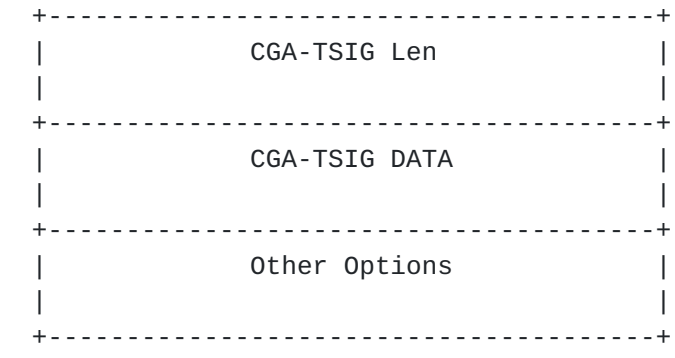


Figure 3 Other DATA section of RDATA field

CGA-TSIG DATA Field Name	Data Type	Notes
Algorithm type	u_int16_t	Name of the algorithm [RFC3972] RSA (by default) CGA
type	u_int16_t	Name of the algorithm used in SEND
IP tag	16 octet	the tag used to identify the IP address
Parameters Len	Octet	the length of CGA parameters
Parameters	variable	CGA parameters Section 3 RFC 3972
Signature Len	Octet	the length of CGA signature
Signature	variable	Section 3.2.1 This document
old pubkey Len	variable	the length of old public key field
old pubkey	variable	Old public key
old Signature Len	variable	the length of old signature field
old Signature	variable	Old signature generated by old public key.

Type indicates the Interface ID generation algorithm that was used in SeND. This field allows for the use of future, optional algorithms in SeND. The default value for CGA is 1. The IP tag is a node's old IP address. A client's public key can be associated with several IP addresses on a server. The DNS server, or the DNS message verifier node, SHOULD store the IP addresses and the public keys so as to indicate their association to each other. If a client wants to add

RRs to the server by using a new IP address, then the IP tag field will be set to binary zeros. The server will then store the new IP address that was passed to it in storage. If the client wants to

replace an existing IP address in a DNS server with a new one, then the IP tag field will be populated with the IP address which is to be replaced. The DNS server will then look for the IP address referenced by the IP tag stored in its storage and replace that IP address with the new one. This enables the client to update his own RRs using multiple IP addresses while, at the same time, giving him the ability to change IP addresses. If a node changes its public key in order to maintain privacy, then it MUST add the old public key to the old pubkey field. It MUST also retrieve the current time from Time Signed field, sign it using the old private key, and then add the digest (signature) to the old signature field. This enables the verifier node to authenticate a host with a new public key. The detailed verification steps are explained in sections 5.1, 6.1 and 7.1.

4.2. Generation of CGA-TSIG DATA

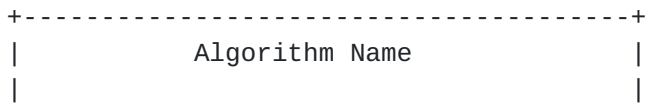
In order to use CGA-TSIG as an authentication approach, some of the parameters need to be cached during IP address generation. If no parameters are available in cache, please see section 8. If the Type (section 4.1) is CGA, then the parameters that SHOULD be cached are the modifier, algorithm type, location of the public/private keys and the IP addresses of this host generated by the use of CGA.

1.Obtain required parameters from cache.

The CGA-TSIG algorithm obtains the old IP address, modifier, subnet prefix, and public key from cache. It concatenates the old IP address with the CGA parameters, i.e., modifier, subnet prefix, public key and collision count (the order of CGA parameters are shown in section 3 RFC 3972). If the old IP address is not available, then CGA-TSIG must set the old IP address (IP tag) to zero.

2. Generate signature

For signature generation, all CGA parameters (modifier, public key, collision count and subnet prefix), that are concatenated with the DNS update message, the IP tag and the Time Signed field, are signed by using a RSA algorithm, the default, or any future algorithm used in place of RSA, and the private key which was obtained from cache in the first step. This signature must be added to the signature field of the CGA-TSIG DATA. Time Signed is the same timestamp as is used in RDATA. This value is the number of seconds since 1 January 1970 in UTC obtained from the signature generator. This approach will prevent replay attacks by changing the content of the signature each time a node wants to send a DNS message. The format of DNS messages is explained in section 4.1.2 RFC 1035 [RFC1035].


Algorithm Name

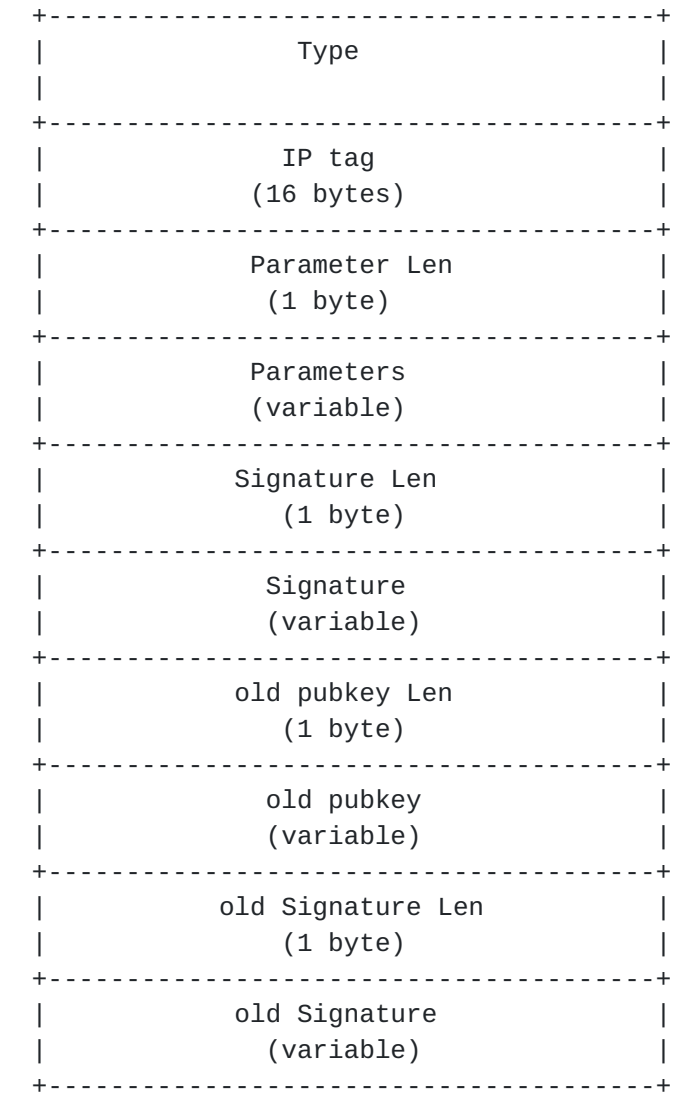


Figure 4 CGA-TSIG DATA Field

3. Generate old signature

If the nodes generated new key pairs, then they need to add the old public key and message, signed by the old private key, to CGA-TSIG DATA. A node will retrieve the timestamp from Time Signed, will use the old private key to sign it, and then will add the content of this signature to the old signature field of CGA-TSIG DATA. This step MUST be skipped when the node did not generate new key pairs.

5. DNS Update communication

This section discusses the use of CGA-TSIG for the authentication of a host in a DNS server. In this case, the messages sent from a host

will need to contain the CGA-TSIG option.

5.1. Verification process

Sender authentication is necessary in order to prevent attackers from making unauthorized modifications to DNS servers through the use of spoofed DNS messages. The verification process executes the following steps:

1. Execute the CGA verification

These steps are found in section 5 RFC 3972. If the sender of the DNS message uses another algorithm, instead of CGA, then this step becomes the verification step for that algorithm. If the verification process is successful, then step 2 will be executed. Otherwise the message will be discarded without further action.

2. Check the Time Signed

The Time Signed value is obtained from TSIG RDATA and is called t1. The current system time is then obtained and converted to UTC time and is called t2. If t1 is in the range of t2 and t2 minus x minutes (see formula 1, x minutes may vary according to transmission lag time) then step 3 will be executed. Otherwise, the message will be considered a spoofed message and the message should be discarded without further action. The range is used in consideration of the delays that can occur during its transmission over TCP or UDP. Both times must use UTC time in order to avoid differences in time based on different geographical locations.

t2-x <= t1 <= t2 (1)

3. Verify the signature

The signature contained in CGA-TSIG DATA should be verified. This can be done by retrieving the public key and signature from CGA-TSIG DATA and using this public key to verify the signature. If the verification process is successful, then step 4 will be executed. If the verification fails, then the message should be discarded without further action.

4. Verify the public key

The DNS server checks whether or not the public key retrieved from CGA-TSIG DATA is the same as what was available in the storage where the public keys and IP addresses were saved. If no entry is found for this public key in storage, then the DNS server adds this public key to its storage and processes the Update Message. If it is available, and it is the same as what is in storage, then the Update Message should be processed. Otherwise step 5 will be executed.

5. Verify the old public key

If the old public key length is zero, then skip this step and discard the DNS update message without further action. If the old public key

length is not zero, then the DNS server will retrieve the old public key from CGA-TSIG DATA and will check to see whether or not it is the same as what was saved in the DNS server's storage where the public keys and IP addresses are stored. If it is the same, then step 6 will be executed, otherwise the message should be discarded without further action.

6. Verify the old signature

The old signature contained in CGA-TSIG DATA should be verified. This can be done by retrieving the old public key and the old signature from CGA-TSIG DATA and then using this old public key to verify the old signature. If the verification is successful, then the Update Message should be processed and the new public key should be replaced with the old public key in the DNS server. If the verification process fails, then the message should be discarded without further action.

6. Stub to resolver communication

A DNS query request sent by a host, such as a client or a mail server, does not need to include CGA-TSIG DATA because the resolver responds to anonymous queries. But the resolver's response SHOULD contain the CGA-TSIG DATA field in order to enable this client to verify him.

In generation of the CGA-TSIG for a resolver, there is no need to include the IP tag. This is because resolvers don't usually have several IP addresses so the client does not need to keep several IP addresses for the same resolver.

6.1. Verification process

When a resolver responds to the host's query request for the first time, the client saves its public key in a file. This allows the client to verify this resolver when it changes its IP address due to privacy or security concerns. The first 2 steps of the verification process are the same as those steps explained in section 5.1 These steps are as follows:

1. Execute the CGA verification

2. Check the Time Signed

3. Verify the Source IP address

If the resolver's source IP address is the same as that which is known for the host, then step 4 will be executed. Otherwise the message SHOULD be discarded without further action.

4. Verify the signature

The signature contained in CGA-TSIG DATA should be verified. This can be done by retrieving the public key and signature from CGA-TSIG DATA and using this public key to verify the signature. If the verification process is successful, then step 5 will be executed. If the verification fails, then the message should be discarded without further action.

5. Verify the public key

The host checks whether or not the public key retrieved from CGA-TSIG DATA matches any public key that was previously saved in the storage where the public keys and IP addresses of resolvers are saved. If there is a match, then the message is processed. If not, then step 5 will be executed.

5. Verify the old public key

If the old public key length is zero, then skip this step and discard the DNS query response without further action. If the old public key length is not zero, then the host will retrieve the old public key from CGA-TSIG DATA and will check whether or not it is the same as what was saved in the host's storage where the public keys and IP addresses are stored. If it is the same, then step 6 will be executed, otherwise the message should be discarded without further action.

6. Verify the old signature

The old signature contained in CGA-TSIG DATA should be verified. This can be done by retrieving the old public key and old signature from CGA-TSIG DATA and then using this old public key to verify the old signature. If the verification is successful, then the DNS Message should be processed and the new public key should be replaced with the old public key of the resolver in the host. If the verification process fails, then the message should be discarded without further action.

7. DNS server to DNS server communication (zone transfer)

In the case of processing a DNS update for multiple DNS servers (authentication of two DNS servers), there are two possible scenarios with regard to the authentication process, which differs from that of the authentication of a node (client) with one DNS server. This is because of the need for human intervention.

a. Add the DNS servers' IP address to a slave configuration file

A DNS server administrator should only manually add the IP address of

the master DNS server to the configuration file of the slave DNS server. When the DNS update message is processed, the slave DNS server can authenticate the master DNS server based on the source IP address and then, prove the ownership of this address by use of the CGA-TSIG option from the TSIG RR. This scenario will be valid until the IP address in any of these DNS servers changes.

To automate this step's process, the DNS Update message sender's public key must be saved on the other DNS server, after the source IP address has been successfully verified for the first time. In this case, when the sender generates a new IP address by executing the CGA algorithm using the same public key, the other DNS server can still verify it and add its new IP address to the DNS configuration file automatically.

b. Retrieve public/private keys from a third party Trusted Authority (TA)

The message exchange option of SEND [RFC3971] may be used for the retrieval of the third party certificate. This may be done automatically from the TA by using the Certificate Path Solicitation and the Certificate Path Advertisement messages. Like in scenario b, the certificate should be saved on the DNS server for later use for the generation of its address or for the DNS update process. In this case, whenever any of these servers want to generate a new IP address, then the DNS update process can be accomplished automatically without the need for human intervention.

7.1. Verification Process

The verification steps are the same as those is explained in section 5.1, but with one additional step.

1- Execute the CGA verification

2- Check the Time Signed

3- Verify the signature

4- Verify the source IP address

The source IP address of the Update requester MUST be checked against the one contained in the DNS configuration file. If it is the same, then the Update Message should be processed, otherwise, step 5 will be executed.

5- Verify the public key

6- Verify the old public key

7- Verify the old signature

8. No cache parameters available

In a case where there are no cache parameters available during the IP address generation, the sender of DNS message needs to generate a key pair and generate the CGA-TSIG data structure as explained in section 4. The node SHOULD skip the first section of verification processes explained in section 5.1 , section 6.1 and section 7.1.

9. Security Considerations

The approach explained in this draft, CGA-TSIG, is a solution for securing DNS messages from spoofing type attacks like those explained in section 3.

A problem that may arise here concerns attacks against the CGA algorithm. In this section we will explain the possibility of such attacks against CGA [5] and explain the available solutions that we considered in this draft.

a) Discover an Alternative Key Pair Hashing of the Victim's Node Address

In this case an attacker would have to find an alternate key pair hashing of the victim's address. The probability for success of this type of attack will rely on the security properties of the underlying hash function, i.e., an attacker will need to break the second pre-image resistance of that hash function. The attacker will perform a second pre-image attack on a specific address in order to match other CGA parameters using Hash1 and Hash2. The cost of doing this is (2^59+1) * 2^(16*1). If the user uses a sufficient security level, it will be not feasible for an attacker to carry out this type of attack due to the cost involved. Changing the IP address frequently will also decrease the chance for this type of attack succeeding.

b) DoS to Kill a CGA Node

Sending a valid or invalid CGA signed message with high frequency across the network can keep the destination node(s) busy with the

verification process. This type of DoS attack is not specific to CGA, but it can be applied to any request-response protocol. One possible solution ,to mitigate this attack, is to add a controller to the verifier side of the process to determine how many messages a node has received over a certain period of time from a specific node. If a determined threshold rate is exceeded, then the node will stop further receipt of incoming messages from that node.

c) CGA Privacy Implication

Due to the high computational complexity necessary for the creation of a CGA, it is likely that once a node generates an acceptable CGA it will continue its use at that subnet. The result is that nodes using CGAs are still susceptible to privacy related attacks. One solution to these types of attacks is setting a lifetime for the address as explained in RFC 4941.

10. IANA Considerations

The IANA has allowed for choosing new algorithm(s) for use in the TSIG Algorithm name. Algorithm name refers to the algorithm described in this document. The requirement to have this name registered with IANA is specified.

In section 4.1, Type should allow for the use of future optional algorithms with regard to SeND. The default value for CGA might be 1. Other algorithms would be assigned a new number sequentially. For example, a new algorithm called SSAS [4,5] could be assigned a value of 2.

11. Appendix

- A sample key storage for CGA-TSIG

```
create table cgatsigkeys (

id           INT auto_increment,

pubkey       VARCHAR(300),

primary key(id)

);
```

```
create table cgatsigips (

id           INT auto_increment,
```

```
idkey               INT,

IP          VARCHAR(20),

FOREIGN KEY (idkey) REFERENCES cgatsigkeys(id)

primary key(id)

);
```

CGA-TSIG tables on mysql backend database

- a sample format of stored parameters in the node

For example, the modifier is stored as bytes and each byte might be separated by a comma (for example : 284,25,14,...). Algorithmtype is the algorithm used in signing the message. Zero is the default algorithm for RSA. Secval is the CGA Sec value that is, by default, one. GIP is the global IP address of this node (for example: 2001:abc:def:1234:567:89a). oGIP is the old IP address of this node, before the generation of the new IP address. Keys contains the path where the CGA-TSIG algorithm can find the PEM format used for the public/private keys (for example: /home/myuser/keys.pem).

```
<?xml version="1.0" encoding="UTF-8"?>
<Details>
<CGATSIG>
   <modifier value=""/>
   <algorithmtype value="0"/>
   <secval value="1"/>
   <GIP value=""/>
   <oGIP value=""/>
    <Keys value=""/>
</CGATSIG>
</Details>
```

XML file contains the cached DATA

12. Acknowledgements

The author would like to thank all those people who directly helped in improving this draft and all supporters of this draft, especially Ralph Droms, Andrew Sullivan and Brian Haberman.

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)," RFC 3972, March 2005.

[RFC3971] Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, March 2005.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2930] Eastlake 3rd, D., "Secret Key Establishment for DNS (TKEY RR)", RFC 2930, September 2000.

[RFC1035] Mockapetris, P., "Domain Names - Implementation And Specification", RFC 1035, November 1987.

[RFC4941] Narten, T., Draves, R., Krishnan, S., "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, September 2007.

[RFC2136] Vixie, P. (Editor), Thomson, S., Rekhter, Y., Bound, J., "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.

13.2. Informative References

[RFC2845] Vixie, P., Gudmundsson, O. , Eastlake 3rd, D., Wellington, B., " Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, May 2000.

13.3. Informative References

[1] Aura, T., "Cryptographically Generated Addresses (CGA)", Lecture Notes in Computer Science, Springer, vol. 2851/2003, pp. 29-43, 2003.

[2] Montenegro, G. and Castelluccia, C., "Statistically Unique
and Cryptographically Verifiable (SUCV) Identifiers and
Addresses," ISOC Symposium on Network and Distributed System

Security (NDSS 2002), the Internet Society, 2002.

[3] AlSa'deh, A., Rafiee, H., Meinel, C., "IPv6 Stateless Address Autoconfiguration: Balancing Between Security, Privacy and Usability". Lecture Notes in Computer Science, Springer(5th International Symposium on Foundations & Practice of Security (FPS). October 25 - 26, 2012 Montreal, QC, Canada), 2012.

[4] Rafiee, H., Meinel, C., "A Simple Secure Addressing Generation Scheme for IPv6 AutoConfiguration (SSAS)". Work in progress, http://tools.ietf.org/html/draft-rafiee-6man-ssas, 2013.

[5] Rafiee, H., Meinel, C., "A Simple Secure Addressing Scheme for IPv6 AutoConfiguration (SSAS)", 11th International conference on Privacy, Security and Trust (IEEE PST), 2013.

[6] AlSa'deh, A., Rafiee, H., Meinel, C., "Cryptographically Generated Addresses (CGAs): Possible Attacks and Proposed Mitigation Approaches," in proceedings of 12th IEEE International Conference on Computer and Information Technology (IEEE CIT'12), pp.332-339, 2012.

[7] Rafiee, H., Meinel, C., "A Secure, Flexible Framework for DNS Authentication in IPv6 Autoconfiguration" in proceedings of The 12th IEEE International Symposium on Network Computing and Applications (IEEE NCA13), 2013.

Authors' Addresses

Hosnieh Rafiee
Hasso-Plattner-Institute
Prof.-Dr.-Helmert-Str. 2-3
Potsdam, Germany
Phone: +49 (0)331-5509-546
Email: ietf@rozanak.com

Dr. Christoph Meinel
(Professor)
Hasso-Plattner-Institute
Prof.-Dr.-Helmert-Str. 2-3
Potsdam, Germany
Email: meinel@hpi.uni-potsdam.de

Dr. Martin von Loewis
(Professor)
Hasso-Plattner-Institute
Prof.-Dr.-Helmert-Str. 2-3
Potsdam, Germany
Email: martin@v.loewis.de