CGA as alternative security credentials with IKEv2: implementation and analysis

Jean-Michel Combes
France Telecom - Orange, Orange Labs
38 rue du General Leclerc
92130 Issy-Les-Moulineaux, France
Phone: +33 1 45 29 45 94
Fax: +33 1 45 29 65 19
Email: jeanmichel.combes@orange.com

Aurelien Wailly
France Telecom - Orange, Orange Labs
38 rue du General Leclerc
92130 Issy-Les-Moulineaux, France
Phone: +33 1 45 29 85 10
Fax: +33 1 45 29 65 19
Email: aurelien.wailly@orange.com

Maryline Laurent
Institut TELECOM, TELECOM SudParis
CNRS Samovar UMR 5157
9 rue Charles Fourier
91011 Evry, France
Phone: +33 1 60 76 44 42
Fax: +33 1 60 76 47 11
Email: Maryline.Laurent@it-sudparis.eu

***Abstract*—*Internet Protocol security* (IPsec) is a protocol suite enabling secure IP communications by authentication and/or encryption. *Internet Key Exchange version 2* (IKEv2) mechanism is recommended to configure dynamically IPsec between IP nodes and the authentication of each peer is usually based on either pre-shared keys, X.509 certificates or *Extensible Authentication Protocol* (EAP). However, these methods may have drawbacks. On the other hand, *Cryptographically Generated Addresses* (CGA), IPv6 addresses with specific security properties, are the main component of the mechanism to secure the IPv6 Neighbor Discovery protocol but these security properties are only used in a local scope. An interesting solution could be the use of CGA as alternative security material for IKEv2.**

In this paper, we analyze advantages and drawbacks of CGA use compared to classical IKEv2 security materials, decide design choices regarding modifications of IKEv2 to integrate CGA, and finally, describe the resulting implementation.

***Index Terms*—Security, IPv6, IPsec/IKEv2, Credentials, CGA**

I. Introduction

Today, *Internet Protocol security* (**IPsec**) is widely used to set-up secure flows through the Internet. Indeed, this protocol allows to bring authentication and/or encryption to any IP communication. To establish dynamically IPsec connections between peers, *Internet Key Exchange version 2* (**IKEv2**) mechanism is recommended. This mechanism may use three types of security credential for the peer authentication: pre-shared keys, X.509 certificates or *Extensible Authentication Protocol* (**EAP**). Each of these types of security credential has drawbacks from a deployment point of view. On the other hand, *Cryptographically Generated Addresses* (**CGA**) are IPv6 addresses with security properties. This type of addresses is limited to a local range, as the main component of the mechanism to secure the IPv6 Neighbor Discovery protocol. Extending the usage of CGA and its security properties to a larger range could be interesting and especially as alternative security credential for IKEv2.

Our article is organized as follows. First, IPsec and IKEv2 are introduced. Then, CGA and the advantages/limitations of combining CGA with IKEv2 are presented. Finally, the integration of CGA in IKEv2 and the resulting implementation are described.

II. Internet Protocol security

Internet Protocol security (**IPsec**) [KS05] is a protocol suite enabling secure IP communications by authentication and/or encryption. IPsec, recommended for IPv6 [JLN11], is widely deployed today for IPv4. Even if, IPsec may be manually configured on IP nodes, it is strongly recommended to use *Internet Key Exchange version 2* (**IKEv2**) [KHNE10] to make deployment easier.

A. IPsec

The two main protocols used by IPsec are the *IP Authentication Header* (**AH**) [Ken05a] and the *IP Encapsulating Security Payload* (**ESP**) [Ken05b]. AH and ESP provide data origin authentication and data integrity but ESP provides also data confidentiality. Two modes of use are specified for these protocols: transport and tunnel. In transport mode, the IPsec policy is applied directly over the IP packet. In tunnel mode, the IP packet is encapsulated inside another IP packet on which is applied the IPsec policy.

The IPsec policy, stored in the *Security Policy Database* (**SPD**), is based on rules specifying the processing for a specific IP packet: for example, for each IP packet from peer A to peer B when port 80 is used, apply IPsec/ESP/transport mode. Peer identifier is generally either an IP address or a *Fully Qualified DNS Name* (**FQDN**) (e.g., foo.example.org) but other types of identifier also exist. FQDN is generally preferred because an IP address is not always suited. At first, an IP address may be only temporary allocated to the peer and so the SPD should be modified each time the peer has a new IP address. Another point, IPv6 addresses are not easy to manipulate because they are longer than IPv4 addresses and hexadecimal encoded .

An IPsec *Security Association* (**SA**), also called Child SA [KHNE10], describes how is applied an IPsec rule from the SPD (e.g., from peer A to peer B, apply ESP, transport mode, 3DES-CBC [PA98], the associated key for the encryption, etc.) and is unidirectional, meaning that two SA are required to protect communication between two peers. All SA are stored in the *Security Association Database* (**SAD**). A SA may be configured manually or dynamically (e.g., with IKEv2

[KHNE10]). This last option allows AH and ESP to provide anti-replay protection.

The last IPsec database is the *Peer Authorization Database* (**PAD**) and is used by protocols establishing dynamically IPsec SA, such as IKEv2. The PAD stores information about allowed authentication method, associated authentication data, type and value of the identifier for each IPsec peer (e.g., foo.example.org/pre-shared key/"secret").

B. IKEv2

The *Internet Key Exchange version 2* (**IKEv2**) [KHNE10] is a protocol that dynamically negotiates and updates IPsec SAs (also called Child SA). This mechanism obsoletes the first version, IKE [HC98].

Each communication between IKEv2 peers, the *Initiator* and the *Responder*, is based on a request and a response, called IKEv2 exchange. As illustrated in figure 1, the first IKEv2 exchange is *IKE_SA_INIT* and the second one *IKE_AUTH*. The following ones, optional, are *CREATE_CHILD_SA* and *INFORMATIONAL*.

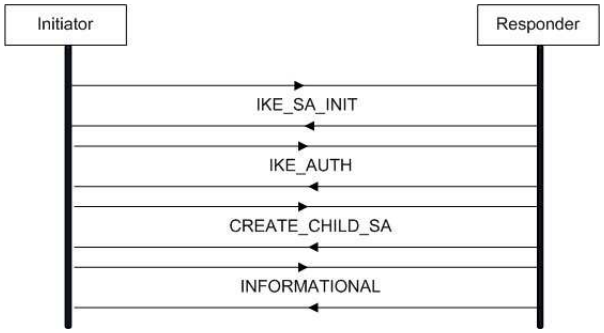


Fig. 1. IKEv2 exchanges

The different IKEv2 exchanges are:

- IKE_SA_INIT, where security parameters, nonces, Diffie-Hellman values are exchanged and the IKE SA, providing data integrity and encryption for the next exchanges, is negotiated and established;
- IKE_AUTH, where identities are exchanged, each peer is authenticated and the first Child SA is negotiated and established;
- CREATE_CHILD, when a new Child SA is created;
- INFORMATIONAL, when a SA is deleted, an error notified or a link tested.

1) IKE_SA_INIT: This exchange initializes trust parameters, negotiates and establishes the IKE SA between peers.

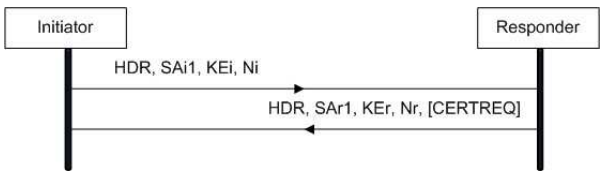


Fig. 2. IKE_SA_INIT exchange

Different information elements are exchanged during this exchange, as illustrated in figure 2. The *Security Parameter Index* (**SPI**), in the HDR payload, identifies the security association for the IKE SA. The SAi1 payload includes proposals from the Initiator for the IKE SA that will secure next IKEv2 exchanges and the SAr1 payload the IKE SA selected by the Responder. The KEi and KEr payloads include respectively the Diffie-Hellman value of the Initiator and the Responder. Identically, the Nonce from the Initiator and the Responder are inside the Ni and Nr payloads.

Optionally, the Responder may request a certificate from the Initiator with the CERTREQ payload.

At the end of this exchange, both peers can generate a key from the exchanged nonces and the Diffie-Hellman result, called *SKEYSEED*, from which all future keys used by IKEv2 will be derived. The cipher one is named *SK_e*, the integrity one is *SK_a* and finally the child one is *SK_d*.

2) IKE_AUTH: During this exchange, protected by the IKE SA, each peer is authenticated and the first Child SA is negotiated and established.

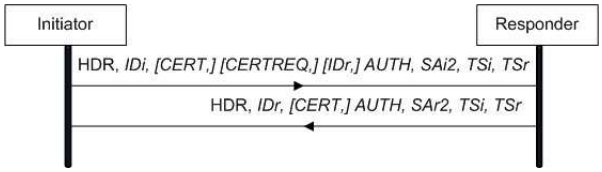


Fig. 3. IKE_AUTH exchange

The IKE_AUTH exchanged information elements are illustrated in figure 3. The *Security Parameter Index* (**SPI**), in the HDR payload, identifies the security association for the Child SA. The SAi2 payload includes the proposal from the Initiator for the Child SA and the SAr2 payload the Child SA selected by the Responder. The AUTH payload contains data used for the sender authentication and the message integrity. The Initiator's and the Responder's identifiers are respectively inside the IDi and IDr payloads.

Optionally, the Initiator may request a certificate from the Responder with the CERTREQ payload. Certificates exchanged between peers are included in CERT payloads.

Payloads in italic, in the figure 3, are encrypted using SK_e. Integrity is provided using SK_a. A peer proves its identity, stored in IDi for the Initiatior and IDr for the Responder, with data located in the payload AUTH.

3) Authentication in IKEv2: Authentication is usually based on one of the following security materials: pre-shared keys, X.509 certificates [CSF+08] and *Extensible Authentication Protocol* (**EAP**) [ABV+04]. It is important to notice that EAP support is not mandatory in IKEv2.

However, these methods may have drawbacks. Indeed, provision of pre-shared key is complex and generally not scalable. X.509 certificates may require to deploy a dedicated architecture, like a *Public Key Infrastructure* (**PKI**), which may be heavy to manage. Finally, EAP support is not mandatory in

IKEv2 specifications and so interoperability issues may appear between peers.

By the way, two other authentication methods exist for IKEv2 but, like EAP, are not mandatory in IKEv2 implementations. The first one is to store its public key in a *Domain Name Server* (**DNS**) [Ric05]. But this would require the deployment of *Domain Name Server security* (**DNSSEC**) [AAL+05] to guarantee the validity of this public key. The other "authentication" method is known as *Better-Than-Nothing Security* (**BTNS**) [WR08]: the assumption is that there is no malicious node trying to do a man-in-the-middle attack during IKE_SA_INIT and so no authentication is required during IKE_AUTH ...

An alternative solution could be the use of *Cryptographically Generated Addresses* (**CGA**) as security material.

III. Cryptographically Generated Addresses

An IPv6 address is the concatenation of two 64-bits parts where the first part is the network prefix and the second one is the *Interface Identifier* (**IID**). The IID is generally derived from the MAC address [Cra98] but other methods exist to generate it [NDK07]. *Cryptographically Generated Addresses* (**CGA**) [Aur] are IPv6 addresses where the IID is the hash computation over the concatenation of a public key and specific parameters. Such a type of addresses is an element of the mechanism to secure the Neighbor Discovery protocol [AKZN05], where the CGA security proprieties are only used in a local range (i.e. on the link where the CGA owner is located).

To generate a CGA, an IPv6 node needs first a RSA public/private key pair [RSA78]. After performing the standardized algorithm [Aur05], the IPv6 node should get an IID, associated with the key pair, which results from the first 64 bits of the SHA-1 hash function [iost02] applied over the data structure called *CGA Parameters*, as illustrated in figure 4.

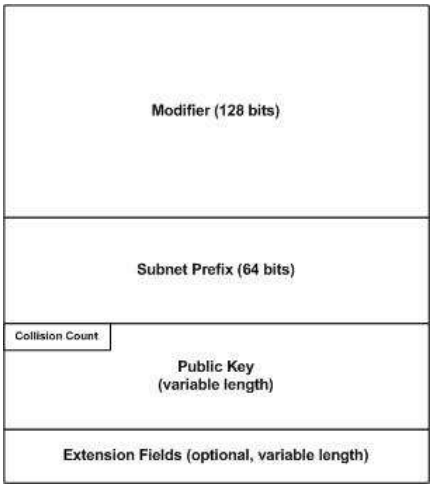


Fig. 4. CGA Parameters

To verify the ownership of a CGA, an IPv6 node needs first getting the associated CGA Parameters and a bunch of fresh data signed with the private key related to this CGA. The IPv6 node checks that it can regenerate the same CGA, following the standardized algorithm [Aur05], and if so, then it checks the validity of the signature to confirm the node using the CGA is the real owner of the public key related to this address.

IV. Advantages and limitations of using CGA with IKEv2

Using CGA with IKEv2 brings advantages and limitations that we describe in this section. For each limitation, we give potential solutions to solve or mitigate the vulnerability.

A. Infrastructureless approach

The use of CGA doesn't need any infrastructure because the CGA is generated by its owner and all the security material needed to check the CGA, and the associated message, is sent directly to the message receiver. In comparison, self-signed certificates, which only require a public/private keys pair and no infrastructure like CGA, can be generated by its owner but it is impossible to check whether the information within the certificate, especially the identifier (e.g. the IP address) if included, are correct or not. As such, CGA are more secure as the attacker should break hash function and find a collision to get the same level of security. This is far more difficult than masquerading.

Moreover, because the whole verification is done by the message receiver, a CGA doesn't need a certification chain unlike X.509 certificates built on a *Public Key Infrastructure* (**PKI**). A PKI includes many entities like a *Certificate Authority* (**CA**) and a *Registration Authority* (**RA**) - in some deployment architecture, these two entities can be co-located. These entities and the messages exchanges between them can introduce potential security holes. So, if any of them has been correctly exploited, the whole trust chain is broken and the consequences are the same as with self-signed certificates.

B. Identification

CGA, as an IP address, is hard for a human to be remembered (especially an IPv6 address which is longer than an IPv4 address and hexadecimal encoded). Moreover, a message receiver doesn't know naturally who is the message sender behind the CGA. CGA may be used also for anonymity (i.e., no link between the layer 3 address and the layer 2 address) and so could change frequently. A first logical idea would be to associate a CGA to a *Fully Qualified Domain Name* (**FQDN**) stored in a *Domain Name Server* (**DNS**). However, this would introduce classical security threats linked to the DNS exchanges like DNS Cache poisoning [AA04] and so the security level provided by CGA would be lost.

To solve this issue, we need DNS updates exchanges and DNS resolution operations to be secured. DNS updates are used when a CGA owner registers its CGA with its associated FQDN in a DNS. The DNS resolutions are used when an IPv6 node needs to know the FQDN attached to a CGA. Note that securing DNS update exchanges is possible with TSIG

[VGrW00] and SIG(0) [3rd00]. DNS resolutions can be secured by deploying *Domain Name Server security* (**DNSSEC**) [AAL+05] over DNS. With such protection tools, a potential attacker has no other option than finding a collision.

C. "Hard-coded" cryptographic algorithms

CGA standard [Aur05] only allows the use of two cryptographic algorithms: the public-key algorithm RSA and the hash function SHA-1.

Regarding RSA, this cryptographic algorithm is known to be expensive from a computing ressource and time point of view (i.e. CPU and battery power constraints). Thus, RSA is not well adapted for constrained nodes like mobiles or sensors. But, present CGA standard forbid the use of alternative algorithms like Elliptic Curve based algorithms even if papers [CBL10] have been published and standard proposals [SXZ11] submitted to the IETF about such a topic.

Regarding SHA-1, based on recent cryptanalysis, the security community expects this algorithm should be broken in a near future and so collisions should be generated easily. The security community is working on a next generation hash function: SHA-3. But again, CGA standard forbid the use of alternative algorithms for the moment.

By the way, it is important to mention that the use of cryptographic algorithms can be constrained in some countries or companies: for example, in Russia, the algorithm GOST [Dol10] must be used instead of any other hash function.

D. Revocation

A CGA provides a strong relationship between an address and its owner but nothing prevents a malicious host being able to use it in the future when the CGA has been compromised by a collision. Unlike X.509 certificates built on a PKI where either a *Certificate Revocation List* (**CRL**) [CSF+08] or *Online Certificate Status Protocol* (**OCSP**) [MT07] can be deployed, a CGA is infrastructureless and so cannot be revoked.

A potential solution to consider could be the use of DNS again (and DNSSEC to keep the same security level as described before). Indeed, a value of the *Time To Live* (**TTL**), in the AAAA record for the FQDN associated to the CGA, shorter than the expected time to find a collision for this CGA, could limit the collision attack. Now, the proposed solution would become useless if an attacker tries to pre-compute collisions for a large number of CGA.

V. Integration of CGA into IKEv2

A method to use CGA with IKEv2 was firstly described in an academic paper [CMLN04] where the goal was to perform opportunistic encryption between security gateways. Following this work, a proposal [LMK07], specifying now how to use CGA for any IPsec scenario, has been submitted to the IETF. For our work, we decided to do design choices regarding modifications of IKEv2 to integrate CGA based on this last document. Choices are explained below.

A. Modifications of IKEv2 payloads

At first, the ID payloads must contain the peer's identity that will be authenticated with the AUTH payload: this is the peer's CGA as ID_IPV6_ADDR format.

IKEv2 specifications describe the CERT payload may contain any type of material allowing authentication. Here, we decide the structure of CGA parameters is included and encoded in a format that looks like a self-signed certificate. From the implementation point of view, a new CERT type value has been assigned, which is 222.

The CERTREQ payload should contain the same CERT type as previously in order to specify that the sender requests the receiver's CGA parameters.

Finally, the AUTH payload contains different data for the authentication of the CGA owner: a digital signature based on the private key associated to the public key from the CGA parameters located in the CERT payload.

B. Modifications of IKEv2 exchanges

During the IKE_SA_INIT exchange, the only change concerns the Responder's reply. This one must include a CERTREQ payload with the new type 222.

The IKE_AUTH exchange is now built with the following information:

- *Security Parameter Index* (SPI) in the HDR payload: It identifies the security association for the Child SA,
- IDi payload: Initiator's CGA,
- IDr payload: Responder's CGA,
- CERT payload: CGA parameters,
- CERTREQ payload: CGA Type (=222),
- IDr payload: Responder's identity,
- AUTH payload: Signature (based on the private key associated with the public key in the CERT Payload).

VI. IKEv2 with CGA implementation

The integration of CGA in IKEv2 is based on two existing implementations:

- CGA implementation
  The CGA library in the Secure Neighbor Discovery implementation developed by DoCoMo USA Labs was selected because this is one of the few available public working implementations. Now, this implementation is no longer maintained.
- IKEv2 implementation
  StrongSwan[1] was selected because this IKEv2 implementation provides a well-coded and maintained development framework.

To implement directly our design choices in StrongSwan, many modifications have been done. At first, the IPsec configuration file and the associated parser have been modified to accept CGA based rules. This CGA must be now in the ID payload during IKEv2 exchanges. Next, to request and provide CGA Paramaters, inside respectively CERTREQ and CERT payloads, the new value "222" for Certificate Encoding

[1] http://www.strongswan.org/

has been specified. These CGA Parameters are encoded in a X.509 certificate template [CSF+08] as illustrated in figure 5. The default parser inside StrongSwan can dynamically include functions via plugins (i.e. *dlopen* system function). So, to be able to parse the CGA Parameters inside this certificate and to transform them into a valid format for StrongSwan, a new plugin was implemented. Regarding the AUTH payload process, now, before checking the validity of the signature, the IKEv2 daemon must verify the validity of the CGA (i.e. the IKEv2 daemon re-generates the CGA using the provided CGA parameters as specified in the CGA specifications [Aur05]).

Finally, to be able to check the information were correctly provided during IKEv2 exchanges, a plugin for Wireshark[2] was implemented.

VII. Conclusion and future works

In IKEv2 standard, pre-shared keys and X.509 certificates are the only mandatory credentials. These ones are not easy to deploy and manage. EAP is only supported by some commercial products, resulting in interoperability issues.

In this paper, we identified the advantages and limitations of using CGA with IKEv2 as alternative credential. Finally, we described a possible integration of CGA in IKEv2 and our implementation.

Our work through this publication could be used for getting feedback from the IETF and push the IETF proposal [LMK07] to standardization.

References

[3rd00] D. Eastlake 3rd. DNS Request and Transaction Signatures (SIG(0)s). RFC 2931, Internet Engineering Task Force, September 2000.

[AA04] D. Atkins and R. Austein. Threat Analysis of the Domain Name System (DNS). RFC 3833, Internet Engineering Task Force, August 2004.

[AAL+05] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. DNS Security Introduction and Requirements. RFC 4033, Internet Engineering Task Force, March 2005.

[ABV+04] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz. Extensible Authentication Protocol (EAP). RFC 3748, Internet Engineering Task Force, June 2004.

[AKZN05] J. Arkko, J. Kempf, B. Zill, and P. Nikander. SEcure Neighbor Discovery (SEND). RFC 3971, Internet Engineering Task Force, March 2005.

[Aur] Tuomas Aura. Cryptographically generated addresses (cga). In *Information Security*, volume 2851 of *Lecture Notes in Computer Science*, pages 29–43. Springer Berlin / Heidelberg.

[Aur05] T. Aura. Cryptographically Generated Addresses (CGA). RFC 3972, Internet Engineering Task Force, March 2005.

[BAN90] Michael Burrows, Martin Abadi, and Roger Needham. A logic of authentication. *ACM Trans. Comput. Syst.*, 8:18–36, February 1990.

[CBL10] Tony Cheneau, Aymen Boudguiga, and Maryline Laurent. Significantly improved performances of the cryptographically generated addresses thanks to ecc and gpgpu. *Computers & Security*, 29(4):419 – 431, 2010.

[CMLN04] Claude Castelluccia, Gabriel Montenegro, Julien Laganier, and Christoph Neumann. Hindering eavesdropping via ipv6 opportunistic encryption. In *in Proceedings of the European Symposium on Research in Computer Security, Lecture Notes in Computer Science*, pages 309–321. Springer-Verlag, 2004.

[2]http://www.wireshark.org/

[Cra98] M. Crawford. Transmission of IPv6 Packets over Ethernet Networks. RFC 2464, Internet Engineering Task Force, December 1998.

[CSF+08] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280, Internet Engineering Task Force, May 2008.

[Dol10] V. Dolmatov. GOST R 34.11-94: Hash Function Algorithm. RFC 5831, Internet Engineering Task Force, March 2010.

[HC98] D. Harkins and D. Carrel. The Internet Key Exchange (IKE). RFC 2409, Internet Engineering Task Force, November 1998.

[iost02] National institute of standards and technology. FIPS 180-2, Secure Hash Standard, Federal Information Processing Standard (FIPS), Publication 180-2. Technical report, DEPARTMENT OF COMMERCE, August 2002.

[JLN11] E. Jankiewicz, J. Loughney, and T. Narten. IPv6 Node Requirements. RFC 6434, Internet Engineering Task Force, December 2011.

[Ken05a] S. Kent. IP Authentication Header. RFC 4302, Internet Engineering Task Force, December 2005.

[Ken05b] S. Kent. IP Encapsulating Security Payload (ESP). RFC 4303, Internet Engineering Task Force, December 2005.

[KHNE10] C. Kaufman, P. Hoffman, Y. Nir, and P. Eronen. Internet Key Exchange Protocol Version 2 (IKEv2). RFC 5996, Internet Engineering Task Force, September 2010.

[KS05] S. Kent and K. Seo. Security Architecture for the Internet Protocol. RFC 4301, Internet Engineering Task Force, December 2005.

[LMK07] J. Laganier, G. Montenegro, and A. Kukec. Using IKE with IPv6 Cryptographically Generated Addresses. Internet-Draft draft-laganier-ike-ipv6-cga-02, Internet Engineering Task Force, July 2007. Obsolete.

[MT07] M. Myers and H. Tschofenig. Online Certificate Status Protocol (OCSP) Extensions to IKEv2. RFC 4806, Internet Engineering Task Force, February 2007.

[NDK07] T. Narten, R. Draves, and S. Krishnan. Privacy Extensions for Stateless Address Autoconfiguration in IPv6. RFC 4941, Internet Engineering Task Force, September 2007.

[PA98] R. Pereira and R. Adams. The ESP CBC-Mode Cipher Algorithms. RFC 2451, Internet Engineering Task Force, November 1998.

[Ric05] M. Richardson. A Method for Storing IPsec Keying Material in DNS. RFC 4025, Internet Engineering Task Force, March 2005.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21:120–126, February 1978.

[SXZ11] B. Sarikaya, F. Xia, and G. Zaverucha. Lightweight Secure Neighbor Discovery for Low-power and LossyNetworks. Internet-Draft draft-sarikaya-6lowpan-cgand-01, Internet Engineering Task Force, May 2011. Work in progress.

[VGrW00] P. Vixie, O. Gudmundsson, D. Eastlake 3rd, and B. Wellington. Secret Key Transaction Authentication for DNS (TSIG). RFC 2845, Internet Engineering Task Force, May 2000.

[WR08] N. Williams and M. Richardson. Better-Than-Nothing Security: An Unauthenticated Mode of IPsec. RFC 5386, Internet Engineering Task Force, November 2008.

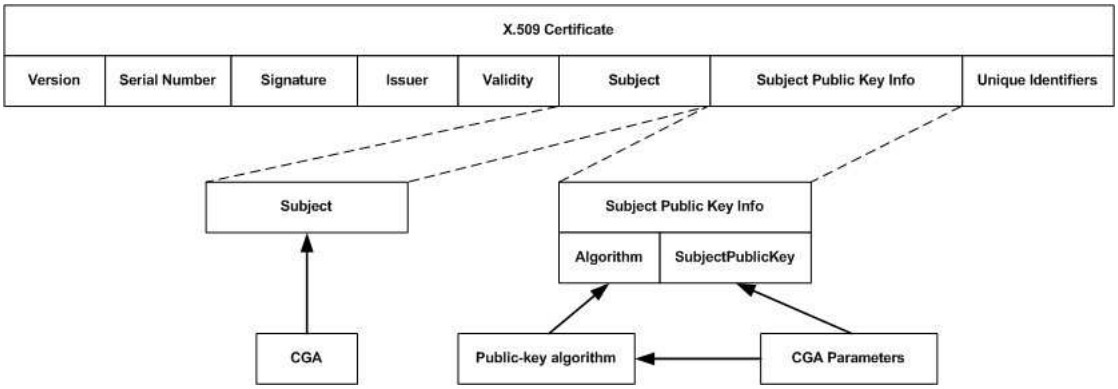


Fig. 5. CGA Parameters included in a X.509 certificate template