Secure Signaling in Next Generation Networks with NSIS

Roland Bless and Martin Röhricht
Institute of Telematics
Universität Karlsruhe (TH)
Zirkel 2, D–76128 Karlsruhe, Germany
Email: {bless, roehricht}@tm.uka.de

***Abstract*—The IETF working group Next Steps in Signaling (NSIS) develops signaling protocols for Quality-of-Service (QoS) reservations or dynamic NAT and firewall (NAT/FW) configuration. QoS signaling allows for on-demand resource reservations in order to provide guaranteed quality-of-service for real-time oriented services in IP-based next generation networks whereas NAT/FW signaling allows for establishing pinholes in firewalls or bindings in NAT devices. QoS signaling must be secured to allow for a reliable accounting and NAT/FW configuration is a sensitive operation per se. This paper presents an approach that provides an integrity protection of NSLP signaling messages by extending an NSLP Session Authorization Object. A worked example for secure QoS signaling in a Kerberos-secured domain is given.**

I. Introduction

Signaling protocols for IP resource control constitute an important component for next generation networks. For instance, interactive real-time services like Voice-over-IP usually require some stronger Quality-of-Service (QoS) guarantees than today's Internet can provide. For such interactive media streams a minimum throughput and an upper bound for end-to-end delay should be guaranteed by the network. Hard guarantees require resource reservation and some kind of *admission control* within a service class in order to maintain guarantees for already admitted users. As an example the Expedited Forwarding class of the Differentiated Services Architecture [1] can provide strong guarantees for interactive real-time services, but it requires admission control to work properly. Such *on-demand admission control* requires a *QoS signaling* solution.

The IETF working group Next Steps in Signaling (NSIS) was established to address the shortcomings [2], [3] of the already existing Resource Reservation Protocol (RSVP) [4]. Some of the deficiencies of RSVP include lacking support for mobility, the constitution of only a single-layer protocol, limitation to end-to-end reservations, or that reservations could only be performed from the receiver-side. The NSIS working group started to define a new suite of Internet signaling protocols with Quality-of-Service signaling as a first use case. As a second use case NAT/FW configuration was considered, that allows for establishing pinholes in firewalls or bindings in NAT devices.

In both cases secure signaling is an important requirement: Services that require QoS support must be accounted differently from normal services, otherwise every user would simply request always the best QoS, leading to a best-effort scenario again. Thus, a reliable and secure accounting is an essential feature in the context of QoS signaling. Consequently, this means that every resource reservation request must be authenticated in order to allow for a secure and reliable accounting. Likewise every firewall configuration action must be authenticated properly to avoid unauthorized access.

Currently missing, however, is a possibility to authenticate NSIS signaling messages on a per session or per user basis. As discussed in Section III, there exist several proposals that convey authorization tokens within signaling messages, but they do not correlate the signaling message itself with the authorization information.

We propose to use an NSIS-based signaling solution as part of an overarching QoS control mechanism for the ScaleNet architecture [5], which constitutes a Next Generation Network approach. This paper intends to extend an already existing proposal to provide authenticated and integrity protected NSIS signaling. The latter is important, because only an initial authentication for a session offers many opportunities for attackers to gain unauthorized access to resources. In order to allow for securing every signaling message the chosen approach is lightweight, because it uses keyed hashing to "sign" each message and is therefore computationally inexpensive.

The paper is structured as follows: first we give a brief overview of the NSIS architecture and protocols in Section II. We describe related work in Section III, before our approach is presented in Section IV. In Section V an example using Kerberos is shown before the paper is concluded in Section VI.

II. Signaling with NSIS

The NSIS protocol suite has a two-layer protocol structure, in order to fulfill the requirements for extensible, generic signaling [6]. Its conceptual architecture is illustrated in Figure 1. The lower layer is termed NSIS Transport Layer Protocol (NTLP) and is mainly responsible for transporting any kind of NSIS signaling. The upper layer protocol is called NSIS Signaling Layer Protocol (NSLP) and depends on the particular signaling application, like resource reservation or NAT and firewall configuration for instance. The following sections mainly discuss QoS NSLP as an example, but the proposed mechanisms will work for other NSLPs as well.

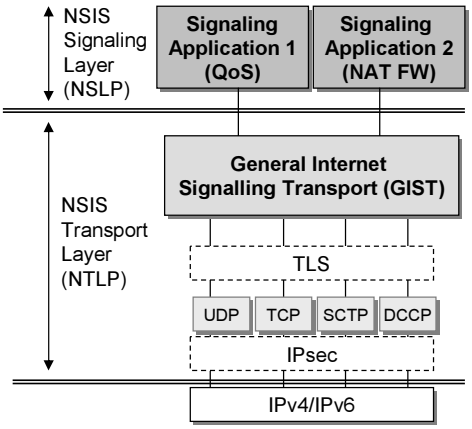

Figure 1. Layered architecture of the NSIS protocols

The Quality-of-Service NSIS Signaling Layer Protocol (QoS NSLP) [7] was defined by the NSIS working group as one part of a larger suite of signaling protocols that are supported by the NSIS framework [8]. It works on top of the NTLP layer which is responsible for hop-by-hop based signaling message delivery. QoS NSLP is independent of a specific QoS model such as IntServ or DiffServ and is mainly used to establish and maintain state along the nodes of a path to guarantee resource reservations within IP networks. It supports both, sender or receiver-initiated reservations, or even reservations between arbitrary nodes. QoS NSLP is used as a signaling protocol only, so that Resource Management Function related information must be carried in a separate object, called QSPEC.

The General Internet Signalling Transport (GIST) [9] was defined as a protocol implementation of the NTLP layer to provide a common protocol stack for transporting and routing signaling messages through the network. GIST uses message routing to determine the next GIST peer along the signaling path and builds therefore a transport abstraction for the upper NSLP layer. GIST was designed to use already existing transport protocols, such as UDP, TCP, or SCTP, as well as underlying security mechanisms, such as Transport Layer Security (TLS) or IPsec. These secured connections are however only based on a hop-by-hop node level, i. e., they can not be used for a secured end-to-end path as being established by the signaling applications itself. In order to manage critical resources like QoS reservations or firewall pinholes GIST does not offer appropriate mechanisms to verify a given host's identity (Authentication), check whether the host is authorized to make use of a particular service (Authorization), and finally account a host's credit once it actually uses this service (Accounting).

Security within NSIS is therefore limited to already existing channel security mechanisms like TLS or IPsec as provided by the NTLP layer. However, TLS at GIST level between nodes is not appropriate, since this protection is not related to sessions and the user level. That is, security of NSIS is insufficient for secure accounting or authorization of resource reservation.

III. Related Work

GIST's security mechanisms are primarily concerned on a secure delivery of signaling messages between adjacent peers, whereas the upper layer needs different security properties with a stronger authorization functionality [10]. Therefore QoS signaling protocols should provide capabilities for authentication and authorization of QoS requests. Tsenov et al. [11] provided an approach for authentication and authorization that can be used by NSIS, by integrating the Extensible Authentication Protocol (EAP) into QoS-NSLP. The authors defined a new EAP policy element that carries EAP packets within exchanged QoS-NSLP messages. However, although this approach integrates smoothly into the existing NSLP message exchange procedure, it requires all NSLP nodes to be aware of this policy element. Further it does not integrity protect specific NSLP objects like QSPEC, but rather focuses on providing an authentication mechanism only and accomplishes a cryptographic channel binding by tying EAP with GIST's TLS. In [12] a special authorization application is defined and public key cryptography is used for authentication. Compared to the mechanisms proposed in this paper, both approaches are more heavy weight, because they use backend communication to policy servers as well as public cryptography for verification of the authenticity of requests. Furthermore, NSLP message content is not included into such authenticity checks. In [13] an NTLP-based hop-by-hop and per session protection based on a temporary public/private key pair (Diffie Hellman exchange) was proposed. This approach is less scalable due to its session orientation and signatures based on public-key cryptography.

In order to provide a policy-based per-session authorization and admission control it is necessary to define a Session Authorization Policy Element [14] that supports the exchange of this particular information. A resource reservation message must hold information that may be used by a host to verify the validity of a reservation request. An individually submitted Draft [15] was proposed that defines a new NSLP layer object, called AUTH_SESSION that should be used to allow for a finer-grained admission control.

The AUTH_SESSION object relies on the provision of authorization tokens by a trusted third party. These tokens can be used by clients to sign their requests and prove authenticity. Network elements are then able to verify the requests and process the resource reservation message according to admission policies. It is possible to add more than one AUTH_SESSION object in one single message in order to provide end-to-end signaling along a path of different networks. This allows network elements to use the identifiers of authorizing entities that belong to third parties they trust.

The Session Authorization Object format—as depicted in Figure 2—contains a generic NSLP object header followed by a list of Session Authorization Attributes that describe the session and can be used to authorize the resource reservation request. The bits A and B are extensibility flags and specify the desired treatment for objects yet unknown to the receiver.

They specify mandatory objects or objects that can be ignored, or whether they should be forwarded.

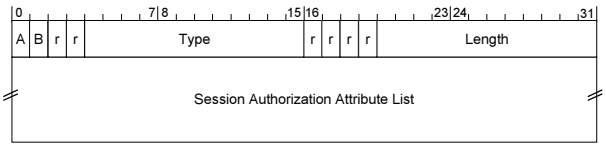

Figure 2. Session Authorization Object Format

A Session Authorization Attribute is composed of four different fields, namely the length, its type and subtype, and the attribute specific information. Figure 3 illustrates the generic format of an attribute. Attributes that are currently defined by the draft are the Authorizing Entity Identifier, such as a Fully Qualified Domain Name or a digital certificate, source and destination address of the authorized session, start and end time of the authorized session, and finally the authentication data which signs all the data in the policy element.

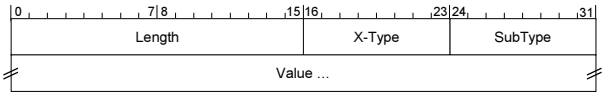

Figure 3. Format of a Session Authorization Attribute

The draft refers to different use case scenarios that can be applied to ensure the integrity of AUTH_SESSION policy elements. They are namely: *Shared symmetric keys*, *Kerberos*, *Public key*.

The shared symmetric key's operational setting relies on shared symmetric keys between the Authorizing Entity and the network router/PEP. In this case the first 32 bit of the authentication data field serve as key ID to identify the corresponding signing key. The key context further holds the lifetime parameters of its validity as well as a cryptographic algorithm identifier which indicates the algorithm that is to be used with this key.

Kerberos authentication mechanisms are currently not defined by the draft [7]. Public key authorization may be applied based on X.509 or PGP certificates. An authorizing entity, e. g., a AAA server, uses its private key to generate the authentication data. Its public key can be used by authenticators like network routers to verify the validity of the policy element.

An authorizing entity generates the AUTH_SESSION policy element according to a framework for session set-up and resource reservations for media flows [16]. In order to prevent replay attacks either a start time or a session ID must be included into the element. The AUTH_SESSION object can be used with QUERY, RESERVE, and even RESPONSE messages. The latter may apply if the authorizing entity changed the original request, e. g. by authorizing only parts of the requested resource reservations or by changing session time attributes.

As soon as an NSIS Initiator receives a policy element from the authorizing entity, it must be copied without modification into the AUTH_SESSION object. The AUTH_SESSION object itself is then part of an NSLP message.

Upon reception of an NSLP message by a QoS NSLP Entity (QNE) that is policy aware, the QNE extracts the AUTH_SESSION object from the message and contacts the Policy Decision Point (PDP) by using the Diameter QoS application [17] or the RADIUS QoS protocol. In case the response from the PDP is negative, the request must be rejected.

The basic concept of a session authorization by a policy decision is shown in Figure 4. A QoS NSLP Initiator (QNI) requests a policy decision from a policy decision point, e. g., a AAA server (step 1). The request must usually carry enough credentials to authenticate the user and the type of service that is requested. The AAA server's response (step 2) contains a Session Authorization Object which is embedded into the NSLP message that the QNI further sends (step 3) towards a QNE. The QNE extracts the Session Authorization Object and verifies its integrity. In case the signature from the authorizing entity is accepted, the NSLP message will be processed further.

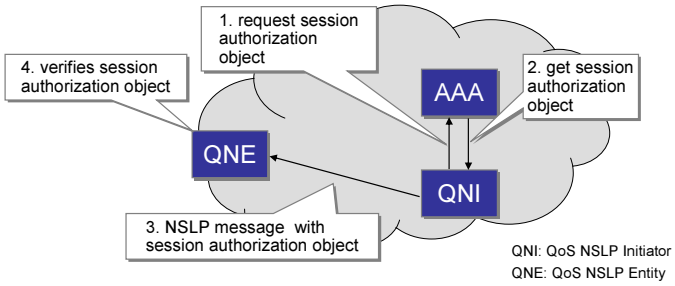

Figure 4. Session Authorization by Policy Decision

However, the scope of the Session Authorization Object as defined in the current draft is too limited as it only allows for passing authorization tokens within NSLP messages. The Session Authorization Object is not related to the actual QoS NSLP objects—it is rather an opaque authorization token. QoS NSLP objects, such as the Message Routing Information (MRI), the Session ID, or the QSPEC should be protected, too, and a protection of those types would be required for reliable accounting. Otherwise, an attacker would be able to change some of these values. An end-to-end protection of NSLP data cannot be achieved by the use of TLS at GIST level and this TLS protection cannot be assigned per user or per session, since GIST usually uses a TLS transport connection to multiplex several different NSLP sessions over it.

All of the above mentioned approaches, however, do not consider protection of the QoS NSLP message itself. They are simply embedding authorization tokens into the NSLP messages. An authentication of the QoS NSLP message itself is neither defined nor provided.

IV. Authentic QoS Signaling

In order to provide a stronger protection for NSIS signaling protocol messages, we propose some modest extensions to the current Session Authorization Object specification. The major

lacking piece is the missing link between the authorization object and the NSLP messages so that the information in the Session Authorization Object is nearly totally decoupled from the NSLP message itself. The only information that bind the authorization object and the NSLP message together is the source address attribute, if present, that must match the source address of the signaling message:

> The source address of the request (e. g., a QoS NSLP RESERVE) MUST match one of the SOURCE_ADDR attributes contained in this Session Authorization Data Policy Element.

So an attacker could use an isolated authorization object and re-use it for his own forged messages (must contain the original SOURCE_ADDR at least) within the valid time frame defined by start or end time respectively.

The base idea is now to extend the message authentication code in the Session Authorization Object to cover also parts of the GIST and NSLP message, e. g., *session ID*, the *MRI*, the *message type*, *RII*, *INFO_SPEC*, or *QSPEC* respectively, if present. For the QSPEC object special rules apply, because some parts of the QSPEC objects may be intentionally modified by QNEs during the reservation process. Thus, only those parts of the QSPEC object should be signed by the hash that are "read-only", i. e., not modified during transit, e. g., *QoS Desired*.

The proposed NSLP Session Authorization Object with the herein defined extensions should be registered under a different NSLP object type, so that the original proposed Session Authorization Object can still be used as session authorization token. Within the Session Authorization Object a new sub-type of the AUTH_ENT_ID indicates the self-signed NSLP message. Replay protection is provided by including the START_TIME attribute in every signed message. This is a 64-bit NTP timestamp providing a precision of 200 pico seconds.

For the remainder of the paper we assume that the *Message Authentication Code* (MAC) is calculated based on the HMAC algorithm [18] with a symmetric key. Other algorithms could be applied, but the HMAC is very efficient and simple to calculate.

The message authentication code could now be used to protect such parts of an NSLP message, too. In order to be flexible we propose to use a new attribute that contains a list of NSLP object types that are included in the HMAC (cf. Figure 5). The NSLP object type values comprise only 12 bit, so four bits per type value are currently not used within the list. Depending on the number of signed objects, a padding must be supplied. The receiver must look for the NSLP signed object list attribute and simply compute the hash over the content of these objects including all the attributes of the authorization object. Since all NSLP object types are unique over all different NSLPs, this works also for all NSLPs, not only QoS NSLP. Basic objects like the session ID, the NSLPID and the MRI are always included in the HMAC. Since they are not carried within the NSLP itself, but only within GIST, they must be delivered via the GIST API and normalized to their network representation from [9] again before calculating the hash. These values are hashed first, before any other NSLP object values are included in the hash computation.

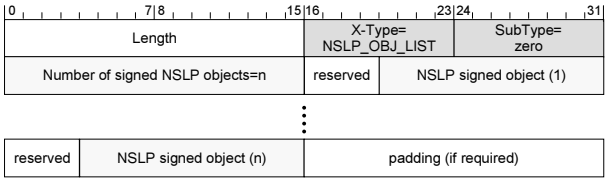

Figure 5. Signed NSLP Object List

A further missing part is the "crypto agility" of the approach in [15]. In order to provide flexibility with respect to different hash functions [19] we propose to specify an identifier for the used hash algorithm. The code is simply one byte using the same value mappings as specified in the IANA registry for Hash Algorithms as Transform ID from Transform Type 3 of the Internet Key Exchange (IKEv2) Protocol registry [20]. This allows for a smooth migration to stronger hash algorithms as they become available.

Since an HMAC requires a shared key on both sides, i. e., at the signing QNE (usually the QNI) and at the verifying QNE, there must be a possibility to provide the signing key to the QNE in a secure manner. The actually used key is referenced by a 32-bit key ID, which must also be transferred together with the signing key. We propose to use the session ID of the Session Authorization Object as the key ID (this session ID is not to be confused with the NSIS session ID in GIST). We aim to keep the overhead low, thus a user-based signature is proposed, i. e., every message is actually signed, but the key is not changed for every transaction or flow. Therefore, the approach is very well scalable, because the number of keys to store is related to the number of users, not to the number of flows. If a ticket is always present, there is even no need to store the key locally. A periodic change of the key is, however, recommended to prevent brute force attacks on the used keys. Furthermore, the key length should be at least 64-bit, but ideally longer in order to defend against brute force attacks. The next section describes a scenario where Kerberos [21] is used to securely distribute the shared signing key.

V. A Worked Example

In this section we describe how a Kerberos-based domain could use the proposed extended Session Authorization Object to simply authorize their users to reserve resources. We use Kerberos, because it provides a secure and simple means to authenticate users and allows as a single sign-on solution for a secure distribution of symmetric keys. Furthermore, Kerberos is the base for most domains that are using Windows Active Directory services.

First of all, we assume that routers in the Kerberos domain are Kerberized resources, i. e., they have a shared key with the Ticket Granting Server and can decode any Ticket Granting Tickets (TGTs) for their own resource. After the initial login the user is authenticated and can request tickets from the TGT

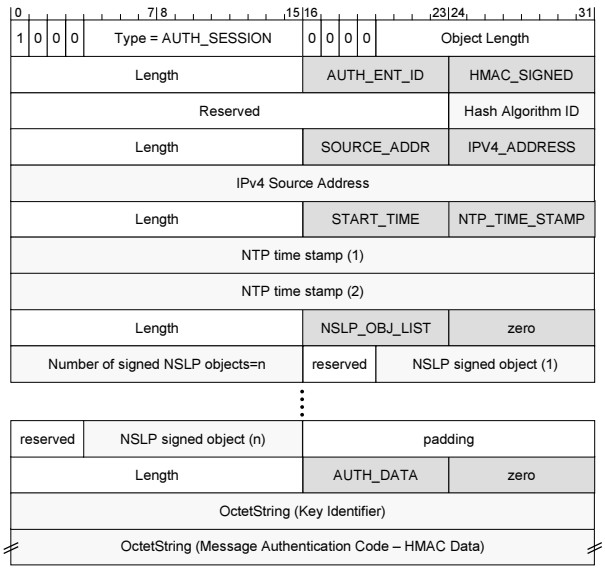

Figure 6. An example AUTH_SESSION object containing a hash-based message authentication code

as authorization against the QoS NSLP capable routers. Since the next signaling hop may not be known in advance (it is GIST's task to figure this out), a shared key between all routers of a domain and the TGT can be used.

The overall procedure works as follows (cf. Figure 7):

1) The user requests a TGT from the Ticket Granting Server (TGS), shown in step 1.
2) The user gets the answer back (step 2) and extracts the session key from the received ticket and uses it for HMAC computation. The user sends the TGT as Session Authorization Object, together with the proposed extended session authorization object as signature for the QoS NSLP message to the next signaling hop (step 3).
3) The router extracts the TGT and the session key. It stores the session key under the key index that is specified in the extended Session Authorization Object (step 4). The key should have a validity that corresponds to the lifetime of the ticket, i. e., the key will expire automatically after some time. Furthermore, the key ID must also point to the user identity, so that a corresponding user profile can be fetched in order to perform (local) policy-based admission control.
4) Subsequent messages usually carry only the start timestamp and the key index. The router then fetches the symmetric key from its internal map, calculates and verifies the HMAC. This is shown in Figure 9.

Figure 8 shows an example of an AUTH_SESSION object containing a resource ticket. The AUTH_ENT_ID contains the Kerberos principal name of the authorizing entity. In contrast to [14], we include the Kerberos ticket into the AUTH_DATA. This avoids an additional exchange of Kerberos messages. The main goal is to provide a secure way of transmitting the session key for the keyed hash algorithm to the QNE.

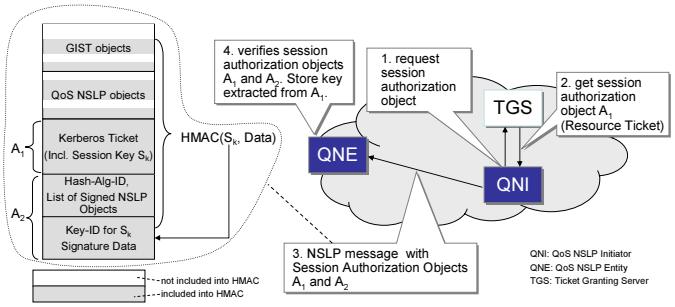

Figure 7. Overview of an initial session authorization using Kerberos

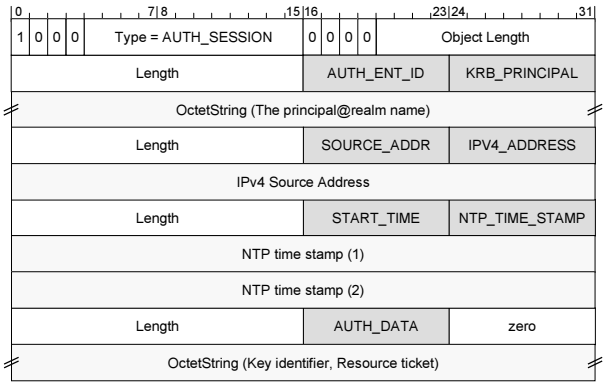

Figure 8. An example AUTH_SESSION object for Kerberos

After the resource ticket has been presented to the QNE, the QNE extracts and stores the session key contained in the ticket. Subsequent QoS NSLP messages do not need to contain another ticket as indicated by Figure 9. They can simply contain the list of signed NSLP objects and the HMAC data only, as shown in Figure 6.

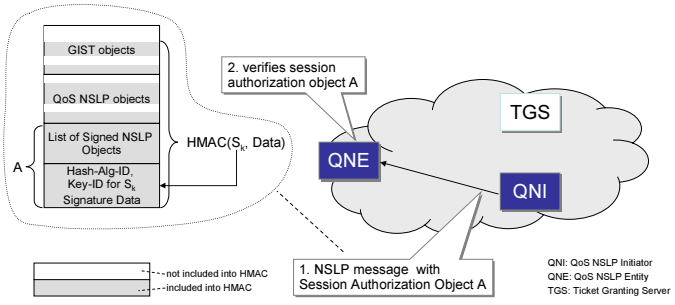

Figure 9. Overview of a subsequent session authorization

However, the currently used session key can be changed seamlessly any time by sending a new ticket to the QNE before the old ticket and before session key expiration. In case the user tries to use an already expired key, an authorization failure will be reported back in the corresponding response. In this case the user should request a new resource ticket before repeating the request.

As already mentioned earlier, although every QoS NSLP message can be signed, it is not necessary to send a new ticket for every new QoS NSLP session. Since user-based

authorization is usually sufficient, a new ticket is not required too often. We also note that responses sent back by the QNE can be signed by the QNE with the same session key. Thus, the QNI can verify the integrity of all response messages sent by the QNE.

VI. Evaluation and Conclusion

The proposed integrity protection of NSLP PDUs was implemented in the freely available NSIS-ka suite [22], which is C++ based. We used benchmarks to determine the overhead caused by the HMAC-based solution. Creation of the session authorization object including HMAC calculation requires up to 8% more time compared to a normal message generation effort and the session authorization HMAC verification costs up to 55% more of a message parsing process. Measurements were performed on an Intel Core2 Quad Q6600@2.40GHz by reading out the system clock at the specific actions and keeping the time stamps in memory. The following table shows the measured values of 100,000 runs in μs:

Action	Min	Max	Mean	Stddev
Serialize	9.148	45.463	9.316	0.209
Ser./HMAC	9.814	121.921	9.987	0.461
Deserialization	14.359	51.124	14.704	0.292
Deser./HMAC	22.434	74.831	22.757	0.476

An advantage of the presented approach is that important parts of the NSLP message are authenticated by using resource efficient HMAC-based signatures. The integrity of the NSLP messages can be verified by QNEs without any further back-end communication to PDP or AAA servers. Furthermore, resource consuming verification of public key-based certificates is avoided. Thus, authenticated signaling is now possible for every NSLP message sent. Other approaches presented in Section III require more communication overhead. Moreover, the presented approach is not restricted to a particular NSLP, thus it can be used to secure any NSLP.

A drawback of the proposed method is that it does not provide real non-repudiation, because symmetric keys are used for signing messages. Thus, the user must trust the QoS provider to not invent any resource reservation requests or to perform false accounting. Furthermore, confidentiality cannot be provided by this approach. But this is a fundamental conceptual problem anyway, because intermediate nodes must be able to read QoS NSLPs and QNEs may change them due to re-routing events.

As a next conceptual step we are investigating methods to use AAA infrastructures instead of Kerberos for providing the session key in a secure manner to the QNE. The QNE could use Diameter to ask the AAA server for the session key if the presented key ID is yet unknown.

Acknowledgment

The authors would like to thank Matthias Friedrich who proposed the usage with Kerberos. Part of this work was supported by Deutsche Telekom Laboratories and T-Systems within the ScaleNet project [5].

References

[1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Service," RFC 2475 (Informational), Dec. 1998, updated by RFC 3260. [Online]. Available: http://www.ietf.org/rfc/rfc2475.txt

[2] M. Brunner, "Requirements for Signaling Protocols," RFC 3726 (Informational), Apr. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3726.txt

[3] J. Manner and X. Fu, "Analysis of Existing Quality-of-Service Signaling Protocols," RFC 4094 (Informational), May 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4094.txt

[4] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification," RFC 2205 (Proposed Standard), Sep. 1997, updated by RFCs 2750, 3936, 4495. [Online]. Available: http://www.ietf.org/rfc/rfc2205.txt

[5] M. Siebert *et al.*, "ScaleNet - Converged Networks of the Future," *it - Information Technology, Themenheft "IP basierte mobile Systeme"*, vol. 48, no. 5/2006, pp. 253–263, Oct. 2006.

[6] X. Fu, H. Schulzrinne, A. Bader, D. Hogrefe, C. Kappler, G. Karagiannis, H. Tschofenig, and S. V. den Bosch, "NSIS: A New Extensible IP Signaling Protocol Suite," *Communications Magazine, IEEE*, vol. 43, no. 10, pp. 133–141, October 2005.

[7] J. Manner, G. Karagiannis, and A. McDonald, "NSLP for Quality-of-Service Signaling," Internet: http://tools.ietf.org/id/draft-ietf-nsis-qos-nslp, IETF, Feb. 2008, Internet Draft draft-ietf-nsis-qos-nslp-16.

[8] R. Hancock, G. Karagiannis, J. Loughney, and S. V. den Bosch, "Next Steps in Signaling (NSIS): Framework," RFC 4080 (Informational), Jun. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4080.txt

[9] H. Schulzrinne and R. Hancock, "GIST: General Internet Signalling Transport," Internet: http://tools.ietf.org/id/draft-ietf-nsis-ntlp, IETF, Oct. 2008, Internet Draft draft-ietf-nsis-ntlp-17.

[10] H. Tschofenig and X. Fu, "Securing the Next Steps In Signalling (NSIS) protocol suite," *International Journal of Internet Protocol Technology*, pp. 271–282, 2006.

[11] T. Tsenov, H. Tschofenig, X. Fu, and E. Körner, "Advanced Authentication and Authorization for Quality of Service Signaling," *Security and Privacy for Emerging Areas in Communication Networks*, pp. 224–235, Sep. 2005.

[12] S. G. Polito and H. Schulzrinne, "Authentication and Authorization Method in Multi-domain, Multi-provider Networks," *Next Generation Internet Networks, 3rd EuroNGI Conference on*, pp. 174–181, May 2007.

[13] S. Felis and M. Stiemerling, "Securing a Path-Coupled NAT/Firewall Signaling Protocol," in *IPOM 2007, ManWeek2007*, ser. Lecture Notes in Computer Science, no. 4786. Springer, 2007, pp. 61–72.

[14] L.-N. Hamer, B. Gage, B. Kosinski, and H. Shieh, "Session Authorization Policy Element," RFC 3520 (Proposed Standard), Apr. 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3520.txt

[15] J. Manner, M. Stiemerling, and H. Tschofenig, "Authorization for NSIS Signaling Layer Protocols," Internet: http://tools.ietf.org/id/draft-manner-nsis-nslp-auth, IETF, Jul. 2008, Internet Draft draft-manner-nsis-nslp-auth-04.

[16] L.-N. Hamer, B. Gage, and H. Shieh, "Framework for Session Set-up with Media Authorization," RFC 3521 (Informational), Apr. 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3521.txt

[17] D. Sun, P. J. McCann, H. Tschofenig, T. Tsou, A. Doria, and G. Zorn, "Diameter Quality of Service Application," Internet: http://tools.ietf.org/id/draft-ietf-dime-diameter-qos, IETF, Dec. 2008, Internet Draft draft-ietf-dime-diameter-qos-07.

[18] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication," RFC 2104 (Informational), Feb. 1997. [Online]. Available: http://www.ietf.org/rfc/rfc2104.txt

[19] P. Hoffman and B. Schneier, "Attacks on Cryptographic Hashes in Internet Protocols," RFC 4270 (Informational), Nov. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4270.txt

[20] C. Kaufman, "Internet Key Exchange (IKEv2) Protocol," RFC 4306 (Proposed Standard), Dec. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4306.txt

[21] C. Neuman, T. Yu, S. Hartman, and K. Raeburn, "The Kerberos Network Authentication Service (V5)," RFC 4120 (Proposed Standard), Jul. 2005, updated by RFCs 4537, 5021. [Online]. Available: http://www.ietf.org/rfc/rfc4120.txt

[22] Institute of Telematics, "NSIS-ka – A free C++ implementation of NSIS protocols," Feb. 2008. [Online]. Available: http://nsis-ka.org/