In Proc. of European Wireless 2004, Barcelona, Spain, February 2004

Denial of Service Protection for Optimized and QoS-aware Handover Based on Localized Cookies

Tianwei Chen[1], Günter Schäfer[1], Changpeng Fan[2], Stefan Adams[3], Michel Sortais[3], Adam Wolisz[1]

[1]Telecommunication Networks Group, Technische Universität Berlin, Germany
[2]ICM N PG SP RC PN, Siemens AG, Germany
[3]Mathematisches Institut, Fak. II MA 7-4, Technische Universität Berlin, Germany
e-mail: chen@tkn.tu-berlin.de

***Abstract*: Quality of Service (QoS) mechanisms in networks supporting mobile Internet communications give rise to new threats: these mechanisms could be abused by malicious entities launching so-called Denial of Service (DoS) attacks. If the network can not efficiently check the credibility of a QoS-request during a handover process, malicious entities could flood the network with bogus QoS-requests; if the authentication check is performed by means of an AAA protocol before the access network commits its resources to the request, the authentication process may not only introduce a notable latency to the handover process, but also generate an extensive traffic which degrades the signaling capacity in the network when there are a considerable amount of malicious requests. In order to defend against these kinds of attacks and meet the low-latency micro-mobility handover requirement, we [1]propose to have a preliminary authentication check with a cookie-based mechanism before processing the requests and performing authentication and authorization. The performance evaluation shows that the cookie-based mechanism is efficient in dealing with the identified issues.**

1. Introduction

The introduction of advanced *Quality of Service (QoS)* mechanisms, which aim to guarantee certain service characteristics like end-to-end delay, jitter, etc., in mobile networks give rise to new threats that these mechanisms could be abused by malicious entities to launch so-called *Denial of Service (DoS)* attacks, which aim at reducing the availability of services to legitimate users.

In an IP-based access network, a mobile node (MN) sends a request to an access router (AR) for a certain resource (see Fig.1 for an illustration). If the network can not check the credibility of a QoS request (i.e. whether the request originates from a MN that is actually authorized to use the services it is requesting), malicious entities can flood the network with bogus QoS requests in order to cause the exhaustion of the available resources through temporal reservations. This represents one specific DoS threat.

A potential solution consists in the following procedure: when an access router (AR) receives a QoS request, before starting the resource reservation process, the AR communicates with a local security authority, e.g. a local AAA server (AAAL), to authenticate the MN and authorize the QoS request. Only when the check passes, the path reserves resources according to the request. Obviously, the latency introduced by proceeding to security checks at the AAAL, which includes the contribution of the propagation delay and processing time at AAAL, is not desirable when low latency of the registration process is a major concern. Moreover, the same checks at an AAA server have to be performed on all the bogus requests from attackers. Thus all the security check signalings may degrade the performance of the access network substantially by depleting the signaling capacity of the path between the AR and the AAAL and exhausting the computing resource of the AAAL. This represents another specific DoS threat.

To defend against the identified DoS threats and meet the low-latency requirement in intra-domain handovers, we propose a two-step procedure comprising of one preliminary credibility check, after which processing of the signaling request is either aborted or continued, and the second definitive authentication check as described above. The credibility check should have the following properties: performing the first check must be a quick operation and ARs must not keep per-session or per-user state until the verification is complete.

Up to now, two principle approaches for checking the credibility of a request have been proposed: *exchanging "cookies"* and *solving "client puzzles"*. The concept of exchanging "cookies" has been introduced in the context of transactions between web servers and browsers, where cookies are pieces of information generated by a Web server and stored in the user's computer, ready for future access. This idea has been adopted to provide protection against resource exhaustion DoS attacks in IPSec's key exchange protocol ISAKMP and the mobility support in IPv6 design [4]. In the "client puzzle" approach [1], a client is asked to solve a cryptographic puzzle and the server stores the protocol state and executes expensive operations only after it has verified the client's solution. In this way, the puzzle can prevent intensive connection initiations from attackers, thus enhancing the DoS-resistance of a server. However, solving cryptographic puzzles imposes a computational burden to all legitimate clients, as well as requiring additional message elements to be exchanged. It would add a non-negligible latency to the establishment of a connection between client and server.

In this paper, therefore, we propose to use a cookie-based mechanism as the first credibility check. A cookie is verified by an AR to ensure that the QoS-request sender is a credible registered user before processing the requests and performing authentication and authoriza-

[1]This work has been supported in part by a research contract with Information and Communication Mobile, Siemens AG.
The work of S. Adams was supported by DFG Research Center "Mathematics for key technologies" (FZT 86) Berlin.
M. Sortais was supported by the Graduiertenkolleg "Stochastische Prozesse und Probabilistische Analysis" Berlin and by the swiss NSF.

tion. This preliminary check enables us to prevent DoS attacks, both in the form of resource reservations along a path or keeping the AR busy through the processing of malicious QoS requests, with the help of a AAAL or possibly the home AAA server (AAAH).

Our use of the cookie idea as a first step of authentication - a preliminary check of a request's credibility - is completely new in IP based mobile networks. After AR filters out most of the bogus requests without cookie or with false cookies by means of cookie verification, the access network has lower burden to authenticate QoS requests - the second step of authentication. Since it can not be completely ruled out that an attacker gains access in the network with an eavesdropped cookie, it is necessary to perform the second step authentication with the help of the local AAA server.

The remainder of the paper is organized as follows: Section 2. introduces related work; Section 3. describes the cookie-based mechanism in detail; Section 4. presents a performance analysis of the mechanism. An overview of the corresponding results is given in Section 5., and Section 6. summarizes our contribution.

2. Related work

So far the most common form of DoS is to cause excessive bogus traffic to a particular server so as to prevent legitimate users from getting services. To deal with this kind of DoS attack, a couple of defense techniques have been published, e.g. [1].

As mentioned above, the basic idea of cookies is adopted in the mobility support in IPv6 design. The correspondent nodes (CNs) do not have to retain any state about individual MNs until an authentic binding update (BU) arrives. When receiving a BU message, the CN includes a cookie in a message sent back to the sender according to the source address in the in-coming BU message. If the source address is not a bogus one from an attacker, the genuine sender will include the cookie in its following messages to the CN. It is stated that the cookie mechanism can protect the CN against memory exhaustion attacks except where on-path attackers are concerned. In contrast, our scheme deals with DoS attacks on the signaling capacity of the access network, which has not been addressed before.

Currently QoS, DoS and low-latency handovers have been dealt with separately. In the signaling design efforts of the Next Steps in Signaling (NSIS) working group in IETF, especially regarding QoS signaling, the aforementioned DoS attacks have been identified. However, there are no QoS or resource restriction mechanisms that address these issues efficiently and practically. While Context Transfer Protocol [5] can be used to transfer the authentication data from the old AR (oAR) to the new AR (nAR) during a handover, the authentication process at a nAR is not the most optimized in non-predictive handovers since a nAR needs to communicate with an oAR for the authentication data during a handover. Moreover, the assumption that the neighboring ARs share a pre-established security association (SA) is not always true.

3. A cookie-based mechanism

Figure 1 shows an overview of the network with a hierarchical Mobile IPv6 (HMIPv6) [7] and Authentication, Authorization and Accounting (AAA) [2] joint architecture, where the cookie-based mechanism is applicable.

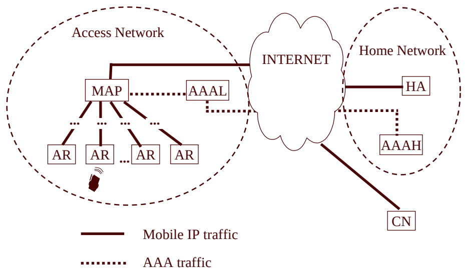

Figure 1: An overview of the involved entities

During the intra-domain handover procedure, the MN needs to find out whether the path from the new AR to MAP can meet its QoS request. Each router along this path in the access network must determine whether it has sufficient resources to satisfy the per-hop QoS requirement of an MN's session. If the path from the new AR to MAP can satisfy the QoS request, it reserves the resource for the related session. Before the resource availability check and reservation, an efficient authentication check is mandatory to prevent DoS attacks and achieve low-latency handovers.

The cookie-based mechanism is designed as an efficient authentication scheme meeting the following requirements: An AR can verify a cookie immediately when it receives a QoS request; the cookie generation and verification must be light-weighted; the entities in the access network are free from keeping per-client states until the cookie check passes. In the following, we first describe the data structure of a cookie and the mechanism operations, then we give a discussion and a summary.

3.1. The data structure of a cookie

Figure 2 shows the data fields of a cookie. The purpose of each field may be listed as follows:

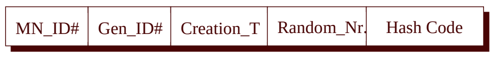

Figure 2: Cookie data fields

- *MN_ID*: is the MN's unique identifier. This can be a local unique identifier the MN gets after its first registration.
- *Gen_ID*: identifies the cookie generator which is always an AR. It can be the AR's IP address or another unique identifier acceptable in the access network.
- *Creation_T*: is the timestamp marking when the cookie was generated. It is used to limit the cookie's period of validity.

- *Random_Nr*: is used to distinguish two cookies which are generated at the same time.
- *CookieHash*: The hash code is a message digest of the cookie information and a cookie key. The hash function can be of either HMAC-MD5 or HMAC-SHA1. The cookie key could be distributed from the MAP to each AR and updated by the MAP periodically. For example a new cookie key is distributed by the MAP every hour or day.

3.2. Mechanism Operations

The cookie-based mechanism is described in an access network which is based on a HMIPv6 and AAA joint architecture. The architecture includes an AAAL, a mobility anchor point (MAP) and ARs positioned linearly as shown in Figure 3.

- First cookie generation

  When a mobile user enters an access network (e.g. it performs a global movement or powers up), the authentication on its first QoS request must involve a trusted network (e.g. the mobile user's home network) because the user is unknown to the access network at the moment.

  After the authentication at AAAH, the access network knows that the user is credible, AAAL caches the MN's authorization information and MAP, AR2 (taken to be the associated AR in our example, see Figure 3) and MN get to know the session key which is generated by the MN's home domain.

  AR2 generates a cookie, encrypts the cookie with the session key, inserts the encrypted cookie in the BU acknowledgement (BU ACK) message which is generated by MAP and destined to MN. The MN can get its first cookie in the access network since the MN can derive the session key due to its long term trust relationship with its home domain. Thus, MN gets its first cookie in the access network.

- Cookie verification

  In a local movement, as shown in Figure 3, MN presents the cookie to a neighboring AR server (say AR3). Because there is no security association between MN and AR3 so far, the cookie is transmitted in plaintext.

  When receiving the cookie, AR3 first verifies that the cookie is valid with the following checks:

  - check the timestamp in the cookie to verify the cookie is not expired;
  - check the identity of the cookie generator to verify the cookie is created by an AR on its trusted list;
  - verify that the cookie is not on the notified cookie list;
  - if the above checks pass, AR3 computes a key-hashed digest of the cookie information by using a cookie key, and compares the computation result with the hash digest contained in the cookie.

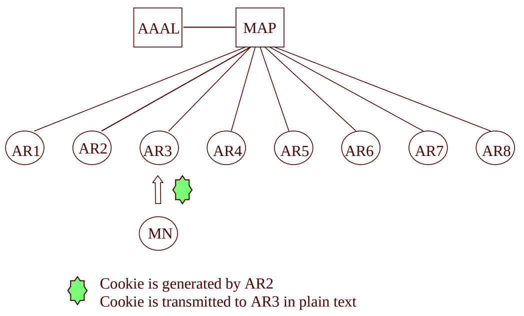
AAAL
MAP
AR1
AR2
AR3
AR4
AR5
AR6
AR7
AR8
MN
Cookie is generated by AR2
Cookie is transmitted to AR3 in plain text

Figure 3: AR3 verifies the cookie presented in plain text from the MN

  If the two hash digests match, the cookie check verification is completed successfully.

  After verifying a cookie successfully, AR3 informs AR2 who originally generated the cookie that it has been presented to it. AR2 then notifies all other ARs on its trusting list to invalidate the cookie, preventing these ARs to accept it again; indeed, an attacker could intercept the cookie from the open wireless interface and replay it to cheat these ARs for access. After the expiration of a cookie's lifetime, all ARs can delete it from the notified cookie list.

- New cookie granting

  When the cookie verification is completed successfully, the QoS-conditionalized BU [3] and authorization processes start. The two processes can proceed in series or in parallel. If the two processes are successful, the AR3 can do authentication of the registration request when it gets the session key embedded in the BU ACK message from the MAP. When this check passes, the AR3 generates a new cookie, encrypts it with the session key and inserts it in the BU ACK message. The new cookie is used for its next registration and the old cookie is no longer valid in the access network.

3.3. Discussion

There are three main points in the design of this cookie mechanism: the "area of validity", the notification of a used cookie, and the presentation of a cookie in plain text.

- Area of Validity

  The "area of validity" is a group of ARs at which a cookie is valid. In other words, the "area of validity" corresponds to the trusting list of the cookie generator.

  When a cookie is presented in plain text to AR3 (see Figure 3), if AR3 were not to notify other ARs about the use of the cookie, the cookie could be intercepted by an attacker on air and replayed to other ARs. Consequently, the attacker could gain access at these ARs so as to play DoS attacks on the corresponding paths. On the other hand, if each AR were to notify the rest of the access network when receiving a cookie, the propagation of notification messages would generate substantial traffic in the access network.

Therefore, we introduce a limited "area of validity" for each cookie, the nodes in this area being the only ones that can accept the cookie. For example, each AR could have its adjacent ARs only on its trusting list and trusted list.

- Notification of used cookies

  After verifying the cookie, AR3 notifies immediately AR2 about the cookie use and then AR2 notifies AR1, who is the remaining AR on AR2's trusting list, not to accept the cookie. All ARs are then free from replay attacks, provided that the notification messages propagate faster than the time needed for an attacker to intercept a cookie and replay it.

- Presenting a cookie in plain text

  Mobile nodes always send cookies in plain text to access routers, as in the case of a handover, since the MN does not yet share a session key with the new access router at the time it sends the cookie. Although the cookie might be intercepted by an attacker when being presented in plain text, the risk of DoS attacks is reduced sufficiently with the mechanisms described above. The reason for this is, that the cookie mechanism reduces the overall number of "credible looking" handover requests, as every cookie can only be presented once.

4. Performance evaluation

In order to evaluate the performance of the cookie-based mechanism in reducing the risks of DoS attacks and in benefitting from the optimized intra-domain handovers, three parameters should be examined:

- *Mean response time*: The duration between the transmission of the first bit of a registration request of a legitimate MN and the arrival of the last bit of the corresponding registration response.

- *Mean queue length at AAAL*: How long in average is the queue of new jobs waiting for service at the AAAL server?

- *Size of the waiting room at AR*: Another disadvantage of the no-cookie scenario lies in the fact that the AR server has to store some (e.g. 500 bits long) jobs until receiving an answer from AAAL, and many of these jobs may actually correspond to false requests. Considering the AR server of a cell that is under attack, we use a (continuous time) Markov chain method in order to compute the mean value of the total queue length in such a "waiting room".

These parameters are examined in three different processing schemes as shown in Figure 4, Figure 5 and Figure 6 respectively:

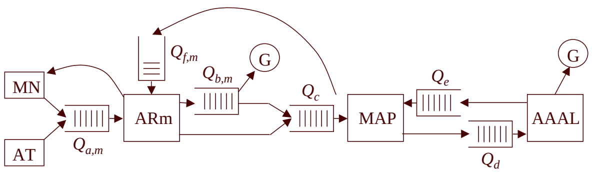

Figure 4: Processing scheme in Case 0

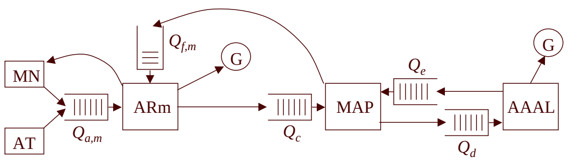

Figure 5: Processing scheme in Case 1

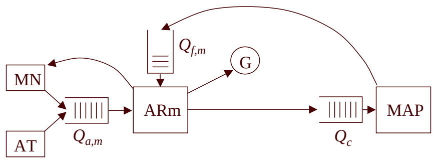

Figure 6: Processing scheme in Case 2

- *Case 0: No cookie protection*: In one cell which has one AR, MNs and attackers (AT) send QoS requests in registration messages to the AR (AR_m). The arrival messages are queued in $Q_{a,m}$ of AR_m if the processor is busy. Since (AA) checks need to be performed at AAAL before initiating the source reservation process, AR_m caches temporarily the corresponding QoS request information in $Q_{b,m}$ (which serves as a "waiting room"). When a reply message for a genuine request arrives at AR_m, AR_m discards all the priors in $Q_{b,m}$ since these are regarded as bogus requests, provided all the (AA) checks at AAAL require the same time.

- *Case 1: Cookie protection and the resource reservation and authorization processes are in series*: In the AR_m's cell, MNs and ATs send QoS requests with cookies to $Q_{a,m}$ of AR_m. Before processing the requests, AR_m verifies the cookie. If the verification fails, AR_m drops the request in "G" (denoting garbage) silently. If the verification succeeds, AR_m sends the notification message and starts the resource reservation procedure immediately, checking its available bandwidth and sending the QoS request to Q_c of the MAP. The MAP performs the same check. Before the MAP sends a reply destined to MN, including the check result and the session key, to $Q_{f,m}$ of AR_m, it generates a new message and sends it to Q_d of the AAAL for the authorization check. When the authorization check passes, the AAAL sends a message to Q_e of the MAP. When AR_m receives the reply from the MAP, AR performs an authentication check to the QoS request with the session key. If the check passes, while removing the session key from the reply message, it generates a new cookie, encrypts it with the session key, inserts the encrypted cookie in the reply, and forwards it to the corresponding MN.

- *Case 2: Cookie protection and the resource reservation and authorization processes are in parallel*: AR_m performs the same cookie verification when receiving a QoS request from either a MN or an AT. If the verification fails, AR_m also drops the request to "G". If the verification passes, the AR_m starts the two processes in parallel. Meanwhile, it sends the notification message. It is assumed that the result of the authorization process arrives earlier than that of the resource reservation process. Therefore, the time spent on the authorization process has no contribution to the response time of a registration process and the authorization process can be ignored from the analysis.

In all three cases, λ_{MN} denotes the Poisson rate of message sendings by MNs in a given cell, whereas λ_{AT} denotes the Poisson rate of messages sent by Attackers in a cell that is currently under attack. We also let M denote the total number of cells, whereas N stands for the total number of cells that are currently under attack. The message sending process in a given cell (cell $\sharp\ m$, $1 \leq m \leq M$) is therefore a Poisson process with an intensity λ that is either equal to λ_{MN} (no attack) or to $\lambda_{MN} + \lambda_{AT}$ (attack).

The asymptotic mean arrival rate at $Q_{a,m}$, namely$\lambda^*_{a,m}$, is defined as:

$$\lambda^*_{a,m} = \lim_{t \to +\infty} \frac{\mathbb{E}\left[\sharp(\text{arrivals at } Q_{a,m} \text{ during time}[0,t])\right]}{t},$$

and we let λ^*_c, λ^*_d, λ^*_e and $\lambda^*_{f,m}$ be the asymptotic mean arrival rates corresponding to Q_c, Q_d, Q_e and $Q_{f,m}$ respectively.

The asymptotic mean arrival rates at each of the remaining queues may then be determined as functions of M, N, λ_{MN} and λ_{AT}. Indeed, in Case 0:

$$\begin{aligned}
&\lambda^*_c = 2M\lambda_{MN} + N\lambda_{AT},\\
&\lambda^*_d = N(\lambda_{MN} + \lambda_{AT}) + (M-N)(\lambda_{MN}),\\
&\lambda^*_e = M\lambda_{MN}, \quad \lambda^*_{f,m} = 2\lambda_{MN};
\end{aligned}$$

in Case 1: $\lambda^*_c = M\lambda_{MN} = \lambda^*_d = \lambda^*_e, \lambda^*_{f,m} = \lambda_{MN}$, and in Case 2: $\lambda^*_c = M\lambda_{MN}, \lambda^*_{f,m} = \lambda_{MN}$.

4.1. Waiting time at different queues, total response time

In our analysis we use the following three basic principles of queueing theory: *Little's theorem*, the *renewal theorem* and the *Pollaczek-Khinchine* formula (as applied to priority queueing systems, see e.g. [6]). Consider the random variables $W^{(k)}_{a,m}$ and $R^{(k)}_{ARm}$ defined by:
$W^{(k)}_{a,m}$:= total time spent waiting in queue $Q_{a,m}$ by job $\sharp k$, and
$R^{(k)}_{ARm}$:= residual service time in ARm upon arrival of k^{th} job.
Define $W^{(k)}_{b,m}$, $W^{(k)}_c$, $R^{(k)}_{MAP}$, $W^{(k)}_d$, $R^{(k)}_{AAAL}$, $W^{(k)}_e$ and $W^{(k)}_{f,m}$ in a similar way (job $\sharp k$ in Q_c is the k^{th} job having been stored in Q_c). We are interested in limits such as

$$\lim_{k \to +\infty} \mathbb{E}\left(W^{(k)}_{a,m}\right),$$

and we simply write $\mathbb{E}(W_{a,m})$ for it in the sequel, calling it the mean queue length or mean waiting time at $Q_{a,m}$. (Refer to Table 2 for a list of processing time parameters).

We come to the computation and plotting of the asymptotic mean value of the *total response time* τ .
$\diamond$ In Case 0 we have

$$\begin{aligned}
\mathbb{E}(\tau) = {} & t_0 + \mathbb{E}(W_{a,m}) + t_1 + C^{(0)}_{2,3} + \mathbb{E}(W_c) + C^{(0)}_{3,4}\\
&+\mathbb{E}(W_d) + t_3 + T + C^{(0)}_{4,3} + \mathbb{E}(W_e) + 2t_2 + C^{(0)}_{3,2}\\
&+\mathbb{E}(W_{f,m}) + 4t_4 + \hat{C}^{(0)}_{2,3} + \mathbb{E}(W_c) + \hat{C}^{(0)}_{3,2}\\
&+\mathbb{E}(W_{f,m}) + t'_0
\end{aligned}$$

where t_0 is the transmission time for the wireless up-link channel and where $C^{(0)}_{i,j}$ are the transmission parameters for the different messages and links.
$\diamond$ In Case 1 we have

$$\begin{aligned}
\mathbb{E}(\tau) = {} & t_0 + \mathbb{E}(W_{a,m}) + 4t_4 + 3T + C^{(1)}_{2,3} + \mathbb{E}(W_c)\\
&+C^{(1)}_{3,4} + \mathbb{E}(W_d) + C^{(1)}_{4,3} + \mathbb{E}(W_e) + t''_0 + t_3\\
&+C^{(1)}_{3,2} + \mathbb{E}(W_{f,m})
\end{aligned}$$

$\diamond$ Finally in Case 2 we have

$$\begin{aligned}
\mathbb{E}(\tau) \quad = \quad & t_0 + \mathbb{E}(W_{a,m}) + 3t_4 + 3T + C^{(1)}_{2,3} + \mathbb{E}(W_c)\\
&+C^{(1)}_{3,2} + \mathbb{E}(W_{f,m}) + t''_0
\end{aligned}$$

4.2. Queue length at AAAL

The mean waiting time at Q_d was already computed in the preceding subsection; we thus obtain for Case 0:

$$\mathbb{E}(L_d) = \frac{(N\lambda_{AT} + M\lambda_{MN})\left\{(N\lambda_{AT})T^2 + (M\lambda_{MN})(T+t_3)^2\right\}}{2(1 - \{(N\lambda_{AT})T + (M\lambda_{MN})(T+t_3))\}}$$

and for Case 1

$$\mathbb{E}(L_d) = \frac{\frac{(M\lambda_{MN})^2}{2}t_3^2}{1 - M\lambda_{MN}t_3}.$$

4.3. Queue length in the "waiting room"

In the analysis of the length of $Q_{b,m}$, the only jobs that really matter (those that will eventually undergo service) are the "good jobs" (corresponding to genuine MN). Now the asymptotic mean total time elapsed between the storage of a "good job" in $Q_{b,m}$ and its marking is given by

$$\begin{aligned}
\mathbb{E}(W_{b,m;MH}) = {} & C^{(0)}_{2,3} + \mathbb{E}(W_c) + 2t_2 + C^{(0)}_{3,4} + \mathbb{E}(W_d)\\
&+t_3 + T + C^{(0)}_{4,3} + \mathbb{E}(W_e) + C^{(0)}_{3,2} + \mathbb{E}(W_{f,m}) + t_4
\end{aligned}$$

Via Little's Theorem one may thus express the mean asymptotic length of $Q_{b,m}$ as

$$\mathbb{E}(L_{b,m}) = \lambda_{MN}\mathbb{E}(W_{b,m;MH})\left(1 + l^{(1)}_{b,m}\right) + l^{(2)}_{b,m},$$

the constant $l^{(2)}_{b,m}$ above denoting the asymptotic mean number of "residual bad jobs" in $Q_{b,m}$ (those "bad jobs" that are located below the lowest "good job" in $Q_{b,m}$), whereas $l^{(1)}_{b,m}$ stands for the asymptotic mean number of "bad jobs" that are located in between two consecutive "good jobs" in $Q_{b,m}$.

Table 1: Link and message length parameters

Parameter	Value
Wireless link MN↔AR	11/54 Mbps
Wired link AR↔MAP↔AAAL	100 Mbps
Wireless PHY+MAC Header	58 bytes
Wired PHY+MAC Header	26 bytes
IPv6 Header	40 bytes
QoS Hop-By-Hop Option	82 bytes
Home Address Option	18 bytes
BU / BU ACK	6 bytes
ESP Header	8 bytes
ESP Authentication Extension	16 bytes
Authenticator	20 bytes
Cookie	32 bytes (HMAC-SHA)

Table 2: Processing time parameters

Symbol	Time*(μs)	Remark
t1	152	Generate an AA request / answer
t2	20	Forward an AA request / answer
t3	152	Perform an authorization check
t4	220	Check, reserve or confirm resources
T	40	Perform authentication check; Generate or verify a cookie

* The processing time values are obtained from measurements on Pentium III 600 machines.

5. Results and Interpretation

In this section we will evaluate the metrics derived in the previous section with actual parameters from exisiting technologies in order to obtain a performance assessment of our cookie protection scheme for realistic scenarios. Table 1 shows the assumed link speeds and the message lengths according to the involved communication and signaling protocols and Table 2 lists up the processing times of individual protocol steps obtained by measurements with a prototypical implementation.

5.1. Total response time

Figure 7 shows the total response time in relation to the attacking rate per cell, when the wireless links are assumed optimistically to offer 54 MBit/s (physical layer according to IEEE 802.11a, leading to $t_0 \approx 50\mu s$), $M = 50$ cells are connected to one AAA-server, 10 cells are under attack and $x = 40[\text{messages}/sec]$ are sent by genuine clients. As can be seen, a DoS situation occurs in case 0 but not in the cookie mechanism cases. In case 0 the total response time grows abruptly when the attacking rate approaches $1500\text{messages}/sec$. The cookie mechanism requires around one additional millisecond of processing but the system is able to serve genuine clients up to any attacking rate, of course provided that there is still bandwidth left in the respective radio cell. Therefore, our DoS protection scheme offers

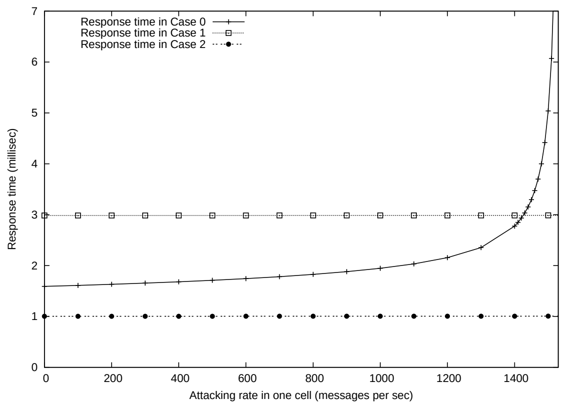

Figure 7: Total response time for High Speed uplink with $t_0 \approx 50\mu s$, M= 50 cells and N= 10 cells under attack for MN rate $x = 40[\text{messages}/sec]$

a significant improvement over the unprotected case.

In case of a slower wireless uplink with $t_0 \approx 300\mu s$ (corresponding to an IEEE 802.11b WLAN operating at 11 MBit/s with long physical layer preamble) Figure 8 again shows that a DoS situation would not occur before around $1500\text{messages}/sec$ are send per cell. However, this attacking rate well exceeds the range in which it can be assumed, that the attacker will be able to send his attacking packets over the wireless channel. Therefore, our DoS protection scheme does not offer a benefit in cases with rather low wireless channel capacity, or otherwise stated, more wireless cells have to be supported with one AAA server until our DoS protection scheme offers a significant improvement.

The cost of implementing the cookie mechanism corresponds to the discrepancy between the lines of response time in Cases 0 and 1 (see Figure 7 and Figure 8). Before the attacking rate reaches a saturation point, Case 0 performs slightly better than Case 1; however, when the attacking rate is running over the saturation point, the cookie mechanism takes obvious effect in preventing a DoS attack and at any rate, a cost of less than 2 milliseconds is negligible.

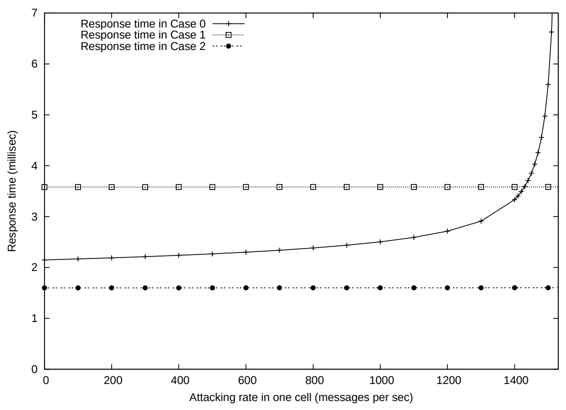

Figure 8: Total response time for Slow Speed uplink with $t_0 \approx 300\mu s$, M= 50 cells and N=10 cells under attack for MN rate $x = 40[\text{messages}/sec]$

5.2. Queue length at AAAL

Figure 9 shows the queue length at the AAA server for case 0 under the same conditions as in figure 7. This graph clearly shows that the DoS situation is caused by the overloading of the AAA server and not by exceeding the transmission capacities of the access network. Under these conditions the AAA server is not able to keep up with checking and discarding bogus messages being sent by attackers so that the genuine requests from honest clients can not be processed in time.

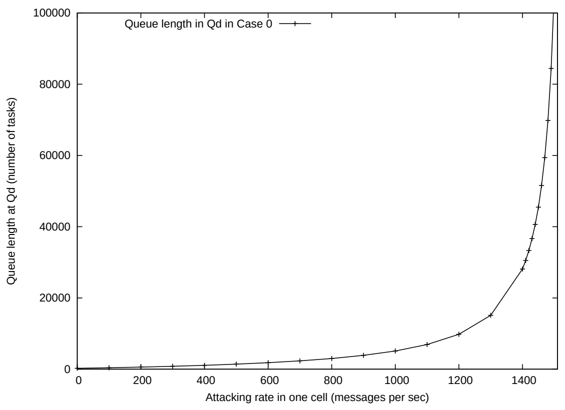

Figure 9: Queue length at AAAL for High Speed uplink $t_0 \approx 50\mu s$, M=50 cells and N=10 cells under attack for MN rate $x = 40[\text{messages}/sec]$

5.3. Queue length in the "waiting room"

Figure 10 shows the queue length in the "waiting room" of an access router in a cell which is under attack (here we assume that 25 out of 50 cells are under attack). Depending on the rate of genuine requests by honest clients ($x = 10\text{messages}/sec$ vs. $x = 40\text{messages}/sec$) a DoS situation will occur earlier. The reason for this behaviour lies in our strategy to silently discard bogus requests directly after they have been identified at the AAA server in order to save AAA processing capabilities. This implies that access routers need to receive a response to a genuine request, in order to be able to discard all bogus messages betweeen two genuine requests. Furthermore, it can be seen from this graph that memory depletion situations can occur at access routers under attack. However, the most critical system in the access network infrastructure is the central AAA server.

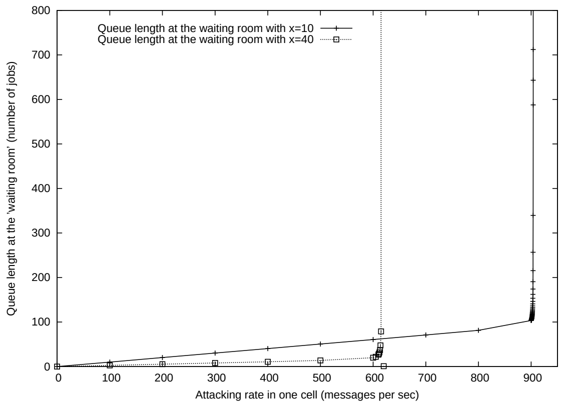

Figure 10: Queue length in the waiting room for High Speed uplink $t_0 \approx 50\mu s$, M=50 cells and N=25 cells under attack and different MN rates x

6. Conclusions

In this paper we described a cookie-based mechanism to protect against DoS attacks for optimized and QoS-aware handovers by performing a simple and preliminary check with a cookie before the QoS reservation and authorization processes begin. We also provided a performance evaluation which shows that the cookie mechanism is a method to protect against DoS attacks in the QoS reservation process in a distributed scenario, to speed up registration in the intra-domain handover case by parallelizing QoS reservation process and AA process without introducing additional DoS risks.

Furthermore, our scheme reduces the risk of replayed cookies by implementing an *"area of validity"* in which a cookie is acceptable, and by communicating cookies that have been used once at a particular AR to other ARs in the same area of validity.

The mechanisms of this solution can additionally protect against the following depletion threats (which exist when authentication and resource reservation are performed in parallel or sequentially) against depletion of the memory of access routers that would have to maintain state while the authentication of the MN is fetched from the AAAL, against depletion of signaling capacity in the access network (by preventing signaling traffic for bogus requests which have not been verified before as being "credible"), and against depletion of the resources of the AAAL (by shielding the AAAL form authentication requests which result from bogus QoS requests).

REFERENCES

[1] T. Aura, P. Nikander, and J. Leiwo. DOS-resistant authentication with client puzzles. *Lecture Notes in Computer Science*, 2133:170, 2001.

[2] P. Calhoun, J. Loughney, E. Guttman, G. Zorn, and J. Arkko. Diameter Base Protocol, September 2003. RFC 3588.

[3] X. Fu, H. Karl, and C. Kappler. QoS-Conditionalized Handoff for Mobile IPv6. In *Proc. of the 2nd IFIP-TC6 Networking Conf. (Networking2002)*, pages 721–730, Pisa, Italy, May 2002. Springer-Verlag.

[4] D. Johnson, C. Perkins, and J. Arkko. Mobility Support in IPv6, June 2003. draft-ietf-mobileip-ipv6-24.txt.

[5] J. Loughney, M. Nakhjiri, C. Perkins, and R. Koodli. Context Transfer Protocol, October 2003. draft-ietf-seamoby-ctp-05.txt.

[6] P. Robert. *Stochastic Networks and Queues*. Applications of Mathematics, Vol. 52, Springer-Verlag, 2003.

[7] H. Soliman, C. Castelluccia, K. El-Malki, and L. Bellier. Hierarchical Mobile IPv6 mobility management (HMIPv6). Internet-Draft: draft-ietf-mipshop-hmipv6-00.txt, June 2003.