Tero Paloheimo

Overload Control in Diameter for Mobility Management

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 6.5.2015

Thesis supervisor:

Prof. Tarik Taleb

Thesis advisor:

M.Sc. (Tech.) Henri Poikonen

Aalto University
School of Electrical
Engineering

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Tero Paloheimo

Title: Overload Control in Diameter for Mobility Management

Date: 6.5.2015 Language: English Number of pages: 9+60

Department of Communications and Networking

Professorship: Networking technology Code: S-38

Supervisor: Prof. Tarik Taleb

Advisor: M.Sc. (Tech.) Henri Poikonen

The purpose of this thesis is to develop an overload control solution for Diameter protocol traffic between the MME and HSS in the EPC. The solution is based on using overload information from HSSs and throttling the traffic as needed based on the information. Diameter message prioritization is used to enable ongoing transactions to succeed and thus reduce the overload. The performance of the solution is measured with four indicators measuring various aspects of the system's performance. Performance measurements are done via simulations and their results show a clear improvement in system performance when the solution is enabled.

Keywords: Diameter EPS EPC HSS MME overload control

AALTO-YLIOPISTO
SÄHKÖTEKNIIKAN KORKEAKOULU

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä: Tero Paloheimo

Työn nimi: Ylikuorman hallinta Diameter-protokollassa liikkuvuuden hallintaa varten

Päivämäärä: 6.5.2015 Kieli: Englanti Sivumäärä: 9+60

Tietoliikenne- ja tietoverkkotekniikan laitos

Professuuri: Tietoverkkotekniikka Koodi: S-38

Valvoja: Prof. Tarik Taleb

Ohjaaja: DI Henri Poikonen

Työn tarkoituksena on kehittää ylikuormanhallintaratkaisu neljännen sukupolven matkapuhelinverkkojen ytimen EPC:n MME- ja HSS-verkkoelementtien väliselle Diameter-protokollan yli tapahtuvalle liikenteelle. Hallintaratkaisu perustuu HSS-elementin MME-elementille välittämään tietoon ylikuormasta ja siihen perustuvaan MME-elementin tarpeen vaatiessa tekemään liikenteen rajoittamiseen. Käynnissä olevien transaktioiden lähettämisen onnistuminen varmistetaan käyttämällä Diameter-sanomien priorisointia, mikä vähentää ylikuormaa, kun niitä ei tarvitse lähettää uudestaan. Ratkaisun suorituskykyä mittaamaan on kehitetty neljä eri osa-alueiden suorituskykyä kuvaavaa mittaria. Varsinainen suorituskyvyn mittaus on tehty toteuttamalla ratkaisua simuloiva sovellus ja simulaatioiden tulokset osoittavat järjestelmän suorituskyvyn selkeästi parantuneen ratkaisun ollessa käytössä.

Avainsanat: Diameter EPS EPC HSS MME ylikuorman hallinta

Preface

I would like to thank my supervisor, Professor Tarik Taleb, and advisor Henri Poikonen for all their advice, guidance, and patience during process of writing this thesis. Additional thanks go to Mika Nevander and Niklas Wik for their insight on Diameter and development suggestions towards the solution. I am very grateful that Nokia provided the possibility to work on this subject and for their financial support.

Espoo, 6.5.2015

Tero Paloheimo

Contents

Acronyms

Symbols

3GPP 3rd Generation Partnership Project.

A

AAA Authentication, Authorization and Accounting.
AQM Active Queue Management.
AuC Authentication Centre.
AVP Attribute-Value Pair.

D

DIME Diameter Maintenance and Extensions.
DOIC Diameter Overload Indication Conveyance.
DRA Diameter Relay Agent.
DTLS Datagram Transport Layer Security.

E

E-UTRAN Evolved Universal Terrestrial Radio Access Network.
EIR Equipment Identity Register.
ELP EPC LCS Protocol.
EPC Evolved Packet Core.
EPS Evolved Packet System.

F

FQDN Fully Qualified Domain Name.

G

GERAN GSM EDGE Radio Access Network.
GGSN Gateway GPRS Support Node.
GMLC Gateway Mobile Location Centre.
GMSC Gateway Mobile Switching Centre.

H

HLR Home Location Register.
HSS Home Subscriber Server.

I

IANA Internet Assigned Numbers Authority.
IE Information Element.
IETF Internet Engineering Task Force.
IMEI International Mobile Station Equipment Identity.

IMSI International Mobile Subscriber Identity.
IP Internet Protocol.
IPsec Internet Protocol Security.

J

JVM Java Virtual Machine.

L

LCS Location Services.
LTE Long Term Evolution.

M

ME Mobile Equipment.
MME Mobility Management Entity.
MSC Mobile Switching Centre.

N

NAI Network Access Identifier.

O

OLR Overload Report.

P

PCC Policy and Charging Control.
PCRF Policy and Charging Rules Function.
PDN Packet Data Network.
PDN-GW Packet Data Network GW.
PLMN Public Land Mobile Network.

Q

QoS Quality of Service.

R

RAN Radio Access Network.
RAT Radio Access Technology.
RED Random Early Detection.
RNC Radio Network Controller.
RNS Radio Network Subsystem.

S

S-GW Serving GW.
SCTP Stream Control Transport Protocol.

SGSN Serving GPRS Support Node.

T

TAU Tracking Area Update.
TCP Transmission Control Protocol.
TLS Transport Layer Security.

U

UE User Equipment.
UMTS Universal Mobile Telecommunications System.
UTRAN Universal Terrestrial Radio Access Network.

V

VLR Visitor Location Register.

1 Introduction

Mobile networks are a crucial part of the modern society and many functions rely on the network working and being reliable. There are many phenomena and events which may disrupt the network, such as, attacks on physical infrastructure, deliberate sabotage, and hacking. Ignoring possible malicious actors, many unintended and seemingly harmless events can cause various problems. One possibility for such an event arises when many people gather at one place and use their mobile phones and other connected devices. Given a large amount of people, only using one's own device can be enough bring down a network. Another possibility for issues is after a network restart since much signaling traffic is sent during network attachment [21]. These are only a few examples of situations where problems may occur and measures are thus needed to create reliable and robust networks having high availability. Such mechanisms must be implemented in different parts of a network to provide a defense in depth type of robustness. Physical security can be achieved by securing of installation sites, e.g. with proper locking. System security requires different approaches to various parts of a network.

Mobile networks are generally divided into two main parts: the radio part called the Radio Access Network (RAN) and the core network [22]. The radio part handles the communication with users' equipment, resource management, and related tasks. The core network, for example communicates with external networks, stores user data, and enables the mobility management needed for user roaming [22]. Both of these parts have entities, which handle the tasks in the network. Some important entities in a 4G network are the eNodeB for the RAN and core network entities called the Mobility Management Entity (MME) and Home Subscriber Server (HSS). The core network entities are mainly responsible for mobility management and storing of user data [21]. Many operations, e.g. location updates and user data management, needed to keep the network operational are dependent on the HSS being operational. In case it were unavailable, the network could be completely unusable or have a significantly reduced capacity. The reason for this is that many signaling and control operations need to be completed before sending of user traffic is possible and without a working HSS these cannot be done. Phenomena leading to a HSS being overloaded or completely non-operational are e.g. a massive traffic spike from the radio network due to a failed base station or an erroneous configuration somewhere in the network [6]. It is however possible to have several HSSs and they can be configured in different topologies as well use direct and indirect connections [19]. Having several HSSs may help in reducing the adverse effects caused by an overload, but it does not help if the absolute volume of the incoming traffic exceeds the capacity of the system. This shows that a solution to handle an overload of a HSS is clearly needed.

The communication between the MME and HSS entities is done through Diameter, which is an so called AAA protocol. Diameter is an evolution of the earlier RADIUS protocol and features improvements like the use of a reliable transport protocol and mandatory encryption [20]. The development and standardization of the protocol is done by the Internet Engineering Task Force (IETF). Communication parties are called nodes, which may either be clients, servers or agents, send

messages between each other in a peer-to-peer manner. Diameter is divided into a base protocol and applications. The base contains commands for common tasks like connection setup and teardown. Applications are extensions to the base protocol for implementing domain specific functionality, like the S6a interface for communication between the MME and HSS [1, 15].

A few solutions have been proposed by organisations as IETF and 3rd Generation Partnership Project (3GPP). Their solutions have some merit, but are usually not completely sufficient and do not provide a complete solution. A base solution is presented in the Diameter Overload Indication Conveyance (DOIC) Internet-draft [19], but it only is a "framework" for transmission of overload information in Diameter answer messages. The draft offers a method for one node (the reporting node) to inform another node (the reacting node) about its overload status through an Overload Report (OLR) [19]. The most important information in an OLR is the requested traffic reduction and the duration of the reduction [15]. Also included is management data needed to maintain overload state. DOIC does not specify how the reacting node should implement the requested traffic reduction [15]. This allows implementers to add their own logic and create their own methods according to specific needs and requirements. In the context of this thesis, the reporting node is a HSS and the reacting node is a MME. The HSS needs to keep track of its physical resource status, so it can inform a MME about overloads if they happen. When one occurs, the MME will reduce its traffic sent to the HSS to allow it to come out of overload [19]. It may happen that there is still too much traffic despite the reduction done by a MME. In that case the HSS may request further reduction until the point where no traffic is sent [19]. The signaling to indicate the end of an overload may happen implicitly or explicitly. In explicit signaling, the end transmitted through a special value sent in an answer message [19]. The implicit ending happens by the letting the OLR expire by waiting it to time out. As MMEs and HSSs can be configured in several ways and have different options for message routing, the overload control solution must also take these into account and be able to work in multiple modes and configurations [19].

This thesis seeks to develop a novel solution for handling Diameter traffic overload between the MME and the HSS. The solution is based on the groundwork laid out in the DOIC Internet-draft [19] and should maximize the performance of HSSs while trying to minimize issues caused by overloads. The development process consists of designing and implementing the solution, simulation and evaluation of the acquired results. The proposed design is conveyed in pseudocode and flow charts, along with further description about the details. A simulation of the solution is implemented in Clojure and uses the DESMO-J discrete event simulation library. The results of several simulations are evaluated according to specifically selected performance indicators, which reflect the goals of the solution.

The thesis is laid out as follows. Chapter 2 describes the Diameter protocol, mobile networks and their common usage. Chapter 3 describes the solution, its implementation and evaluation criteria. The results of the simulation are presented in Chapter 4. Discussion, conclusions as well as further work are discussed in Chapters 5 and 6.

2 Background

This section presents previous work related to the Diameter protocol and overload control mechanisms in mobile networks.

2.1 Diameter

Diameter is an Authentication, Authorization and Accounting (AAA) protocol specified by the IETF in RFC 6733 [15]. It is based on the older RADIUS AAA protocol [20]. Some major new features in Diameter are the use of reliable transport protocols and integrated security [15].

AAA comprises of authentication, authorization, and accounting. Authentication is the verification of the identity of a user, authorization is checking if a user should have access to a resource and accounting is the collection of information related to the use of a resource [15]. In a mobile network context, authentication would be the verification of the identity of a User Equipment (UE) by the HSS. In the same context, authorization is verifying whether the UE should be allowed to attach to the network. Finally, accounting would be monitoring the length of telephone calls or following data usage for billing purposes.

Diameter consists of the base protocol and extensions that are called applications. The base protocol must be supported by all Diameter nodes, while support for specific applications is only required for the nodes which are concerned with processing the specific application. All applications must have an Application Id, which is assigned by Internet Assigned Numbers Authority (IANA). The only exception to this rule is the base protocol, which does not need an ID, because it must be supported by all nodes [15].

The communicating parties in Diameter are called, nodes and may be clients, servers, or agents (relay, proxy, redirect or translation). A client is a node which implements the base protocol and other necessary applications. A server is a node which serves requests sent by non-server nodes. An agent is a node which provides various services (relaying, proxy, redirection or translation). The different agent types are presented below [15]:

Relay Relay agents receive requests from other nodes and route them towards their final destination. The routing decisions are based on information found in the messages. Relay agents maintain transaction state but not session state.

Proxy Proxy agents are similar to relay agents but they additionally make policy decisions. Such decisions may be provisioning, state tracking or dropping of rejected traffic.

Relay A redirect agent provides address and routing information that allow a client or another agent to communicate directly with a server. This agent provides relay service to all Diameter applications and does not perform application level processing.

Translation A translation agent translates messages between two protocols, such as Diameter and RADIUS. This agent enables the co-existence of several AAA protocols.

Agents preserve transaction state for failover purposes, which means preserving the Hop-by-Hop identifier for arriving requests and switching it to node's local value when forwarding the request. When receiving the corresponding answer, the original value is restored before forwarding the answer [15].

Diameter communication is done in sessions between two communicating peers, thus the protocol is of peer-to-peer type. In a session, messages are sent between peers in a peer connection. Depending on the number of intermediate peers, there may be many active connections. Either Transmission Control Protocol (TCP) or Stream Control Transport Protocol (SCTP) is used as the transport protocol for the connections. Because both protocols are connection-oriented, a connection between two peers must be established before communication is possible. The default port number for TCP connections is 3868 and correspondingly port 5658 for SCTP [15].

Protocol security, which is provided by either Transport Layer Security (TLS), Datagram Transport Layer Security (DTLS) or Internet Protocol Security (IPsec), is mandatory with only support for TLS or DTLS being required. TLS is used in conjunction with TCP and DTLS is employed with SCTP. No unencrypted communication is allowed between peers [15].

A Diameter message consists of the header and optionally by, one or more Attribute-Value Pairs(AVPs). The message format is shown in Figure 1.

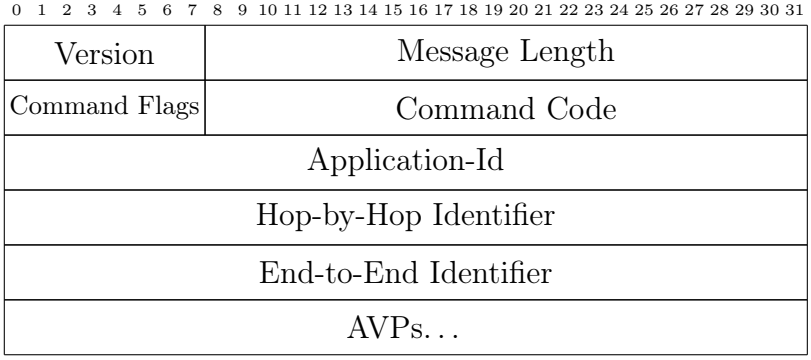

Figure 1: A typical Diameter message format [15].

The `Version` field denotes the Diameter version which must be set to one. The `Message Length` field tells the length of the whole message, including padding. Because of the padding, the length will always be a multiple of four. The following field, `Command Flags`, is a bit field with the following bits defined:

R If set, the message is a request, otherwise it is an answer.

P If set, the message may either be proxied, relayed or redirected.

E If set, the message contains an error.

T If set, the message may be retransmitted. If the same message is sent several times, this bit must be set.

The remaining bits are reserved for further use. The `Command Code` field contains the code of the current command. The `Application-Id` field denotes the ID of application to which the message is related. The `Hop-by-Hop Identifier` field helps in the matching of requests and answers and its value must be unique within one connection at any time. The answers must have the same value as the requests. Also, in case an answer with an unknown `Hop-by-Hop Identifier` is received, it must be discarded. The `End-to-End Identifier` is used to detect duplicate messages. It must be unique for every request and remain unique at least four minutes. The answer to a request must have the same value as the request. The last field in the message is a variable number of AVPs [15].

The payload in Diameter messages is carried in AVPs. The format of a AVP header is shown in Figure 2.

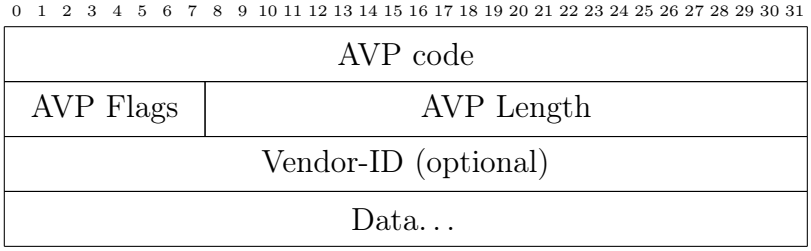

Figure 2: A typical AVP header format.

The first field in the header, `AVP Code`, is used to identify the attribute along with the `Vendor-ID` attribute. In the `AVP Flags` field, three bits are currently defined [15]:

V If this bit is set, the AVP contains the `Vendor-ID` field.

M If this bit is set, the receiver must parse and understand this AVP. If it cannot, it must respond with an error. Informal AVPs may be sent with this bit off. Then it is not mandatory for the receiver to parse and understand the AVP.

P This bit is intended for end-to-end security. There are no such mechanisms yet and thus this bit should be set to zero.

The remaining bits are reserved for future use. The length of whole AVP is in the `AVP Length` field. If the **V** bit is set, the AVP contains `Vendor-ID` field. The vendor IDs are managed by IANA. The final field contains the AVP (data) payload [15].

There are many AVPs defined in the base protocol but it is enlightening to describe some of the most commonly used ones [15]:

Session-Id This AVP contains a string that uniquely identifies the current session, additionally the same value must never occur again. It must remain the same for the duration of the whole session.

Origin-Host This AVP contains the Fully Qualified Domain Name (FQDN) of the host that sent the request. Because it is used to identify the request originator, it must not be modified by agents.

Origin-Realm A realm is an administrative domain [12]. It is the part of the Network Access Identifier (NAI) which is after the @ character. The NAI, defined in RFC 4282 [9], is the identifier from which the user's identity and realm are extracted. The Origin-Realm contains the realm of the sender of a message.

Destination-Host The unique identity of the final destination of a message. If this AVP is not specified, the message can be sent to any node in the realm supporting the message. When the destination is fixed, this AVP is used for routing. An example of this are server initiated messages to specific clients.

Destination-Realm An AVP which is used for routing purposes and contains the destination realm of a message.

User-Name This AVP contains name by which the user is known.

Several commands are defined in the Diameter base protocol. They are presented below:

Abort-Session-Request and -Answer This command is sent when the service across a Diameter session is to be aborted. The client responds to this request with the Abort-Session-Answer message.

Accounting-Request and -Answer A message that a Diameter node sends to a server containing accounting information. The message should contain AVPs that are accounting service-specific. Only a Diameter server may respond with an Accounting-Answer message.

Capabilities-Exchange-Request and -Answer A message sent after the establishment of the transport connection between two nodes to agree on common capabilities. Such capabilities are, for example, protocol version and supported applications. The other node responds with the capabilities supported by both nodes or notifies the request sender that there are no common applications.

Device-Watchdog-Request and -Answer The watchdog message works as a mechanism to detect transport failures. If an answer to a watchdog request is not received, it is concluded that there is a problem in the transport connection. If regular traffic is sent between two nodes, the lack of an answer message can be considered as the presence of a problem.

Disconnect-Peer-Request and -Answer When a Diameter node wants to disconnect the transport layer connection, it indicates its intent with this message. By sending this message, the node also informs that it does not want a connection to be re-established unless it is necessary.

Re-Authentication-Request and -Answer If an authentication has expired, a Diameter server uses this command to request that a user authenticates itself again.

Session-Termination-Request and -Answer When a Diameter client does not need the service provided by a server, it sends this message. After sending the answer message, the server must release the resources needed by the session.

Routing in Diameter is based on the *Destination-Host* and *Destination-Realm* AVPs. Depending on the request properties (proxy ability) and the values of the AVPs, different routing decisions are taken. If a request cannot be proxied, the *Destination-Host* and *Destination-Realm* AVPs must not be included. If the request is intended to any server in a specific realm, only the *Destination-Realm* AVP is included. If a specific server in a realm is the destination, both AVPs must be included. When a node on the way towards the final destination receives a message, it processes the message in the following way. If the request can be locally processed, it is processed and the answer is sent back to the originator. If the next hop is a node with which the local host can communicate directly, the request is forwarded towards the node. Two nodes can communicate directly if destination node is found in the source node's peer table [15].

Two tables are used in message processing and routing: the peer and routing tables. The peer table is used by the routing table in message forwarding. An entry in the table contains the following fields [15]:

Host Identity The identity of the host, which is extracted from the *Origin-Host* AVP.

Status Status of the node in the peer state machine.

Static or Dynamic Denotes whether the node was dynamically discovered or statically configured.

Expiration Time Dynamically discovered nodes have an expiration time after which they must either be refreshed or expired.

Connection type Specifies whether TCP with TLS or SCTP with DTLS is used for communication.

The routing table is used for realm-based routing lookups. Entries in the routing table contain the following fields [15]:

Realm Name The primary key of the table.

Application Identifier The Application Id enables the use of different routes depending on the application. This field is used as the secondary key of the table.

Local Action Tells how a message should be treated. The following values are possible:

LOCAL Messages can be processed locally and no routing is needed.

RELAY Messages must be routed to the next-hop Diameter node. The non-routing AVPs are not modified.

PROXY Messages must be routed to the next-hop Diameter node. Policy actions may be performed by adding new AVPs to the message.

REDIRECT Messages must have an address of a home Diameter server attached. The message is then returned to the sender.

Server Identifier Contains the identity of one or many servers where the message is to be routed. The same identity must also be found in **Host Identity** field of the peer table.

Static or Dynamic Denotes whether this entry was statically configured or dynamically discovered.

Expiration Time Contains a time after which a dynamically discovered entry expires.

It is possible to have a default route in the routing table that is used if no other entry is matched. The table may also consist of this entry only [15].

2.2 Universal Mobile Telecommunications System

The Universal Mobile Telecommunications System (UMTS) is a third generation (3G) mobile network developed by 3GPP [22] and it was initially specified in 3GPP Release 1999. UMTS consists of the RAN, that is, either Universal Terrestrial Radio Access Network (UTRAN) or GSM EDGE Radio Access Network (GERAN) and the core network. The UTRAN consists of base stations, NodeBs, and Radio Network Controller (RNC). The NodeB is an element communicating with UEs over the radio interface. A UE is the device used by a user to access the network's services and terminates the radio interface between a NodeB and itself [5]. It is subdivided into several domains each responsible for different functionalities. The RNC has several tasks: congestion control, traffic encryption and decryption, radio resource management amongst others. The NodeBs and RNCs form the Radio Network Subsystem (RNS). If needed, it is possible to have several RNSs in UTRAN. The UMTS network structure with two RNSs is shown in Figure 3 [8, 22].

The voice handling part of the UMTS core network consists of the Mobile Switching Centre (MSC), Gateway Mobile Switching Centre (GMSC), Equipment Identity Register (EIR), Home Location Register (HLR), and Visitor Location Register (VLR) elements. The MSC is responsible for signaling and switching of the (internal) voice calls between the RNS and the core network. The GMSC handles external voice calls, that is, calls which are mobile originated or terminated from external networks. The HLR contains all subscriber data, such as various numbers and identities and the location of the UE. The data of one subscriber is always stored in one HLR. Functions of the HLR form a subset of the functions of the HSS. The

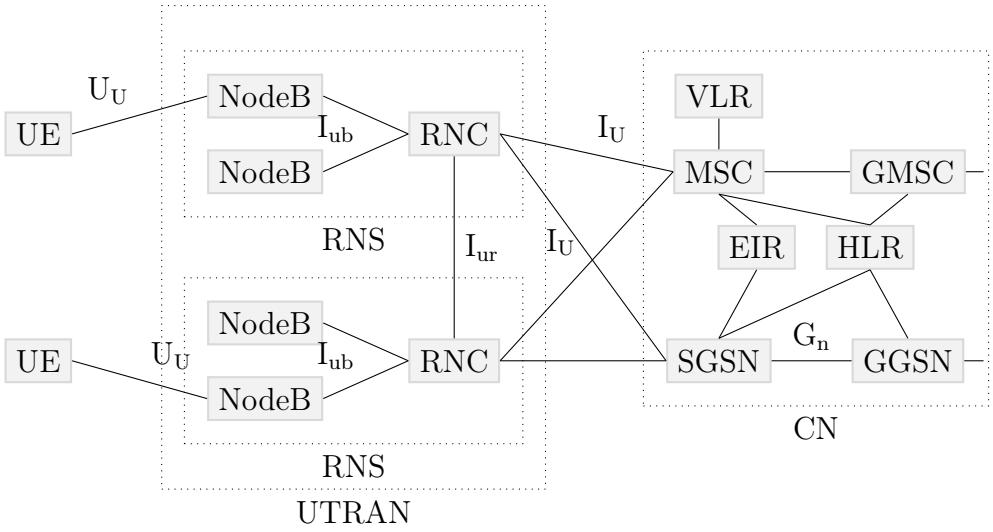

Figure 3: The UMTS network structure with interfaces from Release 99 [22].

VLR tracks the roaming users in the network and in order to enable call handling, several identities and numbers are stored for each roaming user. The EIR contains the identities of UEs. The identities may be classified in three different ways: white, grey or blacklisted. A UE may be denied access to the network based on this classification. It is important to note that in 3G circuit switching is used for voice, but in 4G everything is IP-based [5, 22].

The packet switched data connection part additionally uses the Serving GPRS Support Node (SGSN) and Gateway GPRS Support Node (GGSN) elements. The SGSN is responsible for IP address management, location tracking, collection of billing information, etc. It is the packet switched analogue to the MSC. The GGSN acts as the connection to external packet data networks such as the Internet. The element performs, amongst others, routing and traffic tunnelling [5, 22].

2.3 Evolved Packet System

The Evolved Packet System (EPS) is a fourth generation (4G) mobile network developed by 3GPP. The original specification for EPS appeared in 3GPP Release 8 in December 2008. The network consists of the RAN, which is either Long Term Evolution (LTE) or Evolved Universal Terrestrial Radio Access Network (E-UTRAN) for EPS and UTRAN for UMTS, and the core network, Evolved Packet Core (EPC). The packet core consists of the following entities: MME, HSS, Serving GW (S-GW) and Packet Data Network GW (PDN-GW). The network architecture is presented in Figure 4. The solid and dashed lines in Figure 4 respectively present user and control plane data. The key feature which differentiates EPS from previous networks is that all traffic is packet switched, using the Internet Protocol (IP). Previous systems like UMTS use circuit switching for voice transport [21].

The eNodeB element provides the radio interface for LTE, which includes re-

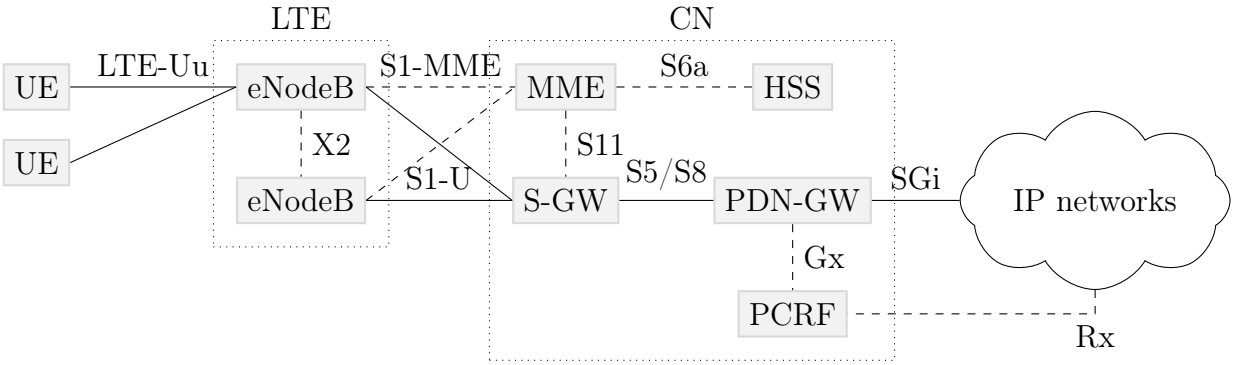

Figure 4: A simplified EPS architecture with control and user plane interfaces [21].

source management, admission control, and scheduling of down- and uplink radio channels. Additionally IP header compression and traffic encryption is done by the element. eNodeBs are connected between each other over the X2 interface, which for example is needed during handovers. The S1 interface, which is divided into the user plane (S1-U) and the control plane (S1-MME), connects the eNodeB to the MME or the S-GW [21].

The Serving GW handles the forwarding of the data packets from the UE and works as a mobility anchor point between handovers when they are required. It also stores some data for the UE. For UEs in idle mode, the S-GW pages the UE when a packet arrives. The PDN-GW acts a gateway to external Packet Data Networks(PDNs) and provides connectivity to them. One UE may be connected to several PDN-GWs if it needs access to several PDNs. The IP addresses for the UEs are also allocated by this element. The gateway role of the element entails the possibility to do packet filtering or marking for Quality of Service (QoS) change purposes [21].

The Policy and Charging Rules Function (PCRF) is responsible for the Policy and Charging Control (PCC) in the EPC. PCC enables flow-based charging, like credit control, as well as policy control functions. The PCRF is the point where the policy control decisions are made. A policy, in the 3GPP context, is a rule for selecting what kind of treatment an IP flow will receive. A prerequisite for all functions of PCC is traffic classification that is done by the PDN-GW or S-GW [12, 21].

2.3.1 Mobility Management Entity

The MME is the main entity for controlling the LTE access network. The entity selects the S-GW for a UE on initial attachment and handover between LTE networks, if needed. Tracking and paging of idle UEs as well as activation and deactivation of radio bearers are also performed by the MME. The control plane functions of mobility between LTE and 2G/3G networks are also provided by the entity [21].

Authorization to use an operator's network and enforcement of possible roaming

restrictions are performed by MME. The authentication of end-users is done in co-operation between MME and HSS [21].

2.3.2 Home Subscriber Server

A HSS is the master database for user data. It also assists in user security (keys and credentials), mobility management, access authorization for roaming and service deployment. Several HSSs may be used for the sake of redundancy if the capacity of one HSS is not sufficient to handle all subscribers [21].

The HSS is a combination of the HLR and Authentication Centre (AuC) entities. The HLR is responsible for storing the user profile and various identities and numbers, e.g. International Mobile Subscriber Identity (IMSI). The AuC stores the identity key of each subscribed user and uses this key to generate the security data needed to provide the confidentiality and integrity for the communication between a UE and the network. When a network entity like the MME needs authentication services, they are requested from the HLR which in turn instructs the AuC to provide the services [5].

2.4 Diameter usage in the Evolved Packet Core

This subsection describes how MMEs and HSSs can be configured in a network and what Diameter procedures are performed. Additionally, some use cases of these procedures are presented.

2.4.1 MME and HSS configurations

The MME and HSS can exist in several configurations. The three most common configurations are a direct connection between the MME and HSS, the second with a Diameter Relay Agent (DRA) between the two entities and finally a hybrid model of the two previous configurations. In all the previous cases, there are usually many MMEs, HSSs, and DRAs. The exact number depends on the capacity of the entities, the size and subscriber count of the network. Below, the different configurations are presented and discussed from a load point of view [15, 19].

Perhaps the most simple configuration is a direct connection between the MME and HSS. Depending on the number of MME and HSS entities, there may be many connections between the entities. A configuration with one MME and two HSSs can be seen in Figure 5.

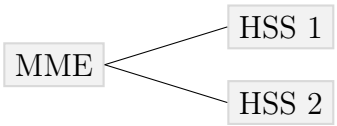

Figure 5: A simple MME and HSS configuration.

The overload situation for this configuration depends on the assumption whether the two HSSs are replicated or not. In the first case an overload could mean that

answers to MME originated requests would be delayed or completely missing depending on the replication type. Further requests would cause additional load on the HSS and make the overload condition even more worse. The resolution to the situation is to reduce the amount of offered requests to the HSS. The second case with non-replicated HSSs has a new overload scenario where only a subset of the servers are overloaded. A new resolution option is to send the requests to another HSS if the subscriber data is available on it. If it is not, the number of requests towards the overloaded HSS must be reduced to reduce the overload.

The second configuration with a DRA between the MMEs and HSS provides certain advantages compared to the previous configuration. This configuration is presented in Figure 6. When the MME sends a request to the DRA, it may select the HSS to which the request is sent. In this way, sending requests to overloaded servers can be avoided. Despite its advantages the configuration has some downsides: the agent or all the servers behind the agent may become overloaded. If the agent is overloaded, it may not be able to route all requests towards the servers. Several agents may of course be deployed providing either additional capacity or fail-over support by having inactive spare agents. If many agents are run simultaneously and they have unequal loads, less traffic can be redirected to the overloaded agent. In the case of overload with equal load on all agents, the mitigation is to reduce the total load. In fail-over mode where the primary agent is overloaded, excess traffic can be sent to the spare agent. Again if both agents are overloaded, the total rate of traffic sent to both agents must be decreased. Server overload is similar to the first configuration and an agent can only react to it by limiting the rate of sent requests [13].

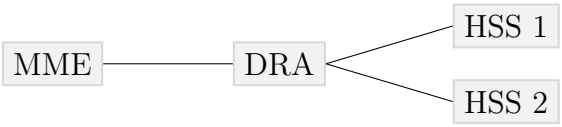

Figure 6: A MME and HSS configuration with a DRA.

The final configuration is a hybrid of the previous two and it is shown in Figure 7. As can be seen from the Figure, HSS 1 and HSS 2 can redundantly be reached directly and through the DRA. The choice of which connection to use may depend on several factors, e.g. the overload status of the DRA and overload mitigation support of the HSSs. This configuration has similar overload properties as the two previous ones. For example, if the DRA is overloaded, all the entities behind it are affected. The same mitigation mechanisms also apply, e.g. redirection of traffic to the less loaded entity or reduction of the total rate of traffic.

2.4.2 MME and HSS Diameter procedures

The MME and HSS are connected to each other through the S6a interface. The different procedures can be classified according to their functionality: authentication, location management, subscriber data handling, notification, and fault recovery [1].

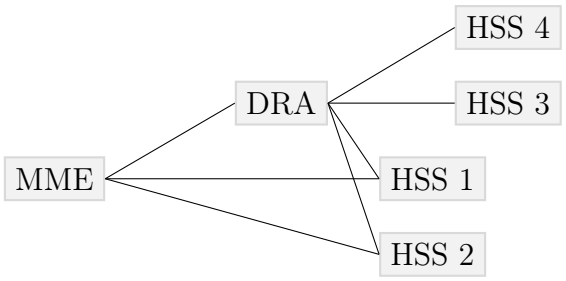

Figure 7: A hybrid configuration with direct connections and via a DRA.

Authentication procedures are used by the MME to fetch authentication information from the HSS. The MME does this with the *Authentication-Information-Request* (AIR) command and the HSS responds with the *Authentication-Information-Answer* (AIA) command. The only mandatory Information Elements(IEs) in AIR are the IMSI of the user, whose information is requested and the ID of the visited Public Land Mobile Network (PLMN). An IE can be seen as a parameter or a piece of information, which is included in the Diameter message and is either optional or mandatory. The value of the IE is mapped to a Diameter AVP, e.g. the IMSI IE is mapped to the *User-Name* AVP. The answer message must include the result of the request. For this the `Result-Code` or the *Experimental-Result* is used. S6a errors are communicated through the latter AVP and may have the following values: `User Unknown`, `Unknown EPS Subscription` and `Authentication Data Unavailable`. The authentication vectors are sent in the *Authentication-Info* AVP [1].

Location management procedures are concerned with maintaining information about which MME is serving the user and optionally updating the user subscription data in the MME. The *Update-Location-Request* (ULR) command is used by the MME to update the identity of the current MME to the HSS. The mandatory information elements in this command are: the user IMSI, ULR flags, the ID of the visited PLMN and Radio Access Technology (RAT) type. The flags amongst other contain options to indicate on which interface (S6a or S6d) this request is sent, skipping the sending of subscription data, and whether the requesting node is a combined MME/SGSN unit. Upon reception of a ULR command, the HSS conducts a series of checks, e.g. ensuring that there is subscription data for the given IMSI. If there are no errors, the HSS should send a *Cancel-Location-Request* command to the previous MME, if there is one, and replace the old MME identity with a new identity. The *User-Location-Answer* (ULA) command includes the result as the only mandatory AVP. ULA flags and subscription data are optional, the subscription data is included unless its sending was explicitly prevented in the ULR flags [1].

Sometimes, the removal of a subscriber record from a MME is needed. The removal need arises when a subscriber's subscription is removed from a HSS, an update procedure is done, that is, the subscriber is moved to another MME or when there is an initial attachment. In the last case, no subscription data is removed. This procedure is implemented in the *Cancel-Location-Request* (CLR) and *Cancel-Location-Answer* (CLA) commands. The CLR command is sent from the HSS to the MME and the answer in the reverse direction. Mandatory information elements

in the request are user IMSI and cancellation type. Possible MME related values for the latter element are `MME-Update Procedure`, `Subscription Withdrawal`, and `Initial Attach Procedure.` The answer command contains only the result as the mandatory information element. Upon reception of a CLR command, the MME is checks if the IMSI is found. If it is not, the success result is also sent [1].

Due to a management command or automatically, e.g. long UE inactivity, a subscriber's subscription data is removed from a MME. This information is sent to the HSS with the *Purge-UE-Request* (PUR) and *Purge-UE-Answer* (PUA) commands. The user IMSI is the only mandatory information element in the PUR command. The PUA command has the result as the only mandatory information element. Possible S6a errors, currently only `User Unknown`, are signaled with the *Experimental-Result* AVP. When the HSS receives a PUR command, it checks whether the IMSI is known and returns an error if it is not known [1].

Subscriber data handling is tasked with inserting, updating, and removal of subscriber data in the MME. Insertion and updates are done with the *Insert-Subscriber-Data-Request* (IDR) and *Insert-Subscriber-Data-Answer* (IDA) commands and deletion with the *Delete-Subscriber-Data-Request* (DSR) and *Delete-Subscriber-Data-Answer* (DSA) commands. The HSS sends the IDR command to the MME when subscriber data needs to be inserted or updated in the MME. Mandatory information elements in the IDR command are user IMSI and the subscriber profile that will be inserted or used to update existing data. It is also possible to update only a part of the subscription with the IDR command. The only mandatory information element in the IDA command is the result. If the MME does not know the received IMSI, it sends the `User Unknown` error as the result [1].

User data deletion is, for example, needed when a user is moved from a MME to another and the subscription data is no longer needed in the old MME. The DSR request is sent by the HSS to MME. Mandatory information elements are the user IMSI and a flag bit mask. This bit mask is used to specify what data is to be removed. The DSA command has the result as the only mandatory information element. Upon receiving a DSR command, the MME checks whether the IMSI is known. If not, the `User Unknown` error is sent as the result. Otherwise the requested data is removed and if it was done successfully, success is sent. In the case of an error, the `Result-Code` AVP `DIAMETER_UNABLE_TO_COMPLY` error value is sent as the result [1].

In case a failure occurs at the HSS and it is restarted, the Reset procedure is used to inform the MME about this event. The *Reset-Request* (RSR) and *Reset-Answer* (RSA) commands implement this procedure. The RSR command contains no mandatory information elements, but a list of user Ids may be included. This list shall include the leading digits of the IMSIs of the affected subscribers. The RSA command contains result of the operation as the only mandatory information element. The HSS shall send this command to relevant MMEs, when the HSS may have lost MME identities of some subscribers served by the MME. This must be done because the HSS cannot send *Cancel-Location-Request* or *Insert-Subscriber-Data* commands if the MME identity is not known. Upon reception of a RSR command, the MME shall determine which subscriber records are impacted and

mark the location information in HSS as not confirmed. During the next radio contact to an affected UE, the restoration procedure shall be done [1].

A HSS may ask a MME to notify it about certain events, for example, an update of terminal information or that a UE is reachable. These notifications are sent with the *Notify-Request* (NOR) and *Notify-Answer* (NOA) commands. The only mandatory information element in the NOR command is the user IMSI. The remaining information elements are either complementary or optional and depend on the type of the requested notification. The only mandatory information element in the NOA command is the result to indicate success or failure. The S6a error `User Unknown` may be returned if the user is not known. When the HSS receives a NOR command it shall check if the IMSI is known, if not the `User Unknown` error must be sent. It must also check that the MME is registered to the HSS if the IMSI is known. Again, if not, an error is sent, in this case `DIAMETER_ERROR_UNKNOWN_SERVING_NODE`. Finally, in case of no errors, the result answer is to be sent and an action corresponding to the request is performed [1].

2.4.3 Other Diameter procedures

Other procedures in EPC also use Diameter; amongst them are for example equipment identity checking and location services.

During certain operations such as attach, the MME needs to check the identity status of the Mobile Equipment (ME) from the EIR. This action is performed over the S13 interface and mapped to the *ME-Identity-Check-Request* (ECR) and *ME-Identity-Check-Answer* (ECA) commands. The mandatory information element is information about the terminal in question, such as, the International Mobile Station Equipment Identity (IMEI). The IMSI may be optionally sent. The answer command contains the result of the request and the equipment status, in case of a successful request. In addition to the standard Diameter errors, the S13 error about unknown equipment (`DIAMETER_ERROR_ EQUIPMENT_UNKNOWN`) may be sent. The equipment may either be white listed, blacklisted or grey listed. Allowed equipment is placed on the white list and barred equipment on the blacklist. Grey listed equipment is usually not barred, but tracked by the network [1, 3].

The EPC LCS Protocol (ELP) provides Location Services (LCS) over the SLg interface between the MME and the Gateway Mobile Location Centre (GMLC). The two main operations in LCS are location requests from the GMLC to the MME and location reports from the MME to the GMLC. In the first one, the GMLC starts by sending a PROVIDE SUBSCRIBER LOCATION REQEUEST message to the MME through the *Provide-Location-Request* (PLR) command to which the MME replies with a *Provide-Location-Answer* (PLA) command. The PLR must contain the location type and information about the LCS client. Also, one way of identifying the user's UE must be included, either IMSI, MSISDN or IMEI. Further information elements may be included specifying e.g. QoS. The PLA answer command contains the result of the request and the requested location data [4].

The MME may implicitly provide the location of a UE by sending the *Location-Report-Request* (LRR) to the GMLC and it acknowledges the request with the

Location-Report-Answer (LRA) command. The LRR must contain the cause which triggered the location event. The remaining information elements are the same as in the PLR command. The LRA command contains the result of the request [4].

The Diameter request commands of the S6a, S13 and SLg interface are summarized in Table 1. It contains the name, the direction of requests, and a short description of every command. Table 1 does not contain answer commands, e.g. *Authentication-Information-Answer*, which are sent in the opposite direction to the request commands [1].

Table 1: Summary of the S6a, S13, and SLg interface request commands.

Name	**Direction**	**Description**
Authentication-Information-Request (AIR)	MME → HSS	Used for requesting authentication vectors
Update-Location-Request (ULR)	MME → HSS	Used for updating the identity of MME currently serving the user
Cancel-Location-Request (CLR)	HSS → MME	Used for removing a subscriber record from the MME
Purge-UE-Request (PUR)	MME → HSS	Used to indicate that a subscriber's profile has been removed from the MME
Insert-Subscriber-Data-Request (IDR)	HSS → MME	Used for inserting or replace user data in the MME
Delete-Subscriber-Data-Request (DSR)	HSS → MME	Used to remove subscription data from the MME
Reset-Request (RSR)	HSS → MME	Used to inform a MME about a HSS restart
Notify-Request (NOR)	MME → HSS	Used for notification of an event
ME-Identity-Check-Request (ECR)	MME → EIR	Used to check equipment identity status
Provide-Location-Request (PLR)	GMLC → MME	Used to request the location of a UE
Location-Report-Request (LRR)	MME → GMLC	Used to implicitly report the location of a UE

2.4.4 Use case examples

To make the use of these procedures more clear, a few cases of different network operations are presented. The first case is an initial attach.

An initial attach, shown in Figure 8, is done, when a UE needs services requiring registration, from the network. The UE starts by sending an attach request to the eNodeB, which is then sent to the MME. The MME will fetch authentication data for the UE by sending the AIR command to the HSS. The identity status of the mobile equipment may also be checked and it is done with the ECR command sent to the EIR. It will respond with the identity status and the authentication procedure

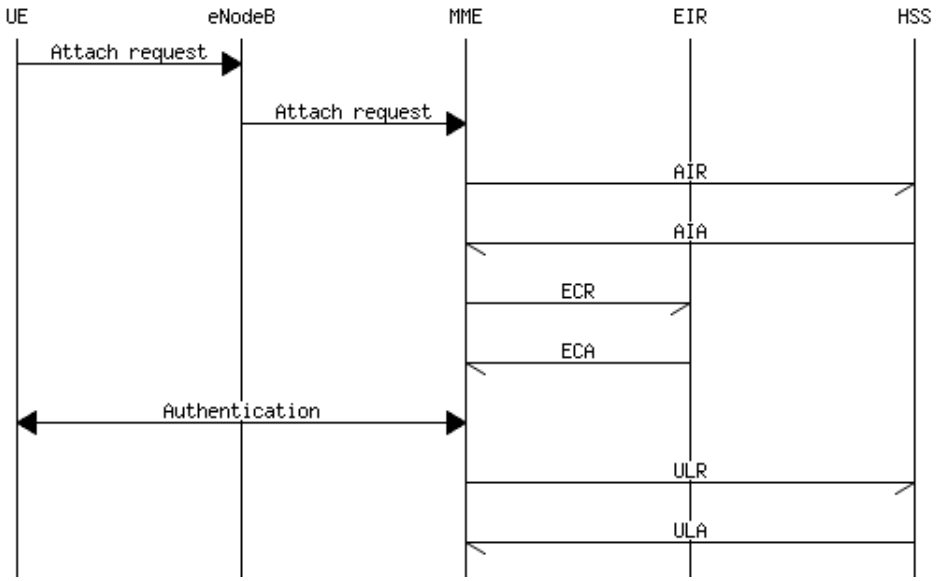

Figure 8: Procedures performed during an attach [2].

itself is performed. In case the MME is different from the previously used MME or there is no valid subscription context, the ULR command is sent to the HSS to inform that the current MME is now serving the UE. The HSS will now send the CLR command to the old MME to inform that it is no longer serving the UE. The old MME acknowledges this with the CLA command. Finally the ULR command from the new MME is acknowledged with a ULA command from the HSS [2]. The attach in Figure 8 does not show the situation with a valid subscription context.

A Tracking Area Update (TAU) is performed e.g. when the UE enters a new tracking area, which is not in the list of registered tracking areas or the TAU timer has expired. The message flow of a TAU can be viewed in Figure 9 [2].

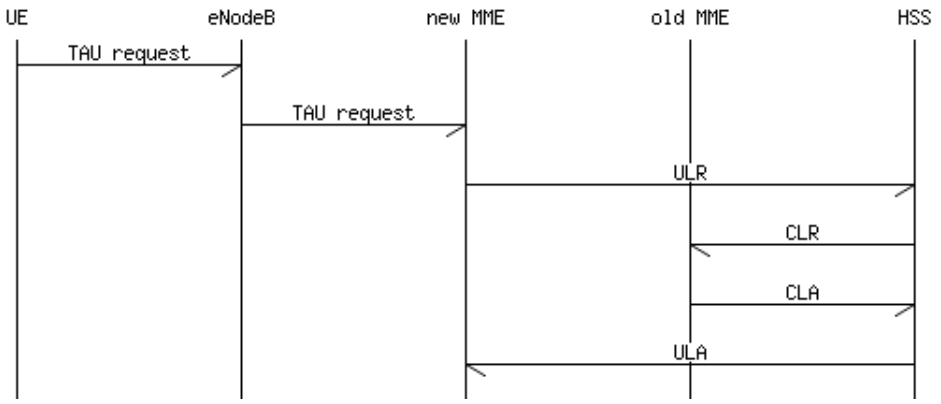

Figure 9: Procedures performed during a TAU [2].

The part of a TAU which concerns Diameter is authentication and maintaining the information about the MME currently serving the UE. As in the initial attach,

the (new) MME sends an ULR command to the HSS informing that it will be serving the user. The HSS will then send a CLR command to the old MME so that it can stop serving the user and the MME responds with the CLA command. Finally, the ULR command is acknowledged by the HSS with an ULA command. The previously described exchange is done only if the MME changes, that is, an inter-MME TAU is done. If an intra-MME TAU is done, only the authentication part is performed [2].

Diameter messages are also used in the HSS-initiated detach procedure. It is used if the operator wants to remove the subscriber's mobility management and EPS bearers at the MME. This procedure is shown in Figure 10 and uses the CLR and CLA commands between the HSS and the MME. The HSS sends the CLR command to the MME to inform it about the detach. The MME will send the detach request to the UE and also acknowledge the request from the HSS with the CLA command. This procedure is only used if the cause in the CLR is `Subscription Withdrawn` [2].

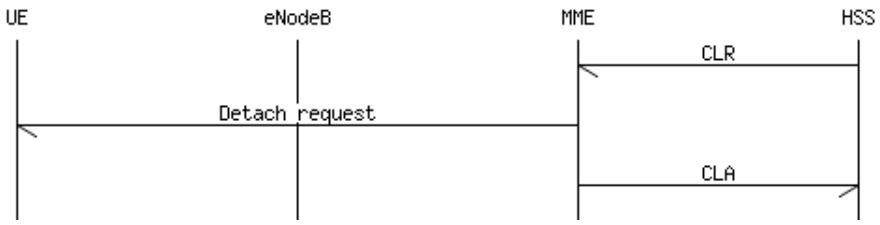

Figure 10: The HSS-initiated detach procedure [2].

2.5 Overload control

A single HSS can be seen as a $G/G/1$ queueing system, which has general arrival and service time distributions and one server. This model is applicable because it is generally impossible to say according to which arrival process Diameter messages arrive and what the service time distribution is. Depending on the characteristics of HSS and the Diameter message processing, the $G/D/1$ model could be used to describe a HSS because the service times may be considered constant within certain bounds. Multiple HSSs may be modelled as $G/G/n$ or $G/D/n$ systems, where n is the number of available entities. A problem with these models is that not very many statistical properties (e.g. queue length and queueing time) are not known for them [10]. For easier theoretical analysis, a $M/M/n/n$ system, having a Markovian arrival process and an exponential service time with n servers and a queue of capacity n, is later used in evaluation of the designed solution. Results given by this model are not so accurate, but still provide some insight on the system performance.

A general queueing system with a predefined service rate μ (services per time unit) and arrival rate λ (arrivals per time unit). The load or utilization of the system is defined as $\rho = \lambda/\mu$ and if $\rho \geq 1$, the system is considered unstable and also overloaded. The aim is generally to operate the system in a way that it will not be overloaded [11].

The main methods to mitigate overload are load balancing and throttling. In load balancing, several elements are used to serve incoming requests. This method requires that several elements are available and that traffic can be directed towards a specific element [7].

Throttling packets is the second alternative to handle overload, in which a throttling scheme decides which new messages are dropped. In this method, overload information from the target element is needed to select the correct throttling parameters. The behaviour of the system can be modified by adjusting the throttling scheme e.g. not to drop messages related to previous events or using a priority to only drop arrivals with a smaller priority. Also Active Queue Management (AQM) techniques such as Random Early Detection (RED)[16] and its modifications may be used but their performance varies depending on the overload status [7, 18].

In order to determine the system load, several measures can be used, e.g. processor occupancy, arrival or acceptance or queue length. As shown in [17, 18], only using the processor occupancy as measure does not provide as good results as using other measures [18].

An overload control solution may either use only local information or local and remote information. If only local information are used, no external parties need to be involved, thus simplifying the design and implementation. On the other hand, not including remote information limits the possible mitigation methods basically to only throttling. The advantage of using remote information is that during overload the source of the traffic can possibly be directed to decrease its traffic towards the overloaded system. The downside is that the changes in the system's state need to be communicated to all involved parties.

2.6 Current and proposed Diameter overload control mechanisms

There are currently two mechanisms by which an overload can be indicated. The first one is to send the `DIAMETER_TOO_BUSY` protocol error in the `Result-Code` AVP of the answer message. When receiving this message, the Diameter node which originated the message should try sending it to another node. This error message does not provide any details about the severity of the overload or when it possibly ends [15, 7].

The second mechanism is to close the transport connection with the *Disconnect-Peer-Request* command with the `BUSY` cause. This mechanism causes additional confusion if the Diameter client and server are communicating through an agent. The client will then receive a `DIAMETER_UNABLE_TO_DELIVER` error, which does not directly tell anything about the cause of the error. After receiving this error, the client may try to reopen the connection, thus making the overload even worse [15, 7].

An indirect way for a node to infer about an overload state is to note that no command answers are received even though the transport connection is working. This is a very unreliable and potentially slow method of overload detection [7].

Some further solutions suggested by the 3GPP are for example signaling optimization, rejecting traffic already in the RAN, and improving user location reporting [6]. These suggestions are particularly aimed at handling and prevention of the core network overload.

The IETF Diameter Maintenance and Extensions (DIME) working group has issued the DOIC Internet-draft [19]. The draft presents a mechanism for conveying overload information from a reporting node to the reacting node. The mechanism is based on piggyback sending of overload information in the Diameter answer message. The protocol application which the overload information concerns is inferred from the command [19].

The announcement of DOIC features is done with the `OC-Supported-Features` AVP. The `OC-Sequence-Number` contains a sequence number used to indicate a change in the `OC-Feature-Vector` AVP, which contains the supported DOIC capabilities of a node. The default loss algorithm for traffic abatement must be supported by all DOIC capable nodes. The capability negotiation is initiated by the reacting node as it sends its supported capabilities to the reporting node. The reporting node then replies in an answer message with its supported capabilities. If there are no common capabilities, the DOIC features are not considered implemented and AVPs are not sent to the reacting node [19].

There may be situations where Diameter clients or servers do not support DOIC. In such situations a Diameter agent may act as a reacting node. This means adding the `OC-Supported-Features` AVP when it is not included in a request and stripping any overload control related AVPs from the answer for relaying it forward. The agent will also perform overload control for non-supporting nodes when required by the current overload status and stored overload reports [19].

The `OC-OLR` grouped AVP is used to convey the overload report. The AVP contains the following AVPs [19]:

OC-Sequence-Number This AVP indicates when this particular `OC-OLR` AVP was first sent. Because the same AVP may be replayed, this AVP indicates the freshness of the `OC-OLR` AVP. When a reporting AVP with new content is sent, this sequence number must be greater than the previous value.

OC-Report-Type This AVP denotes the target of the overload report. Currently two values are defined: zero for a host (reacting node) and one for a realm.

OC-Reduction-Percentage Contains the requested amount of traffic decrease based on the current rate. The value 100 means that no traffic is to be sent and zero indicates that no decrease is required. Sending a value zero after non-zero traffic reduction is considered the end of an overload state.

OC-Validity-Duration Indicates the time how long the AVP and its content is valid. The time is in seconds and allowed values are between zero and 86400 (24 h) with a default value of five seconds. It is recommended that overload reports are never let to expire through a timeout. Instead explicit notification of the end of the overload state should be sent.

One answer message may contain many `OC-OLR` AVPs, but each AVP must have a distinct `OC-Report-Type` AVP value. They must be distinct to make distinguishing of the different overload types possible. A benefit of having several `OC-OLR` AVPs in one message is the possibility to signal multiple overloads in one answer message, thus avoiding the need distribute them over several answer messages [19].

An extension to DOIC is defined in the Diameter Overload Rate Control Internet-draft. It defines a new overload control algorithm using absolute values (request rate) instead of relative values. In this new algorithm, the reporting node defines a maximum request rate instead of reduction percentage. The support for this algorithm is defined by adding the *OLR_RATE_ALGORITHM* flag to the `OC-Feature-Vector` AVP in the `OC-Supported-Features` AVP [14].

The new `OC-Maximum-Rate` AVP in the `OC-OLR` AVP conveys the new rate (requests per second). A rate of zero means that no requests are to be sent to this node. The reporting node must continuously monitor its overload state and adjust its maximum rate accordingly. Also, the node should allocate a proportion of the available rate to all reacting nodes [14].

If a configuration with one or many agents is used, it is possible that an agent becomes overloaded. Thus an agent overload abatement mechanism is needed, which is implemented in an extension defined in the Diameter Agent Overload Internet-draft. The main idea is to react to the situation as close to the overloaded node as possible, usually the adjacent node. To realize this, the DOIC specification has been augmented with some features. The new *OLR_PEER_REPORT* flag has been added to the `OC-Supported-Features` AVP to indicate support for the peer overload report type. A new AVP is also added, the `OC-Peer-Node` AVP contains the Diameter identity of the reporting node. This AVP is used during capability advertisement to infer, whether the node from which a message was received supports this extension. The `OC-Reporting-Node` AVP contains the identity of the reporting node and the value is checked when a reacting node receives an overload report. A new value has been defined in the `OC-Report-Type` AVP, the constant 2 denotes peer, meaning that overload treatment should be applied for requests towards the identified peer [13].

When an agent is overloaded, it sends an overload report to the node originating the traffic. Upon receiving the report, the reacting agent must check that it came from an adjacent peer. This is done by comparing the value of the `OC-Reporting-Node` AVP with identity in the message. In case the values match, the requested traffic reduction should be performed. The preferred way is to re-route traffic to a node that is not overloaded. When this is not possible, traffic throttling should be performed [13].

3 Research questions and methods

The goal of this thesis is to develop a solution for Diameter overload control mitigation and evaluate its performance. The development can on a high level be divided into two parts: theoretical design and practical implementation and evaluation. The design part consists of requirement specification and solution design, and presentation. The second part describes the implementation, evaluation environment, results, and discussion. The two last points of the practical part are presented in Sections 4 and 5 and rest in this section.

3.1 Diameter overload control solution

This part describes requirements, design, implementation, and the evaluation framework. The two first parts are purely theoretical while the rest describe the reference implementation and its evaluation.

3.1.1 Requirements

The entity uses the available overload information from HSS and DRA entities. Messages will be classified in two classes whether they are part of a transaction or not. In the context of the solution, a transaction is either an attach or a TAU procedure. Messages belonging to ongoing transactions will have a higher priority than other messages. The messages which start a transaction do not have a higher priority, but following messages in the same transaction again have a higher priority. The entity shall perform the send or drop action depending on the message type and overload situation.

3.1.2 Design

I will design an algorithm to be used by an overload mitigation entity, which shall decide which requests are to be dropped. The entity that performs the actual drop operation will be the MME or an entity related to the MME. In this case, the reacting and reporting nodes defined in [19] are the MME and the HSS or DRA.

A flowchart presenting the high level overview of the algorithm is shown in Figure 11. The pathological case of no configured HSSs or DRAs is not covered by the algorithm. The assumptions made in the flowchart are presented below:

- The sending rate is defined per application (S6a, S13 and SLg) to enable higher granularity of overload control.
- The observation interval used to measure the rate is one minute. The interval works on a running update mechanism for all hosts. The length of the observation interval naturally affects behaviour of the system. With high loads, a long interval causes all messages to be dropped once the “quota” of that interval has been exhausted. In certain cases, it might be beneficial to change the interval length depending on the environment.

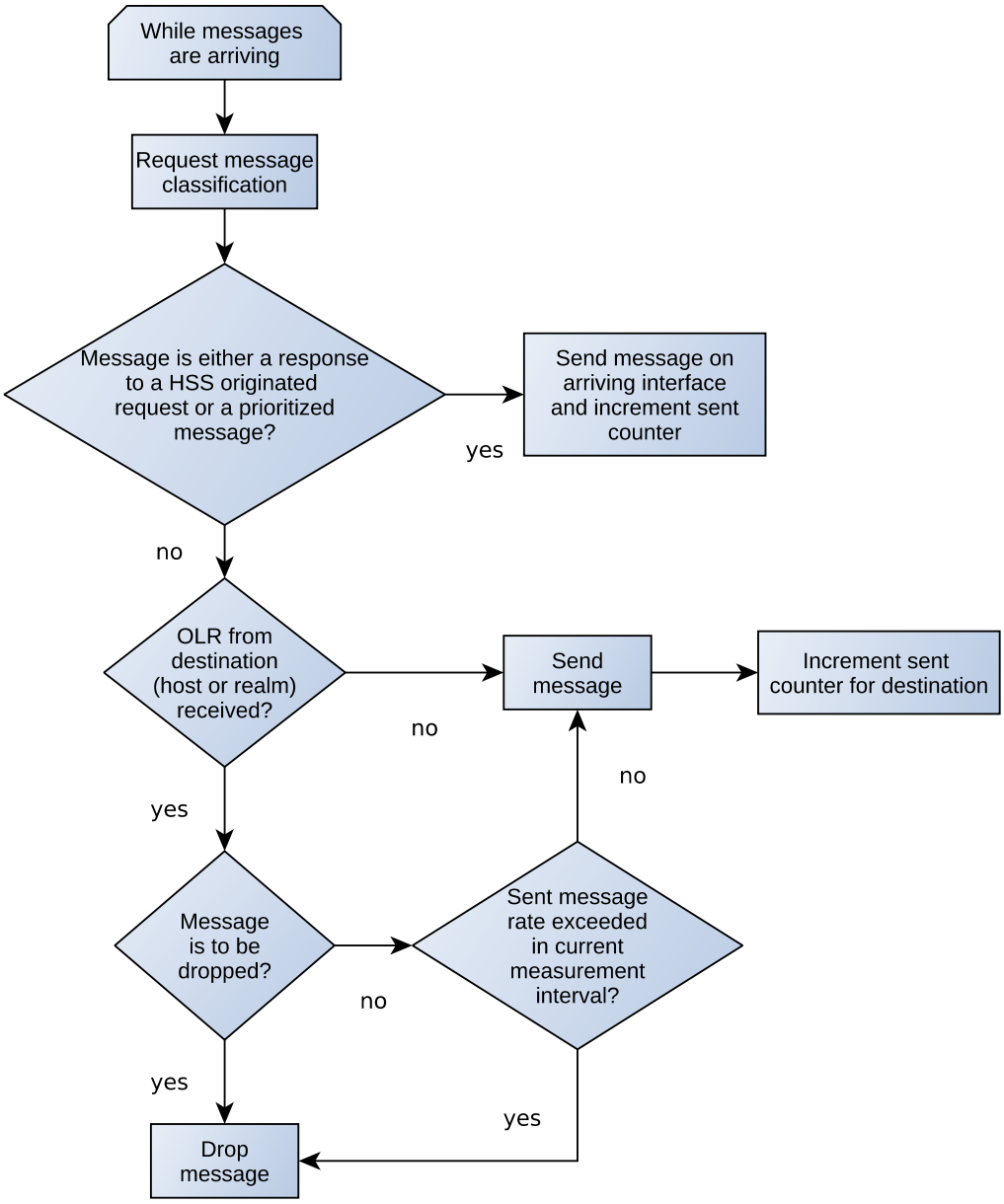

Figure 11: An overview of the arriving message handler of the overload control algorithm.

The process is started by requesting the message to be classified by an upper layer application. The classification will decide what is the priority of the message. Messages belonging to a transaction have a high priority, while other messages are assigned lower priority. The next step is to check if the message in question is an answer to a HSS originated request or a high priority message. Such messages are always sent because they allow the HSS to continue its operation by finishing transactions and thus reduce the load.

The next task is to check whether an OLR has been received from the message's

destination. If not, the message can be send straight away and the sent message counter is incremented. Otherwise an OLR has been received and the need for dropping the message is determined. The precise logic for this is explained later in the pseudocode. In case the message does not need to be dropped a further check is performed before sending. It is possible that the maximum sending rate defined per destination is exceeded. In such a case the message has to be dropped.

The overload reports are received through the `OC-OLR` AVP which is piggybacked in answer messages from HSSs or DRAs. An OLR can concern either a host, realm or peer (DRA) and they are distinguished with the `OC-Report-Type` AVP values 0, 1, and 2 respectively [19, 13]. The OLR reception algorithm can be viewed in Figure 12. The first thing to do when an answer message with an OLR is received is to ensure that is has arrived from a next step or final destination. OLRs from other entities are not of interest because they do not directly influence whether a message is sent or not. By checking the value of the `OC-Validity-Duration` AVP in the `OC-OLR` AVP it can be decided whether an OLR has ended, or a new one has arrived, or an existing is updated [19]. The end of an overload is signaled by an entity with a zero value in the `OC-Validity-Duration` AVP [19]. If the value zero is received, the previous sending rate to the host is restored. When a positive value larger than zero and less or equal 100 is received, there is either a new overload or an existing is updated [19]. In order to enable the restoration of the original rate after the overload has ended, the current sending rate at the moment of the OLR reception, is stored in a table. The value of the validity duration is stored and a timer is either started or updated. When the timer fires, the original sending rate is restored. The penultimate step is to set new rate as calculated from the `OC-Reduction-Request` AVP. The final step is optional depending on the realm OLR support.

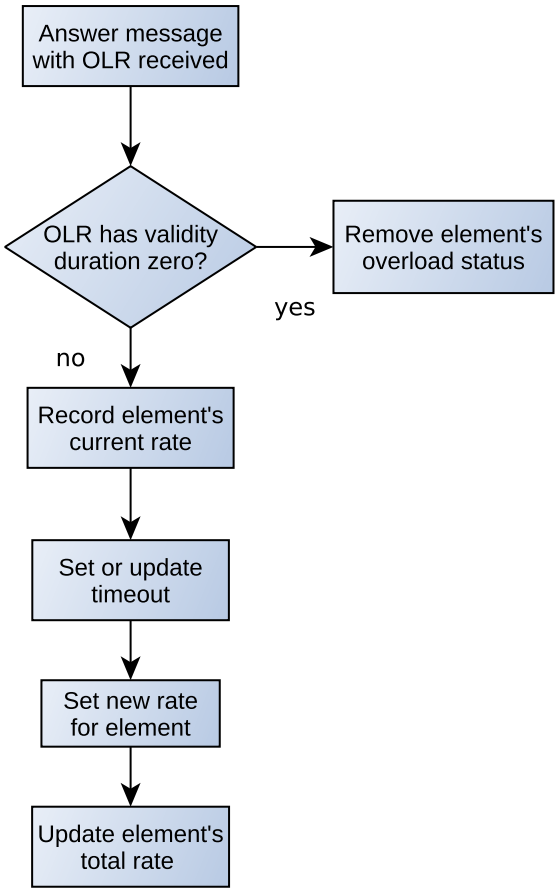

Figure 12: An overview of the OLR reception handling.

A table where the overload status is stored can be seen in Table 2. The Table contains a few rows of example data. The first three columns contain the key of the table (`Application-Id` AVP and host name), its type (host or realm) and the current sequence number. The two following columns contain the sending rates for the application: the current rate and the maximum rate. Both rates are measured in requests per minute. The timeout value of the last received OLR is in the last column. The realm overload data table has a similar format as Table 2 but each entry is identified by the `Application-Id` AVP and `Destination-Realm` AVP pair [19].

Table 2: Table for storing host overload data with example data.

ID	**Type**	**Sequence Number**	**Current Rate**	**Maximum Rate**	**Timeout** [s]
16777251-HSS-1	Host	5	100	140	180
16777251-DRA-1	Host	0	300	320	120

The actual algorithm is presented in Algorithm 1. It takes the message as an input and returns the action (send or drop) done to the message. The overload data table (ODT) and message counter per destination are maintained internally. The *ClassifyMessage()* function is defined in Algorithm 2. The decision to send or drop a message is based on current rate of the message's destination. As previously noted, messages may be dropped even if they survive the initial throttling. Whenever a message is sent towards a destination the C_M counter is updated. At the start or end of each new measurement interval this counter is reset for all destinations.

There may be expired OLRs in the ODT and they must be cleaned up [19]. The *ExpireTimeouts()* function in Algorithm 1 must be called regularly to remove the OLRs that have expired. Another possibility is to use timeout checking at message arrival, but it may create excess load when the message interarrival time is small.

Algorithm 1: Arriving message handling algorithm.

```
1  Function HandleArrivingMessage (M)
       Data: Overload data table (ODT)
       Data: Sent message counter per destination and Application-Id (host or realm) C_M
       Output: Action (send or drop message)
       /* Measurement interval */
2      Interval ← 1 minute
       /* Message classes:  low or high (prioritized) */
3      Class ← ClassifyMessage(M)
4      if M is an answer to a HSS request or Class = high then
5          Send (message on arriving interface)
6          Increment C_M for destination
7      else
8          OLData ← GetHostOverloadData(M)
9          if OLData = null then
               /* No overload */
10             Increment C_M for destination
11             return Send
12         else
13             if OLData[Current Rate] > OLData[Maximum Rate] then
14                 return Drop
15             else
16                 if C_M < (OLData[Maximum Rate] * Interval) then
17                     Increment C_M for destination
18                     return Send
19                 else
                       /* Interval message count exceeded */
20                     return Drop
21                 end
22             end
23         end
24     end
25 end
   /* A function for removing expired timeouts */
26 Procedure ExpireTimeouts
       Data: Overload data table (ODT)
27     CurrentTime ← current time
28     foreach OL in ODT do
29         if OL[Expiry Time] > CurrentTime then
30             ODT.remove(OL)
31         end
32     end
33 end
```

The OLR reception algorithm pseudocode is shown in Algorithm 2. The high level operation is presented in Figure 12. The calculation of the new maximum rate is based on the current sending rate and `OC-Reduction-Percentage` AVP in the following way: maximum rate = (100 - `OC-Reduction-Percentage`) * current rate. When a new overload is received or an existing is updated, the corresponding timer is either created or updated.

Algorithm 2: Functions of the OLR handling algorithm.

```
   /* Processes an arriving message with an OC-OLR AVP */
 1 Procedure ProcessOLR (M)
       Data: Overload data table (ODT)
       Data: List of timers (LT)
 2     Data ← GetHostOverloadData(M)
 3     if M[OC-Validity-Duration] = 0 then
           /* Overload has ended */
 4         if Data = null then
               /* Error:  end received before start */
 5             Stop
 6         end
 7         ODT.remove(Data)
 8     else
           /* A ''new'' overload or a current one is updated */
 9         Data[Timeout] ← M[OC-Validity-Duration AVP value]
10         Data[Maximum Rate] ← (100 - M[OC-Reduction-Percentage AVP]) *
           Data[Current Rate]
           /* Start or reset timer for OLR */
11         LT.add(Data[Timeout])
12     end
13 end
   /* Get a hosts' data from the overload data table */
14 Function GetHostOverloadData (M)
       Data: Overload data table (ODT)
15     if M contains the Origin-Host AVP then
16         Element ← M[Origin-Host] AVP value
17     else
18         Element ← M[Origin-Realm] AVP value
19     end
20     App-Id ← translate the Application-Id AVP to application name
21     foreach Row in ODT do
22         if Row[Hostname] = Element and Row[Application] = App-Id then
23             return Row
24         end
25     end
26     return null
27 end
```

3.1.3 Implementation

The algorithm was implemented in Clojure, a Lisp dialect running on the Java Virtual Machine (JVM). Instead of using a native Diameter implementation, I used a discrete event simulation framework for Java called DESMO-J. It provides features

such as queues, random number generation, and various statistical distributions.

The implementation can be logically divided into three parts: MME, DRA, and HSS. Only one MME and DRA are supported in the code. There can basically be an unlimited number of HSSs but the behaviour with a large number of HSSs is unpredictable. These three parts use some common functionality such as initialization code and functions for common tasks like logging and message sending.

As an approximation for Diameter messages, Clojure maps with Diameter AVPs as content are used. They contain all the mandatory AVPS and use the correct flags and correct code values as specified in RFC 6733 [15] but contain no payload. At simulation start up the normal CER message is sent, but the base protocol watchdog and other messages are not sent. These maps are placed into two queues: one from MME to HSS (server queue) and the second in the opposite direction (client queue). The MME, DRA, and HSSs check these queues for messages intended for the respective entities, remove them from the queue, and process them. If direct routing is used, the DRA does not do anything to the messages. As an example, the path of a request from the MME to a HSS is the following: the message (map) is created and placed in the client queue where the destination HSS sees it and removes it from the queue. The HSS creates the answer and places it in the client queue where the MME at some point removes it from the queue and processes it. If realm routing is used, the messages are placed in the server queue and the DRA removes it from the queue and relays it, that is, re-inserts it to the queue with the target HSS as destination. The HSS selection performed by the MME and the DRA is done in a round-robin fashion. There are certainly other and probably better ways of doing the selection, like distributing the messages according to the load of each HSS. The destination selection in the DRA additionally contains a mechanism for checking all configured HSSs if the originally selected one is overloaded. If one is available it is used, but in case all are overloaded the message is dropped. Once the HSS is ready with the answer, it is placed in the client queue, the DRA removes it and re-inserts again. When removing and inserting the message, the DRA changes the routing AVPs to make the handling transparent to the MME.

The times at which the MME and HSS send requests are sent determined by exponential distributions. The parameters for these distributions are calculated from live network data. Service time at a HSS is simulated with a delay sampled from a uniform distribution. The length of an overload is also random with a sample from an uniform distribution.

The check to see whether an event, like message sending should occur, is done by adding the sample from the relevant distribution to the simulation time of the last occurrence of the event. This value is then compared to the current simulation time and if the value is larger, the event is executed. The same logic is also used for deciding whether a transaction should be started but with the difference that the sample is not random but a constant value.

The rate of sent or received messages calculation is based on comparing the timestamps of the sent or received messages to a window ending at the current simulation time and starting n seconds before the current time, where n is tuneable parameter. The messages which are inside this window are count to the message

rate.

A feature also present in the implementation is the possibility of randomly cancelling an overload. If a sample from a uniform distribution $\mathcal{U}(0,1)$ is larger than a predefined threshold, a randomly selected overload is cancelled as previously described. The motivation for this feature is to simulate a sudden decrease in incoming traffic causing an overload to end.

The overload control solution is implemented as described in Subsection 3.1.2 with some exceptions. The first is that rate observation interval is one second. This change is due to the short run time of the simulation. The second is the lack of support for multiple `OC-OLR` AVPs in one answer message.

3.1.4 Performance evaluation

The performance of the solution is measured by running the algorithm implementation with several configurations.

The simulation is done with two scenarios which both are based on data taken from live networks. Scenario one has one million subscribers and scenario two 300 000 subscribers. The message distributions of the two scenarios are shown in Figures 13 and 14. It can be seen from the Figures that scenario two does not contain all the messages that are in scenario one. Additionally, the AIR, CLR, ECR, and ULR dominate the other messages in terms of occurrence. However, the absolute number of messages is more important to the simulation than the distribution of messages.

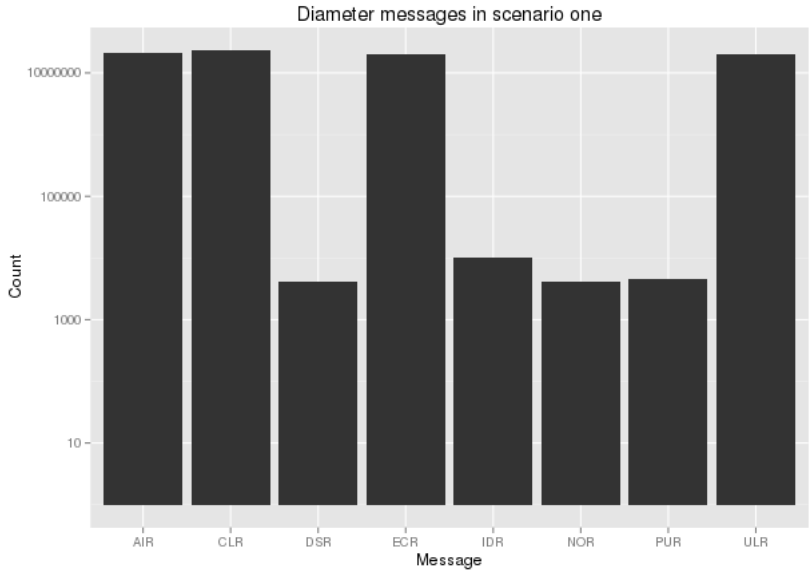

Figure 13: Message distribution of scenario one. Note the logarithmic y-axis.

The evaluation is based on performance indicators measuring the number of successes and failures of both single messages and transactions. The following performance indicators were defined for evaluation purposes:

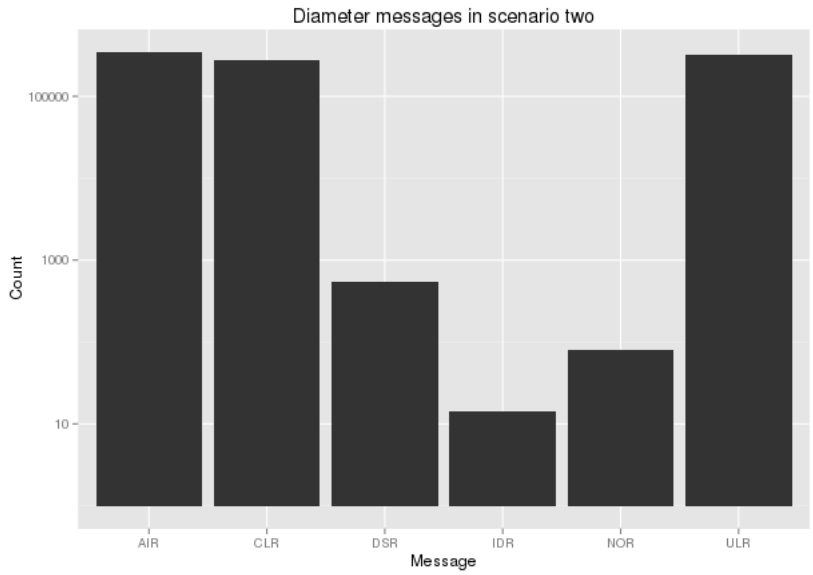

Figure 14: Message distribution of scenario two. Note the logarithmic y-axis.

- **Successful MME originated transactions** This is the total number of all successfully completed transactions initiated by the MME. Attach and TAU are the transactions sent by the MME. The goal of this indicator is to see what effect throttling has on success rate of transactions. The two transaction types offer a good platform for measurements as an attach consists of three messages and a TAU of one message. This indicator is measured by counting the number of consecutive messages, that make up either an attach or a TAU. In case there is a timeout before all the messages are received or one of the answers does not have a `Result-Code` AVP indicating success, the whole transaction is considered failed.

- **Successful MME attach transactions** This measures the total number of attach transactions that were successful, which gives a good picture of the long-term transaction performance since an attach consist of multiple messages. If one of these messages fail, the whole transaction fails. The criteria for attach success is the same as for all transactions.

- **Unsuccessful MME originated messages** An indicator counting the number of failed or unsuccessful messages, including both transactional and non-transactional messages. This is somewhat the opposite to the first indicator but on a finer level. The measurement for this indicator is done by counting the answer messages to requests which have a `Result-Code` AVP with a value indicating an error.

- **HSS originated messages to MME** An indicator for counting messages from a HSS to the MME. Its main purpose is to measure how the prioritization of answer messages to a HSS works. The measurement for this indicator

consists of counting answer messages to requests sent to a MME.

The optimal value for the MME originated transactions as well as HSS originated messages is as large as possible because then the core network can serve as many subscribers as possible. The number of unsuccessful MME originated messages should be as small as possible because then there is little overhead from failed messages which need to be sent again. The optimal combination of the indicators is to have as many successful messages and as few unsuccessful messages or failures as possible.

In the simulation, the performance indicators presented above are calculated in a similar way as explained above. The check for successful MME originated transactions is done when an answer to a MME sent request is received. If the message is a part of a transaction, and the message contains a result with an error, the transaction has failed. Failure also happens if an answer to a request is not received within the specified timeout. The check for successful attach transactions happens in the same way as for all transactions, but only attaches are considered. Unsuccessful MME originated messages are checked at a MME when an answer is received by inspecting the `Result-Code` AVP for a non-successful value. The HSS originated messages are counted when an answer to a MME directed request is received.

4 Results

As previously explained, two scenarios were used to evaluate the performance of the solution. The simulation parameters for each scenario can be divided into two groups: routing and algorithm related and the remaining. The first group defines the message routing mode (direct or realm routing) and algorithm mode (no OLC, no prioritization, and smart). The behaviour of the algorithm modes is explained next. In the no OLC mode, as the name indicates, an overload reported by a HSS is not taken into account. This means that a request to a HSS is sent even if it is overloaded. The smart mode is the implementation of the overload control algorithm proposed in this thesis. Finally, the no prioritization mode is the same as the smart mode with the exception that answers to HSS originated requests are not prioritized, see the flowchart in Figure 11. This means that such answers are dropped if the originating HSS is overloaded.

The second group contains message distribution parameters, timer durations, et al. A further division can be done into scenario dependent and static parameters. The scenario dependent parameters are the mean values of the MME to HSS and HSS to MME message sending distributions. They define the mean values for the exponential distributions which control the time between two successive sent messages.

The static parameters are the following:

- Length of run in simulation time: two minutes
- HSS request service time (uniform distribution): 0.02 - 0.08 s
- Overload duration (uniform distribution): 5 - 15 s
- Time between two started transactions: attach 0.009 s (111 started transactions per second), TAU 0.036 s (28 started transactions per second)
- HSS capacity: 20 requests per second unless otherwise noted
- The probability of an overload being cancelled: 0.9

The scenario dependent parameters are presented next. Scenario one, which is based on a live network with one million subscribers, has the following exponential distribution mean values:

- MME to HSS sent messages: 0.0181 1/s (55 requests per second)
- HSS to MME sent messages: 0.0476 1/s (21 requests per second)

Scenario two, which is based on a live network with 300 000 subscribers, has the following exponential distribution mean values:

- MME to HSS sent messages: 0.0292 1/s (34 requests per second)
- HSS to MME sent messages: 0.0715 1/s (14 requests per second)

A part of these parameters are based on data from live networks. The following are based on real data: MME to HSS and HSS to MME message distribution means and the time between started transactions. The remaining parameters were chosen appropriately to give a good starting point for the performance evaluation. The two minute simulation length may seem short but in conjunction with the other parameters it is enough to measure the performance of the solution. Also the HSS capacity is on purpose set to a low value so the behaviour of proposed solution can be observed during overloads.

Both scenarios were simulated with two, five, and ten HSSs for all combinations of the routing and algorithm modes. In addition, the HSS capacity was increased to see how additional capacity affects the results.

The network setup for the evaluation consists of one MME, one DRA, and a changing number of HSSs. In the direct routing mode there is a direct connection between the MME and each HSS. In the realm routing mode all messages between the MME and a HSS are relayed through the DRA in both directions. These setups for both direct and realm routing modes with two, five, and ten HSS configurations are shown below. The two HSSs configurations can be seen in Figures 15 and 16.

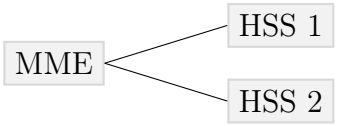

Figure 15: A MME and two HSS configuration in the direct routing mode.

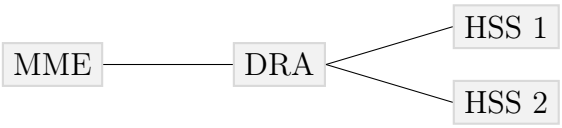

Figure 16: A MME and two HSS configuration in the realm routing mode.

The five HSS configurations in the direct and realm routing modes are shown in Figures 17 and 18.

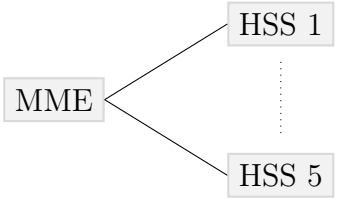

Figure 17: A MME and five HSS configuration in the direct routing mode.

The ten HSS configurations in the direct and realm routing modes are shown in Figures 19 and 20.

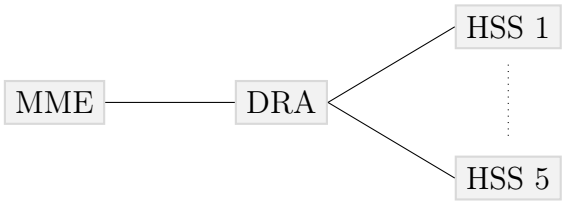

Figure 18: A MME and five HSS configuration in the realm routing mode.

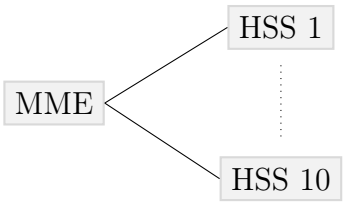

Figure 19: A MME and ten HSS configuration in the direct routing mode.

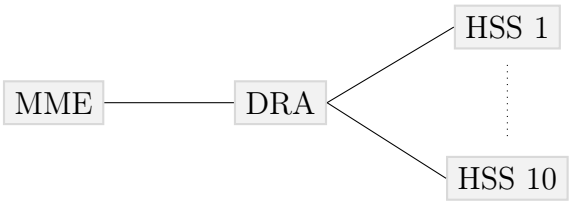

Figure 20: A MME and ten HSS configuration in the realm routing mode.

4.1 Two HSSs

The simulation results of the performance indicators for scenarios one and two with two HSSs for the direct and realm routing modes are presented in Figures 21 and 22. The most obvious observation from the Figures is that the number of successful MME originated transaction is very low for the no OLC algorithm mode. This is due to the overload caused at the HSSs because the arrival rate of Diameter requests from the MME is larger than the HSS capacity. The amount of unsuccessful MME messages is correlated with the transactions as the transaction messages are also MME originated. In the no prioritization and smart modes the values are much higher, which is better, because requests are not sent if overload is reported by a HSS. The smaller values in scenario two are caused by the smaller number of subscribers of that scenario. The amount of started transactions and sent messages is determined by the subscriber amount.

The number of HSS originated requests is larger in the direct routing mode due to differences of the algorithm modes. In the no OLC algorithm mode there is no overhead from sending OLR messages to the MME. Also the wall clock time advances slower when requests are sent all the time and thus a HSS can send more messages when there is no OLC. The smaller number of messages in no prioritization mode, compared to the smart mode, is due to the design of that mode, where answers to requests from HSSs do not have a higher priority and may be dropped. Also here the smaller value of scenario two is due to the distribution parameter controlling the HSS message sending, which does not send as many messages as in scenario one.

The realm routing mode features better results in all performance indicators. This phenomenon is caused by the better usage of available HSS resources. In practice this means that the DRA may select between several HSSs and choose a non-overloaded HSS if one or many HSSs are overloaded. Naturally if all HSSs are overloaded, there is no advantage of realm routing.

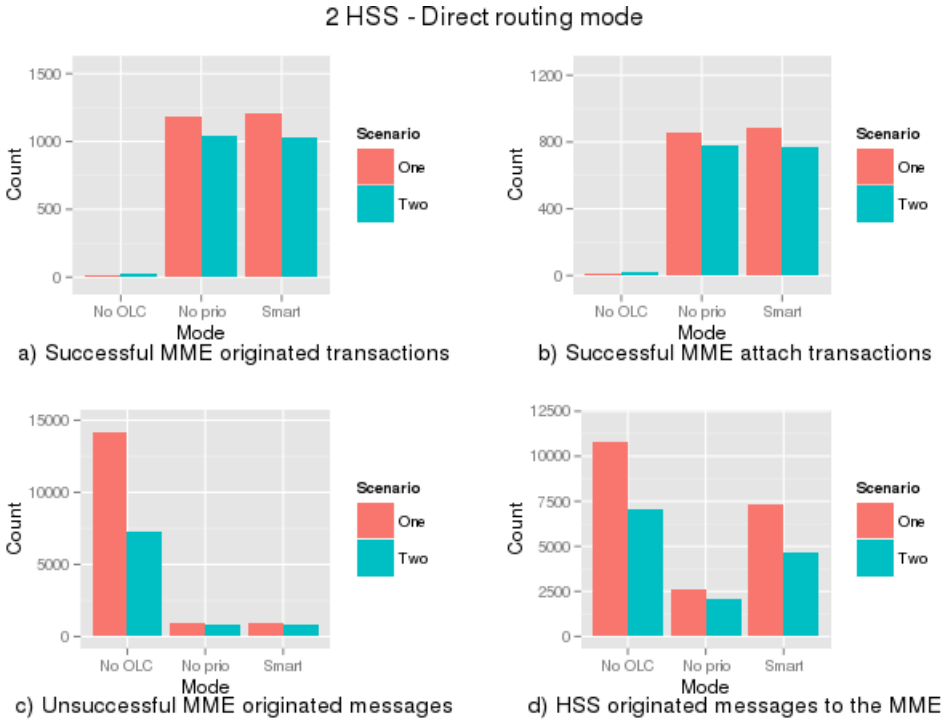

Figure 21: The two scenarios in direct routing mode with two HSSs.

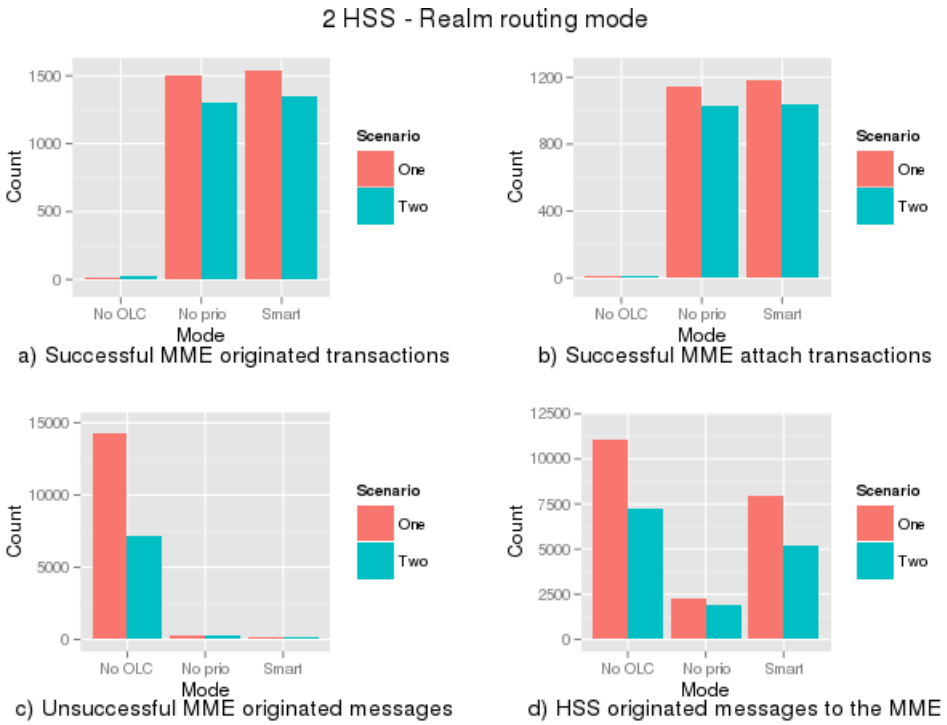

Figure 22: The two scenarios in realm routing mode with two HSSs.

The variation of performance indicator values between the algorithm and routing modes are presented in Tables as Table 3 for both scenarios with two HSSs. The algorithm modes on the lower row in the table header are reference modes to which the mode on upper row is compared to. The values are calculated with the formula

$$c_r(x, x_r) = \frac{x - x_r}{x_r} \cdot 100,$$

where x is the value of the upper row and x_r is the reference value. So, for example, the value -99.0 % for the no OLC and no prioritization modes is calculated as follows

$$c_r(12, 1188) = \frac{12 - 1188}{1188} \cdot 100 \approx -99.0\%.$$

If the change is negative, the value compared to is smaller than the reference value, and conversely in the opposite case. There are certain cases where the percentage change is infinite because the reference value is zero.

As Table 3 and the previous Figures show, the new algorithm modes are superior in three of four performance indicators for scenario one. Only in HSS originated messages the direct routing mode is better. The changes are particularly large for the three first indicators as the results for the no OLC algorithm mode are rather poor. In the realm routing mode, the changes are even bigger because the mode has better performance than the direct routing mode. The difference of 10 % to even 900 % between the direct and realm routing modes can be explained by the different HSS selection method employed by the DRA. In the direct routing mode with the current round-robin HSS selection, it is not possible to select another HSS in case the one originally selected is overloaded. The small variation between no prioritization and smart algorithm modes in the MME transaction indicator values is due to effects caused by the simulation software.

The lower part of Table 3 shows the differences of scenario two. Also here the performance in the no OLC routing mode is again poor in comparison to the other modes. However, the changes between the modes are smaller than in scenario one, but it is partly due to the smaller amount of incoming traffic. The changes between the no prioritization and smart algorithm modes are again rather small, except for the HSS originated messages indicator in which the no OLC mode is the best.

To see what effect the transaction sending rate has on the value of the unsuccessful MME messages indicator, two further scenario one simulation runs with the unmodified two HSS configuration were done. As later simulations will show, this indicator has an interesting behaviour, which motivates these extra runs. In the first run, the time between the start of two transactions was doubled to 0.018 s for attach and 0.072 s for TAU transactions. The performance changes of this run are shown in Table 4. The value of the unsuccessful MME messages indicator for the no OLC and smart algorithm modes in the direct routing mode has dropped from 1438.5 % in the unmodified simulation to 1136.4 % in this simulation. This decrease of 21.0 % is not as large as the change in the simulation parameters, but shows that it has some effect on the value.

A further simulation was done where the time between the start of two transactions was quadrupled to 0.036 s for attach and 0.144 s for TAU transactions. The

Table 3: Relative performance changes between algorithm modes of the two scenarios with two HSSs.

Performance indicator	**No OLC** [%]		**No prio** [%]		**Smart** [%]	
	No prio	**Smart**	**No OLC**	**Smart**	**No OLC**	**No prio**
Scenario one						
Direct routing mode						
All MME transactions	-99.0	-99.0	9800.0	-1.6	9958.3	1.6
MME attach transactions	-99.0	-99.0	9455.6	-2.7	9722.2	2.8
Unsuccessful MME messages	1372.8	1438.5	-93.2	4.5	-93.5	-4.3
HSS originated messages	315.7	47.7	-75.9	-64.5	-32.3	181.4
Realm routing mode						
All MME transactions	-99.0	-99.0	9913.3	-2.0	10120.0	2.1
MME attach transactions	-99.0	-99.1	10290.9	-3.1	10618.2	3.1
Unsuccessful MME messages	4424.8	7209.2	-97.8	61.5	-98.6	-38.1
HSS originated messages	379.7	39.2	-79.2	-71.0	-28.2	244.6
Scenario two						
Direct routing mode						
All MME transactions	-97.5	-97.5	3923.1	1.1	3880.8	-1.1
MME attach transactions	-97.8	-97.8	4494.1	1.8	4111.8	-1.8
Unsuccessful MME messages	779.1	787.7	-88.6	1.0	-88.7	-1.0
HSS originated messages	235.9	51.8	-70.2	-54.8	-34.1	121.3
Realm routing mode						
All MME transactions	-98.1	-98.1	5128.0	-3.3	5304.0	3.4
MME attach transactions	-98.4	-98.5	6312.5	-1.2	6387.5	1.2
Unsuccessful MME messages	2826.8	3990.9	-96.6	39.8	-97.6	-28.5
HSS originated messages	284.6	38.9	-74.0	-63.9	-28.0	176.9

performance changes for this second run are shown in Table 5. Here the same values as in the first run are 1438.5 % and 976.3 % leading to a decrease of 32.1 %. This change is again not as big as the parameter values change, but reaffirms at least a partial effect of the transaction sending on the unsuccessful MME messages.

In order to see how adding more capacity changes the results, two modified simulation runs were done. The first modification was to add more capacity to one of the HSSs. The capacity of HSS1 was boosted to 80 requests per second. This is theoretically enough to prevent the HSS from becoming overloaded and therefore improve the performance indicators.

Figures 23 and 24 show the results for the direct and realm routing modes for both scenarios. The most important change has happened in the number of errors with the MME originated requests in the no OLC algorithm mode, where the absolute number of unsuccessful MME originated messages has approximately halved and the number of successful transactions in the no OLC algorithm mode shows an increase from near zero to over 2000 and about 1500 for scenario one and two respectively in the direct routing mode. Additionally, the value of the same performance indicator in the no prioritization and smart modes has nearly tripled from the unmodified

Table 4: Relative performance changes between algorithm modes of scenario one with two HSSs with increased transaction start time.

Performance indicator	**No OLC** [%]		**No prio** [%]		**Smart** [%]	
	No prio	**Smart**	**No OLC**	**Smart**	**No OLC**	**No prio**
Direct routing mode						
All MME transactions	-98.8	-98.8	8128.6	2.5	7928.6	-2.4
MME attach transactions	-98.8	-98.8	8230.0	-0.8	8300.0	0.8
Unsuccessful MME messages	1157.3	1136.4	-92.0	-1.7	-91.9	1.7
HSS originated messages	269.4	36.1	-72.9	-63.2	-26.5	171.4
Realm routing mode						
All MME transactions	-99.2	-99.2	13063.6	0.4	13009.1	-0.4
MME attach transactions	-99.3	-99.3	13975.0	0.3	13937.5	-0.3
Unsuccessful MME messages	4759.0	5791.2	97.9	21.2	-98.3	-17.5
HSS originated messages	343.8	29.8	-77.5	-70.7	-23.0	241.8

Table 5: Relative performance changes between algorithm modes of scenario one with two HSSs with further increased transaction start time.

Performance indicator	**No OLC** [%]		**No prio** [%]		**Smart** [%]	
	No prio	**Smart**	**No OLC**	**Smart**	**No OLC**	**No prio**
Direct routing mode						
All MME transactions	-97.9	-97.9	4600.0	-2.6	4727.3	2.7
MME attach transactions	-97.9	-97.9	4618.8	-2.7	4750.0	2.8
Unsuccessful MME messages	909.7	976.3	-90.1	6.6	-90.7	-6.2
HSS originated messages	222.1	28.6	-69.0	-60.1	-22.3	150.4
Realm routing mode						
All MME transactions	-98.7	-98.8	7652.9	-3.5	7935.3	3.6
MME attach transactions	-98.8	-98.9	8491.7	-3.3	8783.3	3.4
Unsuccessful MME messages	4456.9	5116.4	-97.8	14.5	-98.1	-12.6
HSS originated messages	258.9	23.0	-72.1	-65.7	-18.7	191.8

run. Interestingly, the MME originated transactions in the no OLC algorithm mode of the realm routing mode shows a smaller improvement than in the direct routing mode. An explanation for this phenomenon is how the no OLC mode works. As it does not take overload into account when selecting the target HSS, the additional capacity of the other HSS is not utilized. In the other algorithm modes, the extra capacity is used and it leads to much improved results. These results indicate that adding more capacity while keeping the load constant improves the performance of whole system, which is a natural and obvious conclusion.

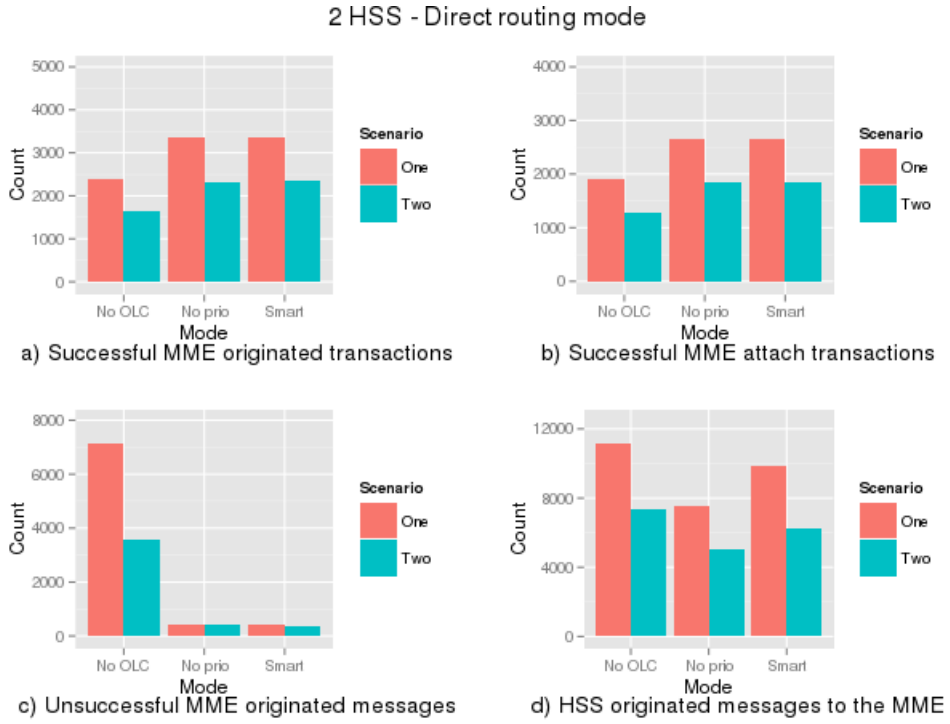

Figure 23: First modified scenarios in direct routing mode with two HSSs.

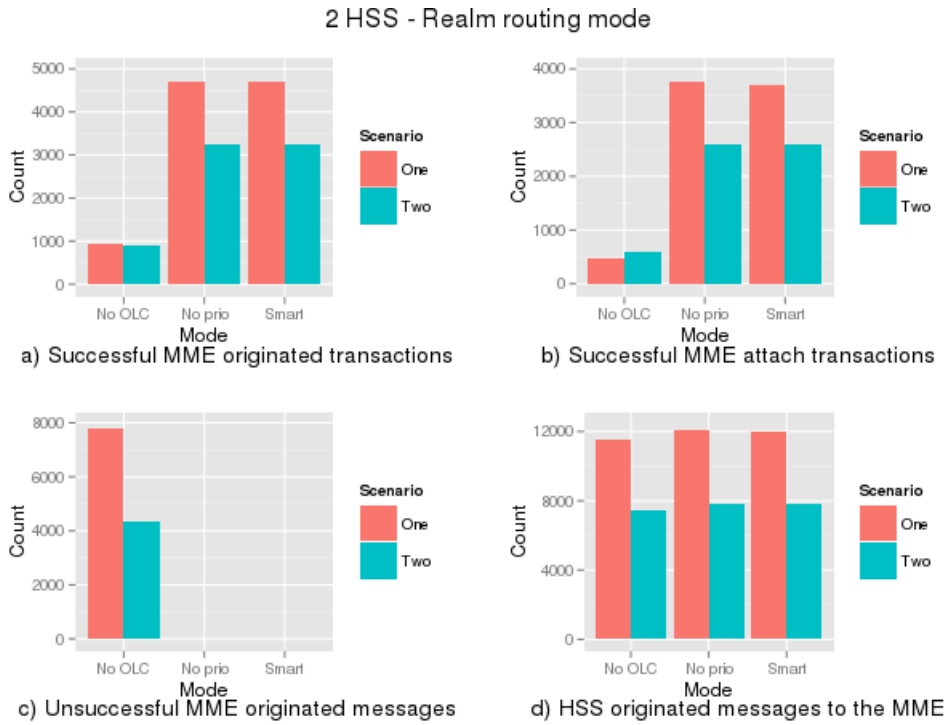

Figure 24: First modified scenarios in realm routing mode with two HSSs.

Table 6 shows the performance change of the various algorithm and routing modes. The relative changes are now much smaller than in the unmodified scenario as seen in Table 3, e.g. the value of the successful MME transaction indicator has decreased about 60 %. Also the other indicators show improvement of various degree. Particularly the MME originated transactions in the no prioritization and smart algorithm modes begin to have very small differences in both routing modes. The same observation also applies for HSS originated messages in realm routing mode. The changes in scenario two do not differ very much from scenario one. Exceptions to this are the unsuccessful MME messages in direct routing mode and the MME originated transactions in the realm routing mode. The exceptions are due to the way the no OLC works and the still insufficient capacity of the HSSs.

Table 6: Relative performance changes between algorithm modes of the first modified scenarios with two HSSs.

Performance indicator	**No OLC** [%]		**No prio** [%]		**Smart** [%]	
	No prio	**Smart**	**No OLC**	**Smart**	**No OLC**	**No prio**
Scenario one						
Direct routing mode						
All MME transactions	-29.3	-28.8	41.5	0.7	40.5	-0.7
MME attach transactions	-28.8	-29.0	40.4	-0.4	40.9	0.4
Unsuccessful MME messages	1640.0	1547.6	-94.3	-5.3	-93.9	5.6
HSS originated messages	47.9	13.9	-32.4	-23.0	-12.2	29.9
Realm routing mode						
All MME transactions	-80.3	-80.2	408.2	0.4	406.2	-0.4
MME attach transactions	-80.2	-87.0	684.4	1.6	672.1	-1.6
Unsuccessful MME messages	∞	∞	-100.0	0	-100.0	0
HSS originated messages	-4.2	-3.4	4.4	0.8	3.6	-0.8
Scenario two						
Direct routing mode						
All MME transactions	-28.3	-29.5	39.4	-1.7	41.9	1.8
MME attach transactions	-29.4	-29.7	41.7	-0.3	42.2	0.3
Unsuccessful MME messages	789.9	887.6	-88.3	11.0	-89.9	-9.9
HSS originated messages	45.7	18.9	-31.3	-18.3	-15.9	22.5
Realm routing mode						
All MME transactions	-72.9	-72.7	268.9	0.7	266.3	-0.7
ME attach transactions	-77.8	-77.7	350.2	0.2	349.1	-0.2
Unsuccessful MME messages	∞	∞	-100.0	0	-100.0	0
HSS originated messages	-4.6	-4.7	4.8	-0.1	4.9	0.1

The second modification was to boost the capacity of both HSSs to 80 requests per second. This capacity is theoretically enough to prevent both HSSs from becoming overloaded. The performance indicator values for the direct routing mode are shown in Figure 25. One can instantly see that almost all performance indicators have the same values for all algorithm and routing modes. The realm routing mode values are omitted as it has practically identical values for the performance

indicators, except the unsuccessful MME originated messages indicator, where there are zero failed messages in all algorithm modes. This confirms that the theoretical expectations are fulfilled, that is, all algorithm modes have approximately equal performance and there are almost no unsuccessful MME originated messages. The conclusion one can draw from these results is that with enough capacity no OLC solution is needed.

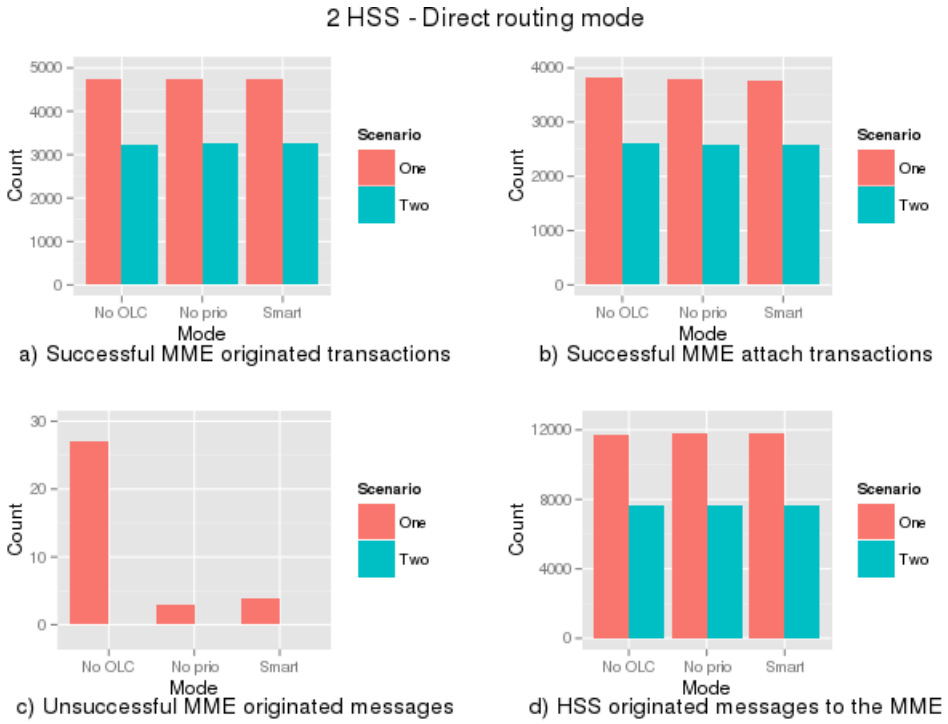

Figure 25: Second modified scenarios in direct routing mode with two HSSs.

4.2 Five HSSs

The next task was to run the simulation more HSSs, in this case five, to see how the solution performs when it has more capacity. In order to get a baseline result, a simulation run with a HSS capacity of 20 requests per second was done.

The results for all routing modes in scenarios one and two are shown in Figures 26 and 27. As the Figures show, the performance is better than in the two HSS run. In fact, the performance is similar to the first modified two HSS run with the exception of HSS originated requests, which is larger in this run. Also the number of unsuccessful MME originated requests in the no prioritize and smart algorithm modes is a bit larger in this run. Five HSSs can be seen as two entities with capacities of 80 and 20 requests per second, which partly explains the similar results. Naturally this comparison is quite crude because many other factors have an impact on the behaviour and performance. The performance of the unsuccessful MME messages indicator in scenario two is superior in comparison to scenario one. It looks like the lower message rate enables more efficient use of capacity.

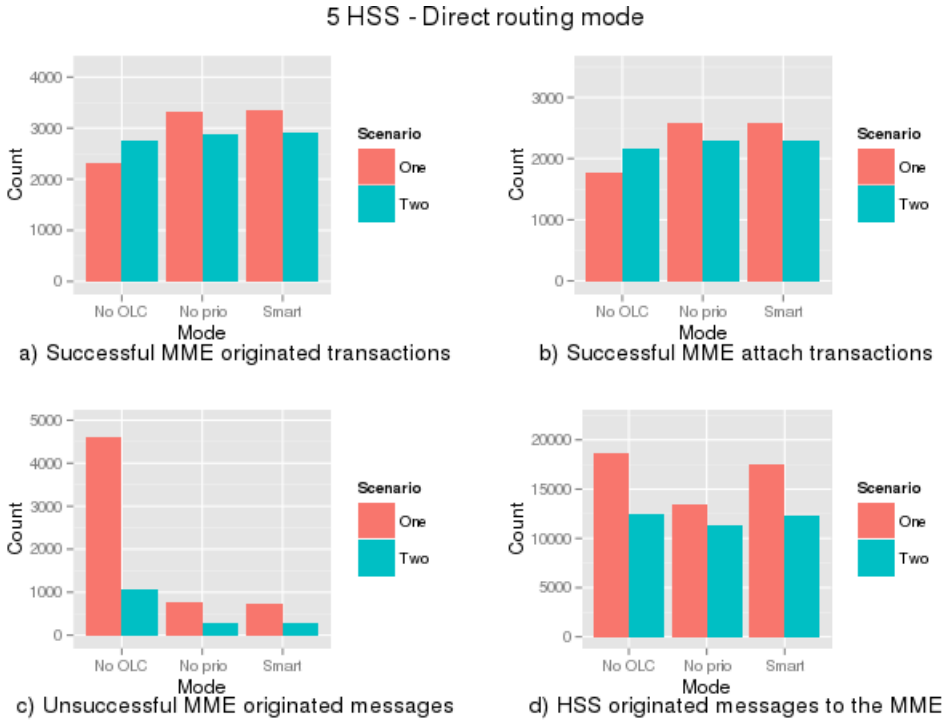

Figure 26: The two scenarios in direct routing mode with five HSSs.

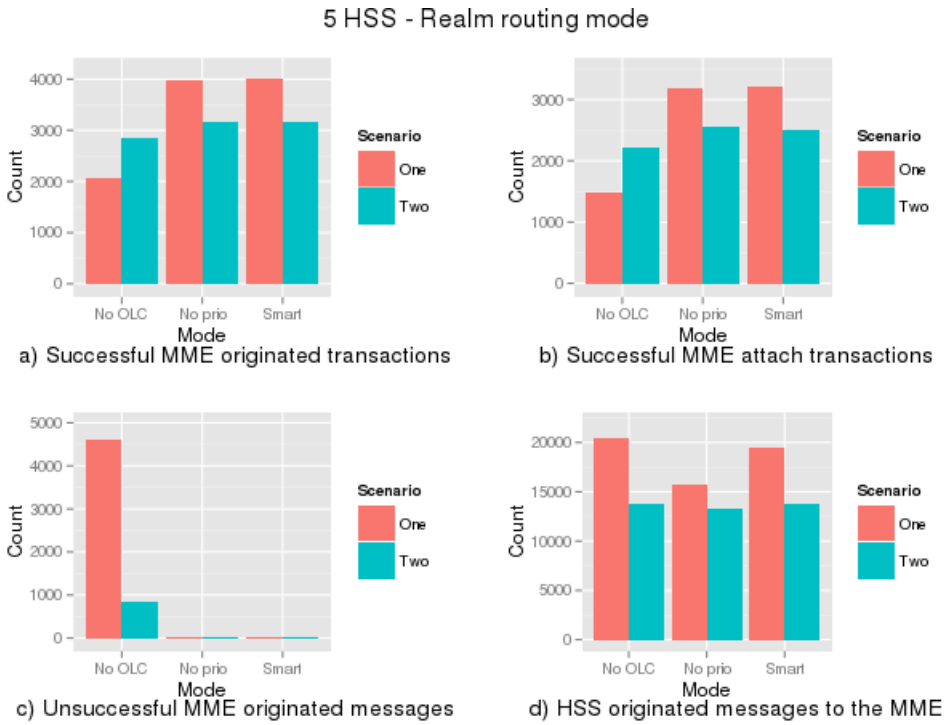

Figure 27: The two scenarios in realm routing mode with five HSSs.

Table 7 shows the performance changes between the algorithm and routing modes. Here the changes are already quite much smaller than with two HSSs demonstrating that more HSSs with a smaller capacity partially improve the performance readily. The exception here is the unsuccessful MME originated messages performance indicator whose value remain high with tripled capacity. This shows that the indicator depends more on the absolute capacity of one HSS than on the amount of HSSs until a threshold is exceeded. One additional thing to note is the high value of the same indicator in the realm routing mode. Its value is high because the value in the no prioritization and smart algorithm modes is so low in comparison to the no OLC mode. The absolute value has decreased to approximately one third in comparison to the unmodified two HSS run.

Table 7: Relative performance changes between algorithm modes of the two scenarios with five HSSs.

Performance indicator	**No OLC** [%]		**No prio** [%]		**Smart** [%]	
	No prio	**Smart**	**No OLC**	**Smart**	**No OLC**	**No prio**
Scenario one						
Direct routing mode						
All MME transactions	-29.9	-30.5	42.7	-0.8	43.8	0.8
MME attach transactions	-31.7	-31.4	46.5	0.5	45.8	-0.5
Unsuccessful MME messages	514.1	533.5	-83.7	3.2	-84.2	-6.2
HSS originated messages	39.3	6.6	-28.2	-23.4	-6.2	30.6
Realm routing mode						
All MME transactions	-48.1	-48.3	92.6	-0.5	93.6	0.5
MME attach transactions	-53.2	-53.5	113.8	-0.6	115.1	0.6
Unsuccessful MME messages	11692.3	12329.7	-99.2	5.4	-99.2	-5.1
HSS originated messages	10.1	5.1	-23.2	-19.2	-4.9	23.8
Scenario two						
Direct routing mode						
All MME transactions	-4.7	-0.5	5.0	-0.3	5.3	0.3
MME attach transactions	-6.0	-6.1	6.4	-0.1	6.5	0.1
Unsuccessful MME messages	304.1	292.4	-75.3	-2.9	-74.5	3.0
HSS originated messages	10.0	2.2	-9.1	-7.0	-2.2	7.6
Realm routing mode						
All MME transactions	-10.0	-10.1	11.1	-0.1	11.2	0.1
MME attach transactions	-13.0	-11.5	14.9	1.7	13.0	-1.7
Unsuccessful MME messages	3462.5	3462.5	-97.2	0.0	-97.2	0.0
HSS originated messages	3.7	0.4	-3.6	-3.2	-0.4	3.3

Again, one HSS was modified to have the capacity of 80 requests per second, to see how a non-overloaded HSS affects the performance indicators. The results of the modified simulation can be seen in Figures 28 and 29. The added capacity improves the performance of all indicators slightly. Notable in the realm routing mode is that there are no unsuccessful MME originated requests in the no prioritize and smart algorithm modes. When comparing these results with first modified two

HSS simulation run, one can see that having one HSS with enough capacity to not become overloaded, has the similar effect of having several HSSs, which become overloaded.

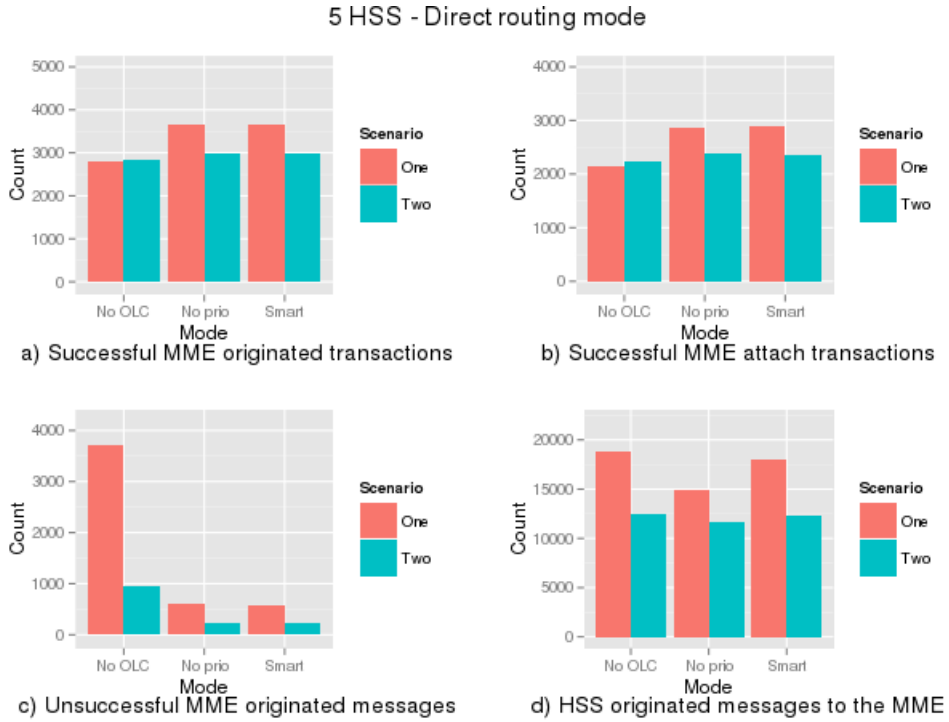

Figure 28: First modified scenarios in direct routing mode with five HSSs.

The relative performance changes are again shown in Table 8. Some observations can be made: the changes between the no prioritization and smart algorithm modes are very small. The differences are dominated by the unsuccessful MME messages indicator due to the no OLC mode's inability to utilize the added capacity. Additionally, the differences between the no OLC and other modes are decreasing, which can be expected as more capacity was added.

A second modified simulation, where two HSSs have their capacity changed to 80 requests per second and three have the standard capacity of 20 requests per second, was also done for the five HSS configuration in order to see the effects of further added capacity. The values of the performance indicators are shown in Figures 30 and 31. It is interesting to note that this configuration does not achieve the same performance as in the similarly modified two HSS configuration. This behaviour is caused by the additional HSSs having the standard capacity. When the requests are sent in a round-robin manner, the HSSs with extra capacity do not help as much, because all HSSs receive the same amount of traffic. If more traffic were sent to HSSs with more capacity, the total performance would be better. Otherwise, the results are quite similar a part to the MME originated transactions in realm routing mode, where the no OLC mode's performance is quite much behind the other algorithm modes. The amount of unsuccessful MME originated messages remains high also in this simulation. The reason for this has been explained earlier.

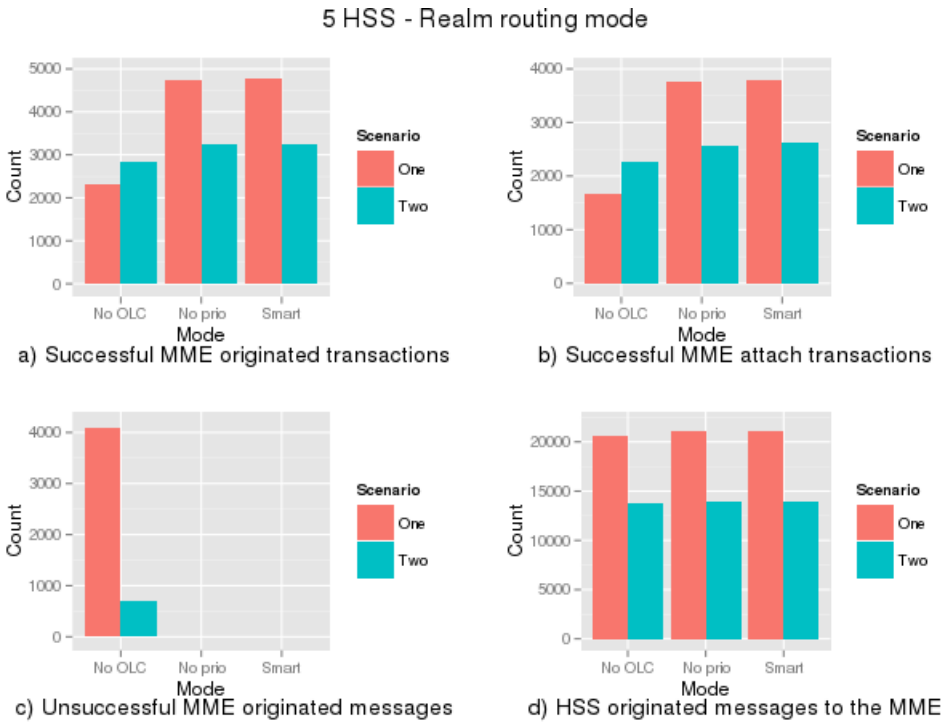

Figure 29: First modified scenarios in realm routing mode with five HSSs.

Table 9 shows the performance differences between the algorithm modes for this run. It is illustrative to compare it with Table 8 and the comparison only shows improvements of some 10 to 20 percent. The conclusion drawn from those results also applies here. The small differences between the indicator values suggest that having many HSSs of which only a few have more capacity does not increase the performance very much. More benefit would likely be provided if a more advanced message routing method was used.

Table 8: Relative performance changes between algorithm modes of the modified scenarios with five HSSs.

Performance indicator	**No OLC** [%]		**No prio** [%]		**Smart** [%]	
	No prio	**Smart**	**No OLC**	**Smart**	**No OLC**	**No prio**
Scenario one						
Direct routing mode						
All MME transactions	-23.4	-23.7	30.5	-0.4	31.0	0.4
MME attach transactions	-25.3	-26.0	33.8	-0.9	35.1	0.9
Unsuccessful MME messages	527.0	546.6	-84.1	3.1	-84.5	-3.0
HSS originated messages	26.2	4.2	-20.8	-17.5	-4.0	21.2
Realm routing mode						
All MME transactions	-51.4	-51.7	105.8	-0.5	106.9	0.5
MME attach transactions	-55.9	-56.2	126.5	-0.9	128.6	0.9
Unsuccessful MME messages	∞	∞	-100.0	0	-100.0	0
HSS originated messages	-2.3	-2.8	2.4	-0.5	2.9	0.5
Scenario two						
Direct routing mode						
All MME transactions	-5.0	-4.7	5.3	0.4	4.9	-0.4
MME attach transactions	-6.1	-4.8	6.5	1.3	5.1	-1.3
Unsuccessful MME messages	327.7	331.7	-76.6	0.9	-76.8	-0.9
HSS originated messages	6.9	2.1	-6.5	-4.5	-2.0	4.7
Realm routing mode						
All MME transactions	-11.8	-12.6	13.4	-0.9	14.4	0.9
MME attach transactions	-12.6	-13.8	14.0	-1.7	16.0	1.7
Unsuccessful MME messages	∞	∞	-100.0	0	-100.0	0
HSS originated messages	-0.3	-1.0	0.3	-0.8	1.1	0.8

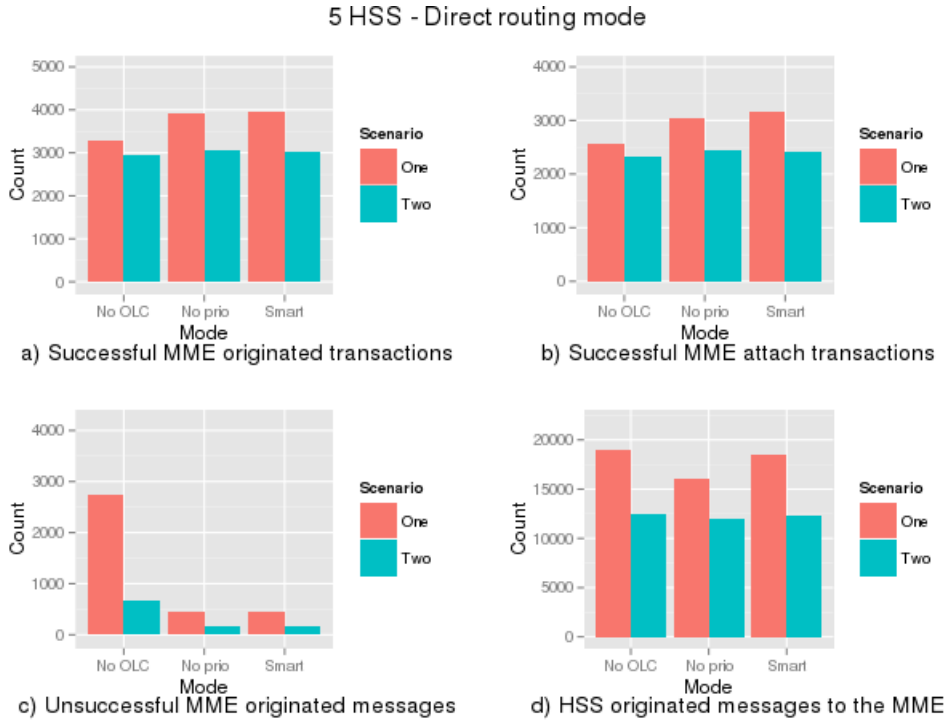

Figure 30: Second modified scenarios in direct routing mode with five HSSs.

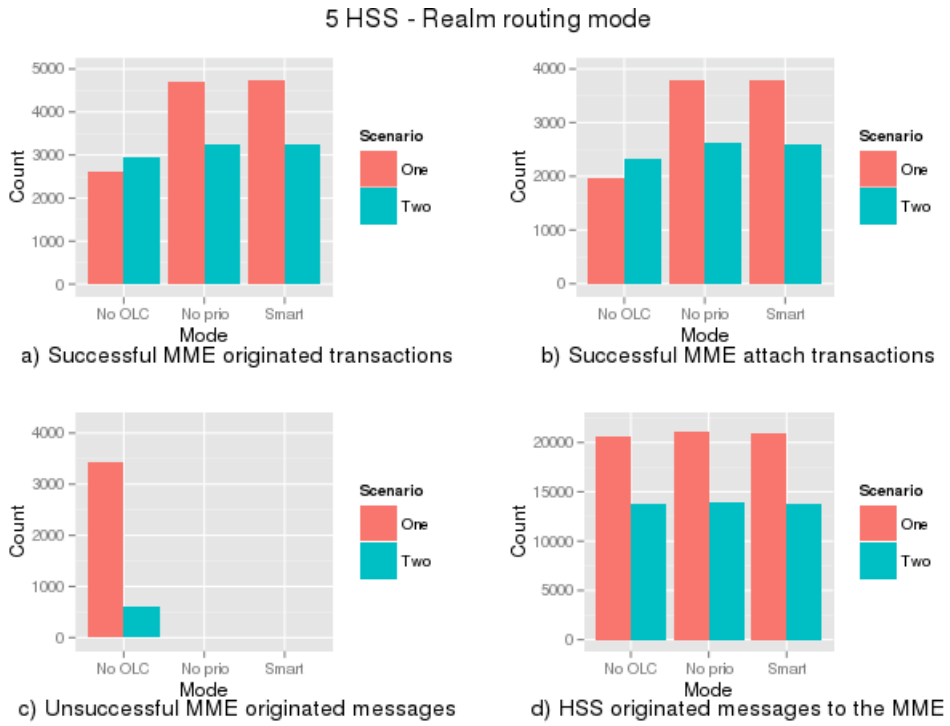

Figure 31: Second modified scenarios in realm routing mode with five HSSs.

Table 9: Relative performance changes between algorithm modes of the second modified scenario one with five HSSs.

Performance indicator	**No OLC** [%]		**No prio** [%]		**Smart** [%]	
	No prio	**Smart**	**No OLC**	**Smart**	**No OLC**	**No prio**
Scenario one						
Direct routing mode						
All MME transactions	-16.5	-17.4	19.7	-1.2	21.1	1.2
MME attach transactions	-15.7	-18.6	18.6	-3.5	22.9	3.6
Unsuccessful MME messages	516.6	530.7	-83.8	2.3	-84.1	-2.2
HSS originated messages	17.5	2.7	-14.9	-12.7	-2.6	14.5
Realm routing mode						
All MME transactions	-44.4	-44.7	80.0	-0.4	80.7	0.4
MME attach transactions	-48.4	-48.3	93.7	0.1	93.6	-0.1
Unsuccessful MME messages	∞	∞	-100.0	0	-100.0	0
HSS originated messages	-2.3	-2.0	2.4	0.3	2.1	-0.3
Scenario two						
Direct routing mode						
All MME transactions	-3.8	-3.4	4.0	0.4	3.5	-0.4
MME attach transactions	-3.4	-3.6	5.1	1.4	3.7	-1.4
Unsuccessful MME messages	319.7	287.6	-76.2	-7.6	-74.2	8.3
HSS originated messages	4.0	1.1	-3.9	-2.8	-1.1	2.9
Realm routing mode						
All MME transactions	-9.7	-10.0	10.8	-0.3	11.1	0.3
MME attach transactions	-11.3	-10.6	12.8	0.8	11.9	-0.8
Unsuccessful MME messages	∞	∞	-100.0	0	-100.0	0
HSS originated messages	-0.9	-0.4	0.9	0.5	0.4	-0.5

4.3 Ten HSSs

The last thing to do was to simulate the algorithm with ten HSSs. All HSSs have the capacity of 20 requests per second. The result of the simulation in direct routing mode is shown in Figure 32. The Figure shows that all routing and algorithm modes have a very similar performance. The only difference lies in the number of unsuccessful MME originated requests and even there it is very small compared to the other simulations. The realm routing mode has practically identical results for all indicators except HSS originated messages, having values of about 30 000 and 20 000 messages for scenario one and two respectively. There are also no unsuccessful MME originated messages. The results for this simulation run provide an upper bound for the achievable performance. The traffic handled by the HSSs cannot obviously exceed the amount sent by the MME.

It can be seen that having enough HSSs with a small capacity are able to serve the incoming traffic, which would overload a smaller number of entities having the same capacity. In other words, a brute force solution also works in this case, meaning that there is no need to have a sophisticated overload control solution when there is enough capacity from the start.

From the results of the two, five, and ten HSS simulation runs, one can draw the conclusion that the number of entities with a capacity of 20 requests per second, needed to handle the incoming requests lies between five and ten. Also having fewer entities with more capacity offers better performance than more entities having less capacity.

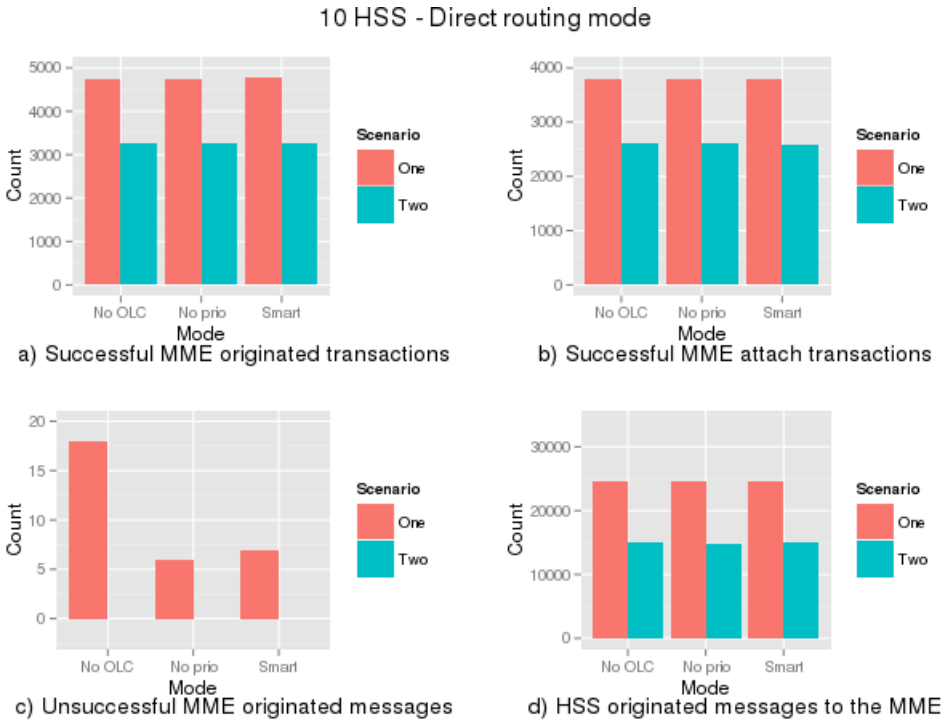

Figure 32: The two scenarios in direct routing mode with ten HSSs.

4.4 Theoretical calculations and comparison

In order to see whether the simulation has any bearing to the theory of queuing systems, the possibility to use an exponential distribution for service times was added. This makes it possible to see the simulator as a $M/M/n/n$ queueing system or an Erlang loss system [10]. For this system some key values such as blocking probability and throughput can be calculated and compared to actual simulation data. A similar scheme is suggested by and has been done in [23].

The initial case is to have two HSSs and thus a $M/M/2/2$ queueing system. The parameters of the models for scenario one and two are following: arrival rate $\lambda = 1/0.0181 \approx 55.25$ requests per second and service rate $\mu = 60$ requests per second and $\lambda = 1/0.292 \approx 34.25$ and $\mu = 60$ for scenario two. The previous values lead to traffic loads or utilization of $\rho = \frac{\lambda}{\mu} = \frac{55.25}{60} = 0.92$ and $\rho = \frac{\lambda}{\mu} = \frac{34.25}{60} = 0.57$ respectively [10].

The following calculations cannot directly be compared to the actual results presented earlier as the real simulation of the system is unstable, because $\lambda > \mu$ with $\lambda = 55$ and $\mu = 20$ requests per second. It is a conscious choice to use these values in the real simulation as its meaning is see its behaviour in exceptional circumstances causing heavy load. The point of these calculations is to see whether the simulation would behave according to theory when correct values are used.

The blocking or loss probability in an $M/M/n/n$ can be calculated with the Erlang B-formula [10]. Table 10 presents the blocking probabilities calculated with the Erlang B-formula. Only the no OLC algorithm mode is presented in the table because the blocking probability is much smaller in the other modes due to the algorithm. In scenario one, the calculated and measured values are quite close to each other having a change of 1.5 %. This difference is likely caused by the simulation software. For scenario two, the change is approximately 9 % and the calculated blocking probability is larger than the measured. Probable reasons for this difference are again the software as well as the smaller amount of traffic in scenario two.

The blocking probabilities of scenario two suggest that the simulator is able to utilize the capacity better than the theory would suggest. In order to test this, a simulation with $\rho = \frac{0.40}{60} = 0.42$ was done, which has the calculated blocking probability is 0.058 and measured probability was zero. This hints that the simulator gains more performance with a small ρ than the theory suggests.

Table 10: The calculated and measured blocking probabilities.

Routing mode	**Calculated**	**Measured**	**Abs. change** [%]
Scenario one			
Direct mode	0.181	0.183	0.2
Realm mode	0.181	0.196	1.5
Scenario two			
Direct mode	0.094	0.002	9.2
Realm mode	0.094	0.004	9

Another theoretical measure is the throughput[10], which for the simulation is the number of sent requests per second. In scenario one, the calculated throughput is 45.3 requests per second and in the direct routing and no OLC algorithm mode the approximate measured throughput is 53.8. The difference between these two values is 18.8 % which is over ten times more than with the blocking probabilities. As the definition of throughput in the simulation is not strictly the same as in literature, this measure is not very exact. In scenario two, the corresponding values are 31.0 and 33.6 requests per second with their difference being only 8.3 %.

For a comparison, the same calculations for a setup with five HSSs ($n = 5$) were also done having the same parameters as in the two HSS case. The calculated blocking probability is 0.002 for scenario one and 0.000 for scenario two. For all routing modes in scenario one and two the measured blocking probability is 0 with no blocked requests. The calculated throughput for scenario one in this case is 55.1 and the measured is 53.8 requests per second. In scenario two, the corresponding values are 34.2 and 33.6 requests per second. These numbers show that with a smaller load the simulator behaves more according to theory than with a higher load.

These calculations and measurements show that the simulator performs somewhat in a similar manner as the theory of $M/M/n/n$ queues would suggest. The actual results have variation caused by different values of λ in the two scenarios. The differences in the calculated and measured values are most likely caused by the simulation software. It can also be concluded that measured values are more in line with the theoretical values when the traffic load is smaller.

5 Discussion

Results of the two scenario simulations show the solution to improve the performance when it is used in comparison to it not being used. This observation is valid for all routing and algorithm modes. A further and rather obvious conclusion is that having more capacity leads to diminishing improvements and it completely disappearing, when enough capacity is available. A more detailed discussion of the two scenarios is presented next.

5.1 Scenario one

In scenario one, which has more subscribers and thus a higher load, the solution provides the largest improvement. In the base case with two HSSs having a capacity of 20 requests per second, an improvement of almost 10 000 % is achieved in the all MME originated transactions performance indicator for the direct routing mode. Other indicators have smaller changes and the no prioritization and smart algorithm modes perform better in three indicators with the exception being the HSS originated messages indicator. A comparison between the two routing modes shows better results for the realm routing mode, which is explained by the different HSS selection procedure used by the modes.

When extra capacity to one HSS is added, the MME originated transactions indicator only shows an improvement of 41 % for the smart algorithm mode. Other indicators show smaller improvements except the unsuccessful MME originated messages indicator, where the number of unsuccessful messages has increased. There are a few reasons for this phenomenon: the messages belonging to transactions use up all available capacity causing single messages to have non-successful results and the inability to use the heterogeneous capacities of HSSs. The latter reason is caused by the round-robin HSS selection, which cannot send more messages to HSSs with more capacity. Once there is enough capacity for all requests to be served, this phenomenon disappears. Finally, when both HSSs have their capacity boosted to 80 requests per second, the results are practically identical between all algorithm modes. This indicates enough capacity for the amount of traffic in question.

With five HSSs the base results are already much improved from the two HSS simulations. In comparison to the two HSS simulations, the values of the MME transaction and message indicators have improved by approximately two thirds and also the HSS originated messages indicator has practically the same value in both no prioritization and smart algorithm modes. In modified scenario one, the MME attach transaction indicators show an improvement of 10 to 15 % in the direct routing mode and the unsuccessful MME messages indicator again exhibits the same phenomenon as with two HSSs, that is, the value has increased when more capacity is added. The reason for this is the same as in the two HSSs simulations.

The second modification with further added capacity shows improvement of a few tens of percent. It is interesting to see that only now the count of unsuccessful MME originated messages has dropped to zero for the no prioritization and smart algorithm modes in the real routing mode. In the two HSS simulations, this already

happened with HSS capacities of 20 and 80 requests per second. The key point here is that even with more capacity all the algorithm modes do not achieve the same performance as they had in the two HSS simulations. In order to maximize performance, the easiest option is to have fewer HSSs with more capacity.

The last case is the ten HSS configuration in which all routing and algorithm modes have practically identical performance. So another option is to have more HSSs with less capacity, but due overhead as well as management and other issues having fewer HSSs with more capacity is likely easier.

5.2 Scenario two

Scenario two has fewer subscribers and thus a smaller load. In the two HSS simulations the trend is similar as in scenario one, meaning the no OLC mode again having a very poor performance except for the HSS originated messages indicator. Also the realm routing mode again outperforms the direct routing mode due to reasons explained earlier. When extra capacity is added in the modified scenario two the differences unsurprisingly decrease. In the second modified scenario all the algorithm modes have again very similar performance as expected with a static load.

The five HSS simulations show quite similar results for all performance indicators except the unsuccessful MME originated messages indicator. The indicator displays a steady increase in performance in all algorithm modes, but the no OLC mode never reaches zero, which it did in the two HSS simulations for this scenario. The similar results in all five HSS simulations demonstrate the natural fact, that the amount of served messages and transactions does not increase when more capacity is added, if the current capacity is sufficient to handle all the traffic. In these simulations the traffic sent by the MME is the limiting factor.

In the ten HSS simulation, all algorithm modes have identical results in each routing mode. This result is expected as the same behaviour can already be seen in scenario one.

5.3 Theoretical calculations

The simulator's adherence to theory was checked by using it to model a $M/M/n/n$ queueing system with $n = 2, 5$. Blocking probability and throughput were the measures used for testing. The results were mixed: blocking probabilities suggest scenario one having measurements, which are more in line with the theory. The throughput on the other hand indicates scenario two having results better fitting to the theory. Because of the contradictory results, it is impossible to say that either scenario fully complies to the theory or that both have no bearing to queueing theory. The natural explanation to this contradiction is the simulation software, which of course does not act like an ideal Erlang loss system.

6 Conclusions and further work

This thesis concerned the development of an overload control solution for traffic between the MME and HSS using the Diameter protocol. This solution is based on an algorithm using overload information sent by the HSSs to on demand throttle the traffic towards the overloaded entity. Additionally, message prioritization is employed to allow existing transactions to be completed during overload and thus decrease the load on the entity. The effectiveness and performance of this solution was evaluated with four performance indicators measuring different aspects of the system, such as, transaction and message successes.

The simulation results clearly show that the solution does in fact mitigate overload. The solution has been simulated with two scenarios and the results of these simulations display tangible improvement through performance measures. The main improvement takes place in cases with a constrained capacity compared to incoming traffic. It is most easily seen in the scenario one simulation with two HSSs. When additional capacity is added either through new HSSs or more capacity to existing ones is added, the performance improvement diminishes. The exact amount extra capacity needed naturally varies depending on the initial capacity and traffic load.

There are a few drawbacks in the solution. The first one is it not being able to take advantage of all the capacity in configurations with HSSs having varying capacities, because of the round-robin HSS selection mechanism. The second one is that prioritized messages in transactions use all the HSS capacity leading to non-successful results for MME originated messages not belonging to a transaction. Once HSSs have enough capacity to serve all incoming requests this issue disappears. This issue is not directly a drawback, because the solution works as designed and the issue vanishes as soon as enough capacity is provided.

6.1 Further work

The traffic rate based overload reporting mechanism described in the Diameter Overload Rate Control Internet-draft [14] has been discussed in this thesis. Implementing it would give the possibility to compare the performance of the two mechanisms and perhaps develop a hybrid percentage-rate based scheme.

The current simple HSS selection mechanism is limiting the full capacity usage in simulations with heterogeneous HSS capacities. A more advanced mechanism, which would take the overload status of each HSS into account when selecting a HSS in the direct routing mode, would make it possible to fully benefit from all system capacity. A drawback in this advanced mechanism is the need to do more processing during HSS selection. In networks having a large number of subscribers, this might lead to a degradation of the core network's performance. Being able to use all the capacity could on the other hand have a larger positive effect on the core network performance.

The current implementation does not use real Diameter library. Therefore implementing the solution with a real library would give a more realistic view on the solution's performance in a more real environment. A further improvement in the

same style is to include actual payload in the Diameter messages. It would show what effect message processing and other costs have on the performance. Once these improvements are in place, further simulations with varying parameters like traffic loads, HSS capacities, and mixed routing modes, could be done. It would give insight in how the solution behaves in various situations and allow improvements to be done as needed.

References

[1] 3GPP. *Evolved Packet System (EPS); Mobility Management Entity (MME) and Serving GPRS Support Node (SGSN) related interfaces based on Diameter protocol*. TS 29.272. 3rd Generation Partnership Project (3GPP), Dec. 2013. URL: http://www.3gpp.org/DynaReport/29272.htm.

[2] 3GPP. *General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (EUTRAN) access*. TS 23.401. 3rd Generation Partnership Project (3GPP), Mar. 2014. URL: http://www.3gpp.org/DynaReport/23401.htm.

[3] 3GPP. *International Mobile station Equipment Identities (IMEI)*. TS 22.016. 3rd Generation Partnership Project (3GPP), Sept. 2012. URL: http://www.3gpp.org/DynaReport/22016.htm.

[4] 3GPP. *Location Services (LCS); Evolved Packet Core (EPC) LCS Protocol (ELP) between the Gateway Mobile Location Centre (GMLC) and the Mobile Management Entity (MME); SLg interface*. TS 29.172. 3rd Generation Partnership Project (3GPP), Sept. 2013. URL: http://www.3gpp.org/DynaReport/29172.htm.

[5] 3GPP. *Network architecture*. TS 23.002. 3rd Generation Partnership Project (3GPP), June 2013. URL: http://www.3gpp.org/DynaReport/23002.htm.

[6] 3GPP. *Study on Core Network Overload (CNO) solutions*. TR 23.843. 3rd Generation Partnership Project (3GPP), Dec. 2013. URL: http://www.3gpp.org/DynaReport/23843.htm.

[7] 3GPP. *Study on Diameter overload control mechanisms*. TR 29.809. 3rd Generation Partnership Project (3GPP), Nov. 2013. URL: http://www.3gpp.org/DynaReport/29809.htm.

[8] 3GPP. *Universal Mobile Telecommunications System (UMTS); UTRAN overall description*. TS 25.401. 3rd Generation Partnership Project (3GPP), Oct. 2014. URL: http://www.3gpp.org/DynaReport/25401.htm.

[9] B. Aboba et al. *The Network Access Identifier*. RFC 4282 (Proposed Standard). Internet Engineering Task Force, Dec. 2005. URL: http://www.ietf.org/rfc/rfc4282.txt.

[10] Soeren Asmussen. *Applied Probability and Queues (Stochastic Modelling and Applied Probability)*. 2nd. Springer, May 2003. ISBN: 0387002111. URL: http://www.worldcat.org/isbn/0387002111.

[11] T. Bonald and M. Feuillet. *Network Performance Analysis*. ISTE/Wiley, July 2011. ISBN: 1118602854, 9781118602850.

[12] Gonzalo Camarillo and Miguel-Angel Garcia-Martin. *The 3G IP Multimedia Subsystem: Merging the Internet and the Cellular Worlds*. 3rd ed. Wiley Publishing, 2008. ISBN: 0470516623, 9780470516621.

[13] S. Donovan. *Diameter Agent Overload*. Internet Draft – work in procgess. Internet Engineering Task Force, Aug. 2014. URL: http://tools.ietf.org/id/draft-donovan-dime-agent-overload-02.txt.

[14] S. Donovan and E. Noel. *Diameter Overload Rate Control*. Internet Draft – work in procgess. Internet Engineering Task Force, Aug. 2014. URL: http://www.ietf.org/id/draft-donovan-dime-doc-rate-control-01.txt.

[15] V. Fajardo et al. *Diameter Base Protocol*. RFC 6733 (Proposed Standard). Updated by RFC 7075. Internet Engineering Task Force, Oct. 2012. URL: http://www.ietf.org/rfc/rfc6733.txt.

[16] Sally Floyd and Van Jacobson. "Random Early Detection Gateways for Congestion Avoidance". In: *IEEE/ACM Trans. Netw.* 1.4 (Aug. 1993), pp. 397–413. ISSN: 1063-6692. DOI: 10.1109/90.251892. URL: http://dx.doi.org/10.1109/90.251892.

[17] S. Kasera et al. "Fast and robust signaling overload control". In: *Proceedings of IEEE International Conference on Networks Protocol (ICNP)*. 2001, pp. 323–331.

[18] S. Kasera et al. "Robust multiclass signaling overload control". In: *Proceedings of IEEE International Conference on Networks Protocol (ICNP)*. 2005.

[19] J. Korhonen et al. *Diameter Overload Indication Conveyance*. Internet Draft – work in procgess. Internet Engineering Task Force, Oct. 2014. URL: http://www.ietf.org/id/draft-ietf-dime-ovli-04.txt.

[20] D. Nelson and A. DeKok. *Common Remote Authentication Dial In User Service (RADIUS) Implementation Issues and Suggested Fixes*. RFC 5080 (Proposed Standard). Internet Engineering Task Force, Dec. 2007. URL: http://www.ietf.org/rfc/rfc5080.txt.

[21] Magnus Olsson and Catherine Mulligan. *EPC and 4G packet networks: driving the mobile broadband revolution*. 2nd ed. Oxford: Academic Press, 2012. ISBN: 012394595X, 9780123945952.

[22] J.H. Schiller. *Mobile Communications*. 2nd ed. Addison-Wesley, 2003. ISBN: 9780321123817.

[23] Tarik Taleb, Nei Kato, and Yoshiaki Nemoto. "Neighbors-buffering-based video-on-demand architecture". In: *Signal Processing: Image Communication* 18.7 (2003), pp. 515 –526. ISSN: 0923-5965. DOI: http://dx.doi.org/10.1016/S0923-5965(03)00039-0. URL: http://www.sciencedirect.com/science/article/pii/S0923596503000390.