STACO - An Accounting Configuration Architecture for Multi-Service Mobile Networks

Racz, P ; Stiller, B

Abstract: Accounting is a key task in commercial networks. With the increasing number of IP-based services and mobility support, accounting needs to evolve towards an integrated, service-oriented accounting approach in a mobile environment. Therefore, this dissertation digest paper presents the Serviceoriented Tailored Accounting Configuration (STACO) architecture that enables a service-oriented accounting configuration management in a mobile, multi-domain networking environment. Additionally, it presents the Diameter flow accounting application as an extension to the Diameter protocol in order to integrate IP flow accounting into any Diameter-based infrastructure and to support an efficient transfer of IP flow records. Keywords–Accounting, Accounting Configuration, Mobility, IP Flow Accounting, Diameter, IPFIX.

DOI: https://doi.org/10.1109/INM.2009.5188886

STACO – An Accounting Configuration Architecture for Multi-Service Mobile Networks

Peter Racz, Burkhard Stiller
Department of Informatics IFI, University of Zurich
Binzmühlestrasse 14, CH-8050, Zurich, Switzerland
[racz|stiller]@ifi.uzh.ch

***Abstract*–Accounting is a key task in commercial networks. With the increasing number of IP-based services and mobility support, accounting needs to evolve towards an integrated, service-oriented accounting approach in a mobile environment. Therefore, this dissertation digest paper presents the Service-oriented Tailored Accounting Configuration (STACO) architecture that enables a service-oriented accounting configuration management in a mobile, multi-domain networking environment. Additionally, it presents the Diameter flow accounting application as an extension to the Diameter protocol in order to integrate IP flow accounting into any Diameter-based infrastructure and to support an efficient transfer of IP flow records.**

***Keywords*–Accounting, Accounting Configuration, Mobility, IP Flow Accounting, Diameter, IPFIX.**

I. Introduction

In commercial network environments accounting plays a central role, where accounting determines the collection and aggregation of information on service consumption, resource usage, and system activities. Accounting data can be used for various purposes, like charging the user for service consumption, collecting traffic traces for network management and traffic analysis, or deriving traffic characteristics for trend analysis and capacity planning.

The Service-oriented Tailored Accounting Configuration (STACO) architecture developed in [9] provides means in order to configure the accounting process in a customized manner. It supports service-oriented accounting with a hierarchical accounting session structure and enables accounting in mobile environments, where user, device, and session mobility occur. The STACO protocol defines the communication between components of the architecture and it is completely specified based on the Diameter protocol. In order to integrate IP flow accounting in a Diameter-based accounting infrastructure and to support the efficient transfer of IP flow records, the Diameter flow accounting application developed in [9] provides a template-based record transfer for IP flow records. It is compatible with the IP Flow Information Export (IPFIX) [2] protocol and it is specified as a new Diameter protocol extension, enabling an easy integration and providing compatibility with the Diameter base protocol. Publications related to [9] address service-oriented accounting [14], [6], an accounting configuration architecture and protocol [3], [4], [5], mobility aspects in accounting [10], [12], and charging support [11], [13].

The remainder of the paper is structured as follows. Section II. gives the motivation and discusses the problem statement. Section III. introduces the STACO architecture and describes its components, the accounting session model, and the accounting context. While Section IV. presents the STACO protocol. The Diameter flow accounting application is presented in Section V. Finally, Section VI. summarizes and concludes the paper.

II. Motivation and Problem Statement

IP networks became the main infrastructure for computer networks and for various information and communication services. The Internet is today a multi-service environment with both free-of-charge and commercial services, where services, valuable for a user, are formed by various services offered by the same or different providers, resulting in service composition. For example, users can access the network via their Internet Service Provider (ISP), browse the web, watch a movie from a Video-on-Demand (VoD) provider, or make phone calls over the Internet. In this example, the user has several parallel services running, where some of them might be correlated, e.g., the video stream and the network access with a certain Quality of Service (QoS). Thus, several different services build a complete end-user service. Additionally, in service provisioning mobility becomes more important, making user, device, and session mobility highly relevant.

Due to this evolution and the increasing number of IP-based services, accounting in IP networks faces new challenges and needs to evolve from the traditional network-oriented view towards a service-oriented approach in a mobile environment. However, current IP accounting architectures and protocols, like the generic Authentication, Authorization, and Accounting (AAA) architecture [7], the Remote Authentication Dial In User Service (RADIUS) [15], the Diameter [1], and the IPFIX [2] protocols, do not meet all these requirements. Current approaches consider only access and network layer services and do not fully integrate higher level and application services. They cannot efficiently cope with a highly dynamic environment, which is mainly due to the pre-defined and static configuration of the accounting process. In a summary, current accounting approaches for IP-based services lack multi-service and composed services support, dynamic accounting configurability, and mobility-awareness.

Therefore, the aim of the STACO architecture is to provide a generic accounting architecture for IP networks that supports the dynamic setup and configuration of accounting in a multi-service, mobile networking environment, enabling the collection of tailored accounting data in an integrated manner. Key challenges addressed by the STACO architecture and the Diameter flow accounting application are the following.

978-1-4244-3487-9/09/$25.00

Service-oriented accounting: The accounting architecture has to consider manifold services, including also application level services, and not only access and IP packet transport services. In case of a Voice-over-IP (VoIP) call for example, the accounting architecture has to support the collection of accounting data related to the network access, the QoS connection, and the VoIP call and it has to correlate these data in order to allow accounting for the single telephony service.

Tailored accounting configuration: Since accounting has very different objectives, the possible configuration of accounting is also manifold. In case of charging for example, accounting depends on the applied charging scheme, which might be in turn different per user and service, and it might also change frequently in time. Therefore, the accounting architecture has to support accounting configuration setup and management, tailored to the given objective of accounting. A tailored accounting configuration reduces accounting overhead and the amount of spare data collected, since only accounting data required by subsequent processes are gathered.

Mobility-awareness in accounting: The accounting architecture has to be capable to deal with user, device, and session mobility, enabling that a running service can be accounted for as a single service despite of handovers and session transfers. The main challenge here is that accounting data are originating from several network components, which are also changing because of mobility. Thus, accounting records have to be correlated to a single service session.

Integrated accounting architecture: The architecture has to support accounting for different purposes in different scenarios in an integrated manner, e.g., accounting for charging purposes, or IP flow accounting for traffic analysis purposes.

III. STACO Architecture

The STACO architecture is based on the generic AAA architecture [7]. The network components defined by the STACO architecture are illustrated in Figure 1. Authentication, Authorization, Accounting, Auditing, and Charging (A4C) servers are enhanced AAA servers and they play a central role, since they are responsible for user authentication, service access authorization, accounting for service usage, auditing of service consumption, and charging for accessed services. However, auditing and charging are out of the scope of this paper. A4C servers manage the accounting process and collect accounting records. Within the network of a single provider several A4C servers are most likely deployed for, e.g., load balancing and redundancy reasons.

Service Equipments (SE) provide the service itself for users. An SE can be for example a web server, a Video-on-Demand server, or a router in the network. SEs meter the service usage and report accounting data to A4C servers. An SE is always assigned to an A4C server, to which it sends its requests.

The A4C Database contains all information required for the accounting and business processes of a provider, including SLAs, user profiles, service profiles, and accounting configuration. Additionally, the database also stores accounting records.

The communication between A4C servers and SEs is defined by the STACO protocol that is a new extension to the Diameter protocol and specifies new Diameter commands and Attribute-Value-Pairs (AVP) in order to support accounting configuration.

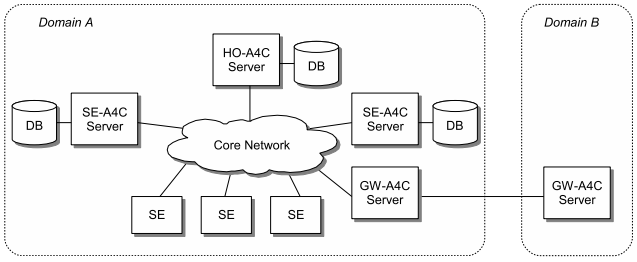

Fig. 1. STACO network components

A. A4C Server Roles

A4C servers play different roles according to their tasks in an accounting process. Roles determine the functionality a server provides in a certain network configuration and according to running accounting sessions.

HO-A4C server role: The Home A4C server (HO-A4C) is always located in the home domain of a user and it is assigned to a set of users. The HO-A4C server holds user profiles and credentials required for authentication and authorization. Additionally, the HO-A4C server holds also the accounting configuration, specifying the accounting process for a certain service and user. It also receives and stores all accounting data of the set of users it is responsible for.

SE-A4C server role: The Service Equipment A4C server (SE-A4C) serves as an administrator of some given SEs. It is deployed physically close to the SEs assigned to it, enabling to keep configuration effort close to the location of the requested service. It coordinates, configures, and manages the accounting process on assigned SEs, including the generation of session identifiers, setting up metering on the SE, and collecting and forwarding accounting data.

GW-A4C server role: The Gateway A4C server (GW-A4C) forwards messages between domains. It is a Diameter proxy agent [1]. GW-A4C servers are deployed in a network in order to reduce the number of interfacing A4C servers and the number of A4C connections to be configured between domains.

B. Accounting Session Model

The accounting session model defines accounting sessions, the accounting session structure, the relation between services and accounting sessions, and the creation and termination of accounting sessions. A session is in general a logical relation between multiple activities and events that are bound together. In the accounting session model, service sessions and accounting sessions are differentiated. A service session represents an instance of a running service, while an accounting session represents an instance of a running accounting process and binds accounting related events and accounting records together.

The accounting session structure represents the logical linking of several accounting sessions. It has a hierarchical structure represented by a tree. According to the hierarchy there are parent and child sessions, while the root of the tree represents the root session. Accounting sessions are identified by unique accounting session IDs (SID). The session structure enables the

identification of sessions in the accounting process and it makes possible to correlate accounting data originating from different SEs. The accounting session structure is derived from the desired accounting process, accounting granularity, and the topology of the accounting infrastructure. There are accounting sessions on every SE participating in accounting and on their assigned SE-A4C servers. Additionally, accounting root sessions are on the HO-A4C server.

Accounting sessions are created and terminated on SEs and A4C servers in order to maintain the accounting session structure. An SE starts a new accounting session, when it starts providing a service and terminates it, when the service is stopped. Accounting sessions on A4C servers are started when servers are notified on the start of a child accounting session on SEs or other A4C servers. Accounting sessions on A4C servers are terminated after the termination of the last child session.

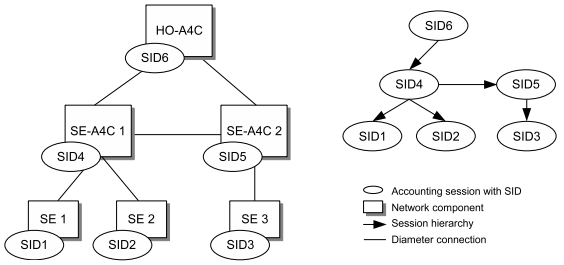

Fig. 2. Accounting session example

To illustrate the accounting session structure, Figure 2 shows an example network topology on the left hand side. The network consists of one HO-A4C server, two SE-A4C servers, and three SEs. It is assumed that a service on SE1 is started and the service consists of two subservices running on SE2 and SE3. The resulting accounting session structure is shown in Figure 2 on the right hand side. Since all three SEs perform accounting, there is a separate accounting session on all three SEs (SID1, SID2, and SID3). SE1 and SE2 are assigned to SE-A4C1, while SE3 to SE-A4C2. Therefore, the parent accounting session SID4 is running on SE-A4C1 for SID1 and SID2, and the parent session SID5 on SE-A4C2 for SID3. Since the service running on SE3 is a subservice of the service running on SE1, SID4 is the parent session of SID5. Finally, the root session SID6 is running on the HO-A4C server.

C. Accounting Context

The STACO architecture introduces the concept of accounting context. The accounting context contains all information required on a network component in order to perform accounting. It is distributed and dynamically established on A4C servers and SEs during the accounting configuration process.

The accounting context contains the accounting session structure, i.e. the accounting SIDs and their interrelation according to the accounting session hierarchy. It also contains information about the traffic to be metered for a given accounting session, together with the specification of accounting attributes to be collected, e.g., time, number of bytes or packets.

D. Accounting Configuration

The STACO architecture enables a tailored accounting configuration. In case the accounting process is triggered, e.g., a user requests a service, the accounting configuration is retrieved from the HO-A4C and SEs involved in the accounting process are configured by the A4C server infrastructure. Afterwards, SEs deliver tailored accounting data on service consumption that is aggregated by the HO-A4C server.

During the accounting configuration, the accounting context is established on A4C servers and SEs in a way that the accounting context is kept on A4C servers physically close to the SE. This enables to improve the performance of a possible accounting reconfiguration. In case of mobility for example, when the mobile terminal changes the SE it is attached to, the accounting process has to be updated. Since the accounting context is already stored by an SE-A4C server close to the new SE, the accounting reconfiguration is performed in a local scope. This reduces configuration overhead and improves the handover performance.

IV. STACO Protocol

The STACO protocol defines all interactions between SEs and A4C servers for a service-oriented dynamic accounting configuration. It is specified as a new extension for the Diameter protocol by introducing new Diameter commands and AVPs. Table I summarizes the newly defined commands, while their formal specification can be found in [9].

TABLE I
STACO Diameter commands

Command name	Description
Authentication-Authorization	It is used to authenticate a user and authorize the service request.
Context-Transfer	It is used to retrieve the accounting context and authentication information from an A4C server.
Accounting-Notification	It is used to notify A4C servers about changes in the accounting session structure.
SE-Configure	It is used to configure accounting and metering on SEs.
Data-Report	It is used to send accounting data to A4C servers.

The operation of the STACO protocol is illustrated by three selected example use cases, namely network connection setup, intra-A4C handover, and inter-A4C handover. The network topology used in these use cases is as shown in Figure 2, while the SEs are access points (AP) now.

The message sequence chart for the network connection setup use case is shown in Figure 3. The user wants to connect via AP1 to the network. Therefore, the MT requests the network access service from AP1 via the Service-Request message, containing username, authentication credentials, and the service ID. Note, that the Service-Request and Service-Answer are not defined by the STACO architecture and they can represent any application-specific protocol messages. AP1 sends an Authentication-Authorization-Request (AAR) to its associated A4C server, i.e. SE-A4C1. Since SE-A4C1 does not have any

information about the user yet to decide on authentication and authorization, it sends a Context-Transfer-Request (CTR) to the HO-A4C server, which holds the user profile. The message is routed by the Diameter protocol based on the username to the right HO-A4C. When the HO-A4C server receives the request, it creates the root accounting session, identified by SID1, and replies with a Context-Transfer-Answer (CTA), containing the SID1 as parent SID, the user profile with the authentication credentials, and the accounting configuration. Based on the information in the CTA message, the SE-A4C1 server authenticates the user and authorizes the network access. Afterwards, it stores SID1 in its accounting context and generates SID2 that will be used by the APs associated to the server as the parent session ID. SE-A4C1 informs AP1 about the AA decision via the AA-Answer (AAA) message and sends back SID2. Note, that Figure 3 shows a simplified authentication process, where a single AAR-AAA message pair is exchanged, however, the STACO architecture supports multi-round authentication as well.

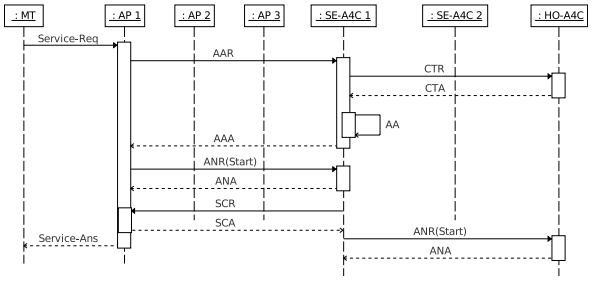

Fig. 3. Network connection setup

After receiving the AAA message, AP1 generates a new SID (APSID1) for the accounting data collected by AP1 and informs SE-A4C1 via the Accounting-Notification-Request (ANR) message that contains the newly created APSID1. SE-A4C1 stores APSID1 in its context information and sends an Accounting-Notification-Answer (ANA) back as an acknowledgement. SE-A4C1 configures accounting on AP1 via the SE-Configure-Request (SCR) message that is acknowledged by AP1 via the SE-Configuration-Answer (SCA) message. Afterwards, AP1 grants network access and informs the MT via the Service-Answer message. The message contains SID2 that is used as a reference to the accounting process in subsequent requests in case of handover. Additionally, SE-A4C1 informs the HO-A4C about the new session (SID2) via the ANR message that is acknowledged by the ANA message.

In the intra-A4C handover use case the MT performs a handover from AP1 to AP2, where an intra-A4C handover means that the SEs serving the user before and after the handover are associated to the same SE-A4C server. Thus, the SE-A4C server does not change during the handover and it can configure accounting on the new SE based on the already locally available accounting context information. The message sequence chart for the intra-A4C handover use case is shown in Figure 4. Since SE-A4C1 already holds the context information for the user, it can authenticate and authorize the user locally. Based on SID2 it can associate the new session to the already existing parent session. Afterwards, the sequence of actions is identical to the network connection setup use case. Additionally, accounting is terminated on the old AP (AP1) and accounting data is transferred via the Data-Report-Request (DRR) to the server. Finally, AP1 informs the server that the accounting session is terminated via the ANR message.

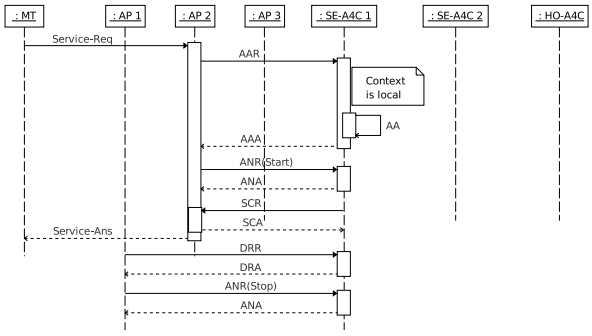

Fig. 4. Intra-A4C handover

In case of an inter-A4C handover the SEs providing the service before and after the handover are associated to different SE-A4C servers. Therefore, the accounting context has to be transferred to the new SE-A4C server. In the inter-A4C handover use case the MT performs a handover from AP2 to AP3, which are associated to SE-A4C1 and SE-A4C2, respectively. The message sequence chart for the inter-A4C handover use case is shown in Figure 5. The message sequence is very similar to the previous use cases. But since SE-A4C2 does not have any context information, it request the accounting context from SE-A4C1 via the CTR message. After the context information is transferred, SE-A4C2 can perform the authentication and authorization and it can configure accounting on AP3 similar to the previous use cases.

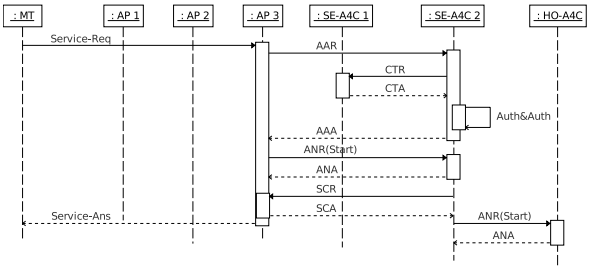

Fig. 5. Inter-A4C handover

V. Diameter-based Flow Accounting

In order to integrate IP flow accounting into a Diameter-based accounting infrastructure, the Diameter flow accounting application defines new extensions and IPFIX-compatible record formats for the Diameter protocol. The Diameter flow accounting application can be used in any Diameter-based accounting infrastructure, such as the STACO architecture or the traditional AAA architecture, as shown in Figure 6.

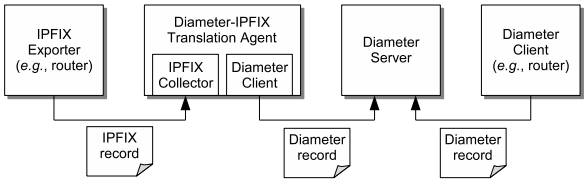

Fig. 6. Network components

The Diameter server acts as an accounting server, collects IP flow records, and stores them in a local database for further processing. Network devices, like routers, gather flow information by observing the traffic locally and send IP flow records to the server. The Diameter client represents a device that natively supports the Diameter flow accounting application, while the IPFIX exporter represents a device supporting IPFIX-based flow record transfer. The Diameter-IPFIX translation agent enables the integration of IPFIX exporters into a Diameter-based infrastructure. It translates IPFIX flow records into Diameter flow records and enables the communication between native IPFIX exporters and a Diameter server. Additionally, the translation agent can filter and aggregate flow records before forwarding them to the server.

The IPFIX exporter and the translation agent communicate via the IPFIX protocol and they exchange IPFIX records. The Diameter client and the translation agent communicate with the Diameter server via the Diameter protocol, exchanging flow records based on Diameter AVPs, as specified below.

A. Record Formats

The Diameter flow accounting application defines three different record types, namely the template record, the option template record, and the flow data record. All three record formats are based on the IPFIX specification [2], [8] in order to provide compatibility. The template record specifies the structure and semantics of a data record and it contains a sequence of type-length pairs, specifying all attributes of the data record. The option template record defines a template for a data record together with additional information, like the flow key, filtering information, or sampling parameters. The data record is used to transfer flow records and it contains attribute values according to its associated template record.

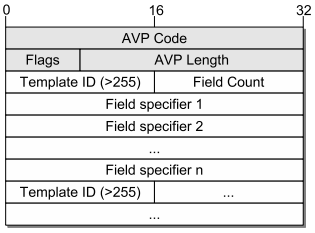

Fig. 7. Diameter template record AVP

The Diameter flow accounting application defines new AVPs for these three record types. The AVP format for the template record is shown in Figure 7. It consists out of the AVP header, containing the AVP code, flags, and AVP length fields, followed by the AVP value. The AVP value shows a structured format, specifying template records. An AVP can include several template records. The template record is identified by the Template ID field which is numbered from 256 for data records [2]. The Field Count specifies the number of fields in the template record. A sequence of Field Specifiers defines information elements, as shown in Figure 8, specifying the attributes of a data record. An information element consists of the Information Element ID, specifying the attribute, and the Field Length, specifying the length of the attribute. If the information element is vendor-specific, it includes

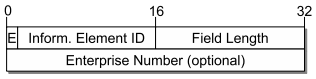

Fig. 8. Field specifier

the optional Enterprise Number and the E bit is set to one. The set of information elements defined by IETF are specified in [8].

The AVP format for the option template record is shown in Figure 9. The AVP value has a structured format similar to the template record AVP and it contains one or more option template records. The Template ID defines the template identifier. The Field Count contains the number of fields in the option template record, including also the number of scope fields. The Scope Field Count specifies the number of scope fields. Both normal fields and scope fields have the same format as shown in Figure 8. The AVP also contains an optional padding field.

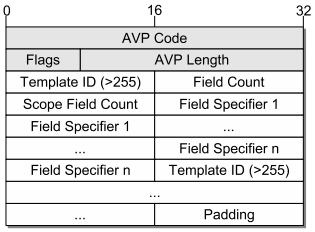

Fig. 9. Diameter option template record AVP

Finally, Figure 10 shows the AVP format for the flow data record. The AVP value has again a structured format and it contains one or more flow data records. The Template ID specifies the template to be used to interpret the flow data record. The template is transferred in the template record or in the option template record as specified above. The Length field specifies the length of the data set that includes one or more data records. The attribute values of flow records are included in the sequence of Field Values. The length of possible field values are determined by their data types specified in [8]. The AVP also contains an optional padding field.

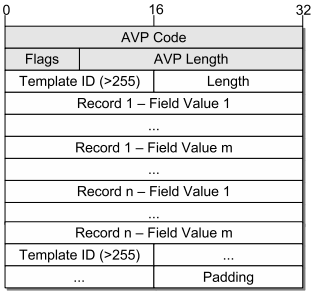

Fig. 10. Diameter flow data record AVP

B. Diameter Commands and AVPs

The Diameter flow accounting application defines new AVPs and specifies their integration into Diameter accounting messages. The new AVPs can be included in the Accounting-Request, specified in [1], as well as in the Data-Report-Request, specified by the STACO architecture above.

The new AVPs are summarized in Table II. The Accounting-Record-Type AVP is extended with a new value specify-

ing flow accounting records. The IPFIX-Version-Number, the IPFIX-Export-Time, the IPFIX-Sequence-Number, and the IPFIX-Source-ID AVPs carry IPFIX header fields and are used by the translation agent when converting IPFIX messages to Diameter messages. Between a Diameter client and server these AVPs are not used. The IPFIX-Template-Record, the IPFIX-Option-Template-Record, and the IPFIX-Data-Record AVPs include template, option template, and data records, respectively.

TABLE II
Diameter flow accounting AVPs

AVP name	AVP type	Description
Accounting-Record-Type	Enumerated	It defines the type of the accounting record [1]. The Diameter flow accounting application defines the new record type FLOW_RECORD, which is used to indicate flow accounting records.
IPFIX-Version-Number	Unsigned32	It contains the version field of the IPFIX packet that defines the version of the record format [2].
IPFIX-Export-Time	Unsigned32	It contains the time at which the IPFIX exporter sent the record as defined in [2].
IPFIX-Sequence-Number	Unsigned32	It is an incremental sequence number of data records as specified in [2].
IPFIX-Source-ID	Unsigned32	It contains the identifier of the IPFIX observation domain as specified in [2].
IPFIX-Template-Record	OctetString	It contains template records according to the format specified above.
IPFIX-Option-Template-Record	OctetString	It contains option template records according to the format specified above.
IPFIX-Data-Record	OctetString	It contains data records according to the format specified above.

The Accounting-Request command is extended with the IPFIX-Version-Number, the IPFIX-Export-Time, the IPFIX-Sequence-Number, the IPFIX-Source-ID, the IPFIX-Template-Record, the IPFIX-Data-Record, and the IPFIX-Option-Template-Record optional AVPs. The IPFIX-Template-Record, the IPFIX-Data-Record, and the IPFIX-Option-Template-Record AVPs can be included several times in the message to transfer several records.

VI. Summary and Conclusions

The STACO architecture enables a customized accounting configuration and supports service-oriented accounting with a hierarchical accounting session structure in mobile environments. It has been completely specified as a Diameter extension, defining new commands and AVPs. For the integration of IP flow accounting into a Diameter-based accounting infrastructure, the Diameter flow accounting application supports an efficient template-based record transfer for IP flow records, while it is compatible with IPFIX. Both the STACO architecture and the Diameter flow accounting application have been implemented as a prototype and an analytical and experimental evaluation based on the prototype have been performed [9].

The evaluation of the STACO architecture shows that the extended functionality of accounting configuration is feasible, that the STACO architecture achieves suitable performance for the accounting configuration, and that it includes performance enhancement mechanisms in case of mobility. The evaluation based on the prototypical implementation shows that the Diameter flow accounting application achieves a better performance for IP flow record transfer than the standard Diameter accounting approach.

References

[1] P. Calhoun, J. Loughney, E. Guttman, G. Zorn, J. Arkko: Diameter Base Protocol; IETF, RFC 3588, September 2003.

[2] B. Claise (Ed.): Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information; IETF, RFC 5101, January 2008.

[3] F. Eyermann, P. Racz, B. Stiller, C. Schaefer, and T. Walter. Generic Accounting Configuration Management for Heterogeneous Mobile Networks. Third ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots (WMASH 2005), Cologne, Germany, 2005.

[4] F. Eyermann, P. Racz, B. Stiller, C. Schaefer, and T. Walter. Service-oriented Accounting Configuration Management based on Diameter. First IEEE International Workshop on Performance and Management of Wireless and Mobile Networks (P2MNet 2005), Sydney, Australia, 2005.

[5] F. Eyermann, P. Racz, B. Stiller, C. Schaefer, and T. Walter. Diameter-based Accounting Management for Wireless Services. IEEE Wireless Communications and Networking Conference (WCNC 2006), Las Vegas, NV, USA, 2006.

[6] Hasan, P. Racz, C. Morariu, D. Hausheer, and B. Stiller. A4C Support for Commercialization of Next Generation Grid Services. European Research Consortium for Informatics and Mathematics (ERCIM) News, No. 70:18–19, July 2007.

[7] C. de Laat, G. Gross, L. Gommans, J. Vollbrecht, D. Spence: Generic AAA Architecture; IETF, RFC 2903, August 2000.

[8] J. Quittek, S. Bryant, B. Claise, P. Aitken, J. Meyer: Information Model for IP Flow Information Export; IETF, RFC 5102, January 2008.

[9] P. Racz. A Generic Accounting Configuration Architecture for Multi-Service Mobile Networks. Doctoral Thesis, University of Zurich, June 2008. http://www.csg.uzh.ch/staff/racz/extern/Dissertation.pdf

[10] P. Racz, J. E. Burgos, N. Inacio, C. Morariu, V. Olmedo, V. Villagra, R. L. Aguiar, and B. Stiller. Mobility and QoS Support for a Commercial Mobile Grid in Akogrimo. In Proc. of the 16th IST Mobile and Wireless Communications Summit, Budapest, Hungary, July 1-5, 2007.

[11] P. Racz, F. Eyermann, and B. Stiller. Statistical Traffic Sampling Methods for Network Management and Inter-Domain Charging. 10th European Summer School (EUNICE 2004), Tampere, Finland, 2004.

[12] P. Racz, V. Förster, and B. Stiller. A Design and Implementation of an Integrated Accounting Architecture for Distributed UMTS and WLAN Networks. International Journal of Wireless and Mobile Computing (IJWMC), Vol. 2(4):275–287, 2007.

[13] P. Racz and B. Stiller. Content-based Charging Support for Multiple Interworking Providers. IEEE Workshop on Local and Metropolitan Area Networks (LANMAN 2004), Mill Valley, CA, USA, 2004.

[14] P. Racz and B. Stiller. A Service Model and Architecture in Support of IP Service Accounting. 10th IEEE/IFIP Network Operations and Management Symposium (NOMS 2006), Vancouver, BC, Canada, 2006.

[15] C. Rigney, S. Willens, A. Rubens, W. Simpson: Remote Authentication Dial In User Service (RADIUS); IETF, RFC 2865, June 2000.