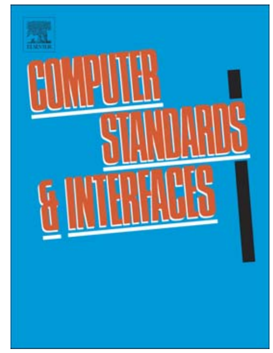
Accepted Manuscript



Extending SIP to support payments in a generic way

Antonio Ruiz-Martínez, Juan A. Sánchez-Laguna, Antonio F. Skarmeta-Gómez

PII: S0920-5489(16)00010-6
DOI: doi: 10.1016/j.csi.2016.01.002
Reference: CSI 3089

To appear in: *Computer Standards & Interfaces*

Received date: 12 August 2015
Revised date: 11 January 2016
Accepted date: 13 January 2016

Please cite this article as: Antonio Ruiz-Martínez, Juan A. Sánchez-Laguna, Antonio F. Skarmeta-Gómez, Extending SIP to support payments in a generic way, *Computer Standards & Interfaces* (2016), doi: 10.1016/j.csi.2016.01.002

Extending SIP to support payments in a generic way

Antonio Ruiz-Martínez, Juan A. Sánchez-Laguna,
Antonio F. Skarmeta-Gómez
Department of Information and Communications Engineering
Faculty of Computer Sciences
Campus of Espinardo
30100 Murcia, Spain
email: {arm,jlaguna,skarmeta}@um.es

Electronic payments, payment framework, SIP, micropayments, payments protocols, negotiation

1. Introduction

Some widely used Internet services such as voice calls, video conferencing, video on demand, instant messages and so on [1, 2, 3] rely on Session Initiation Protocol (SIP) [4]. Service providers offer these and other services for free, on a subscription basis, or on a pay-per-use model. Usually, free services are limited or a mean to attract customers because, obviously, the ultimate goal of any company is to make money.

Vendor's most preferred option is subscription because it guarantees fixed revenues during a defined period of time independently of the customer's real use of the service. However users tend to prefer the pay-per-use model [5, 6] so that they pay only for what they really use.

In order to gain or retain customers, some vendors offer the pay-per-use model on a pre-paid billing or billing basis [7, 8]. However, these methods introduce some drawbacks. The main problem of these proprietary mechanisms is that users must have an account with each different provider.

The use of payment mechanisms instead of billing ones eliminates the need of a previous registration with the payment provider [9, 10]. Thus, Fischl et al. [11, 12] proposed an interesting solution based on SIP. Their solution allows charging for accessing to services, micro billing and, obviously, being base don SIP it is suitable for mobile payment services [13].

However, Fischl et al.'s proposal has two limitations. Firstly, the protocol is not efficient, and secondly its proposal does not completely support macro-payments. In a previous work [14] we show how to improve the efficiency of the protocol. In this paper we propose a new improvement of the protocol that supports both micro and macro payments in a generic way.

Our proposal is generic, that is, any combination of brands, protocols and payment providers is allowed. The idea of facilitating payment interoperability is also being considered for other scenarios such as making payments on the Web [15, 16, 17]. Groups such as the W3C Web Payment Interest Group [18] are working to define generic standards across the e-payment industry. Moreover, in a recent article [19], Jeff Jaffe, CEO of the World Wide, talked about the benefits of supporting different payment mechanisms and how that can enable competition and innovation in Web payments, preventing possible payment vendor monopoly and vendor lock-in. We believe that this is also true for SIP payments.

Additionally, our proposal is based on the standard extensibility mechanisms defined by SIP [4, 20]. We only define new headers, tags and contents for the body part of the message making our extension lightweight, and truly compatible. Moreover, another advantage of integrating the payment process using a SIP extension mechanism is that we meet the requirements of DRM and e-commerce systems [21, 22].

Our proposal also supports negotiation through the so-called offer/answer model [23, 24, 25]. This is an important feature to take into account in the context of purchasing in B2C environments [26, 27]. Thus, our proposal allows the offer of different prices for different levels of quality, and the negotiation of these prices. Besides, we also support additional payments while the session is still in progress. Thus, it is possible to make an initial payment for an initial period of time and then to make additional payments so as to continue with the session without any interruption.

Finally, as our proposal is independent of the payment protocol chosen, the level of security achieved is directly linked to the one offered y that protocol. Thus, if the payment method used (for example, an e-coin) does offer security against forgery or double-spending, our extension provide it because the security resides in the underlying payment method. Regarding the stream authentication, though it is not always required, if needed, it is possible to provide it following approaches such as the one described in [28, 29, 30].

2. Background

As mentioned in the introduction, our proposal aims to support the payment of multimedia sessions that are established by means of SIP. Thus, in this section we provide some descriptions about the underlying technologies: concretely, SIP [4], Session Description Protocol (SDP) which is used in conjunction with SIP [31], the offer/answer model [24, 25] and the *PRACK (Provisional Response Acknowledgement)* [23] method.

In this section we also describe some use case scenarios where payments in SIP are especially useful. These scenarios will be also used to establish the requirements that payments in SIP extension for micro and macro payment should satisfy in order to be generic, and to allow payment interoperability in an efficient manner.

2.1. SDP

The Session Description Protocol (SDP) [31] is used to describe the different parameters of a multimedia communication session. Namely, its purpose is to support the description of session announcement, session invitation, parameter negotiation and other forms of multimedia session initiation.

An SDP session description consists of a number of lines of text according to the following format: *<type>=<value>*. *<type>* is only one case-significant character and *<value>* is a string which contains an structured text whose format is determined by *<type>*. Thus, we specify a session-level description, i.e. we specify the different parameters that are applied to the whole session and all media streams. Optionally, we can include several media-level descriptions to specify details that are applied onto to a single media stream.

2.2. SIP

The Session Initiation Protocol (SIP) [4] is an application level signaling text-based protocol designed for the administration of multimedia sessions (e.g. a VoIP call, a distributed conference, an instant message conversation, etc.). The session initiation using SIP consist on an interchange of messages codified to in plain text by using a three way protocol. Concretely, the protocol works as follows: first a user agent sends an *INVITE* request with all the information that describes the type of session using SDP. This request could also include the media and ports to be used, codecs supported, etc.

When the callee receives the *INVITE* request it immediately answers with a provisional response (*(*100 Trying)) confirming that the request has been received. Then, the callee sends the provisional response (*200 OK*) to indicate the caller that the user agent is trying to alert the callee. When the call is answered, the callee user agent sends a *200 OK* response message to the caller to confirm that that the call is accepted. The caller sends an *ACK* request to confirm the reception of the previous message and, thus, to finish the three way handshake. Next, the media transmission starts using the set of codecs established during the session initiation. Once the session is initiated the peers can also exchange messages by using SIP instant messaging facilities. The session is established until one of the peers signals its termination by sending the *BYE* message. In SIP, the exchange of messages is usually made through UDP, but TCP or TLS could also be used when a higher degree of security is required.

The protocol that we have just described in a summarized way is described in the RFC 3261 [4]. This document also describes extensibility mechanisms to enhance SIP functionality. The mechanisms of extension included are: adding new headers for extending the functionality, new address schemes in order to support communication with new protocols and, the definition of new options tags so as to define new parameters for the headers.

2.3. Session negotiation based on the offer/answer Model

In this section we introduce the different mechanisms available to achieve an agreement on the attributes of a session. Firstly, we explain the basic offer/answer model and, next, we explain how to extend it with the *PRACK* method.

The offer/answer model defines a mechanism that allows two entities to agree on the attributes to use in a multimedia session. This process specified in the RFC 3264 [24] and some examples could be found in the RFC 4317 [25]. It uses SDP to describe the offers and a higher layer protocol like SIP is needed to carry out the offer/answer exchange. In these exchanges the content-type is established as *application/sdp*.

The *PRACK (Provisional Response Acknowledgement)* method is an extension to SIP allowing peers to refine the offer/answer model by offering additional possibilities for these exchanges. It is defined in the RFC 3262 [23]. This extension includes three messages: a *183 Session progress* provisional response, the provisional *ACK* or *PRACK* and its corresponding *200 OK* response to the *PRACK*. The server issues the first message indicating that a *PRACK-200 OK* cycle is about to start. Then, pairs of *PRACK* and *200 OK* messages allow clients and servers exchange offers and answers about the kind of media to be used in the session. When the negotiation is finished, the server issues a *200 OK* response code to the *INVITE* message.

2.4. Use case scenarios

SIP can be used for different kind of multimedia services [1, 2, 3]. In this section we present the different types of scenarios that our proposal tries to cover. These scenarios are similar to the ones described in [32, 10]. All of them use SIP to support the payment of multimedia services and gradually vary in complexity and functionality.

These scenarios will be also used to establish the requirements that payments in SIP extension for micro and macro payments should satisfy in order to be generic, and to allow payment interoperability in an efficient manner.

2.4.1. Scenario 1. SPAM in VoIP Systems

According to Kahn et al., [33] spam over Internet telephony and instant message is expected to be a problem more important than in spam in e-mail services. A solution consisting on requesting a small payment to first time users is depicted in [11, 12].

The amount to charge must be small and it must be refunded after including the user in a whitelist for future communications. Additionally, different payments methods could be supported and the amount to pay for each one can vary. Therefore, it is necessary to negotiate how to make the micropayment.

The refund can be made by using the same payment method or a different one, but in this case the caller and the recipient exchange their roles. In the refund the recipient makes the payment and the caller receives it.

2.4.2. Scenario 2. P2P market for real-time communications services

Let us suppose that an e-learning company offers its language-training services over a P2P multimedia service network. These services are based on the use of video, audio and/or instant messages and require a payment for its use. For the charging of the services, the company could use different models: a pay-per-time, at free model, pay-per-volume, session-based or pay-as-you-watch [34].

The model followed is specified in the description of the services provided. When the payment is made, the transmission start. Depending on the model followed the amounts to pay can vary from a few cents (micropayments) to some Euros/Dollars (macropayments). Additionally, users could be interested in negotiating the price to pay. They could even negotiate the quality of the audio and/or video used to provide the service.

2.4.3. Scenario 3. Voice messages

Let us suppose that a VoIP company provides a voice message service (as it could be the system proposed in [2]) where people can listen to the voice messages that they have received in their voice box. The access to this service can be paid incrementally so that the user can decide how much time listen and so that how much to pay. Thus, the company charge an initial amount for some initial time, lets say the first minute costs twenty-nine cents Euros/Dollars with PayStar [35] or if we use Huszti's proposal (H-P) [36], thirty-two cents Euros/Dollars (both are micropayment protocols).

When the time the user paid for is going to finish, the system notifies the user about it and that a new payment will be required in order to carry on listening. Then the user can decide to finish the call or he could make a new payment for some additional time. Again, the cost of each additional minute will be associated to the payment protocol chosen.

2.4.4. Scenario 4. Virtual conference

Let us suppose that a company wants to offer the access to a virtual conference with two consecutive presentations. The company requires a first payment of ten Euros/Dollars that gives the right to participate in the first presentation. Once the session is initiated, the price of the second one is only five Euros/Dollars.

In this scenario, we suppose that the customer supports several protocols such as Paystar and H-P (as micropayment protocols), Bitcoin [37, 38], Certified Bitcoins (cBitcoin) [39], Modified SET (MSET) [40] or PA-SET [41] (as macropayment protocols). However, the vendor only offers macropayment protocols such as Bitcoin, MSET or PA-SET due to the amount to be received. Thus, in this scenario, we finally suppose the customer chooses the PA-SET protocol because it is the only valid option for both.

The session is initiated with the exchange of payment options supported. The customer decides to use PA-SET. Then, some PA-SET messages are exchanged between the customer and the merchant. The customer also wants a receipt for this payment and requests it to the server. Once the payment is made, the merchant provides the receipt. Furthermore, the merchant will provide some loyalty information to the customer. This information will be used for the future accesses the customer wants to make in order to obtain better conditions or prices.

When the payment is made, the customer can obtain access to the first presentation. Later, when the session is about to finish (for example, there are only ten minutes left), the merchant sends a warning to the customer and requests a new payment. Then, if the customer decides to pay for the second presentation, he/she will send back the loyalty information that we previously received (to get a discount or to accumulate new points) and the payment protocol is executed. When the payment is made, the merchant provides the second presentation. He could also provide new loyalty information as well as a new receipt. Finally, the session finishes when the second presentation concludes.

2.4.5. Other scenarios

Similar to these scenarios where multimedia contents or services are accessed on a pay-per-use basis, we could consider other situations in which it is required or suitable the establishment of a (multimedia) session based on payment, e.g., accessing to whole courses from educational institutions [42] or copyrighted contents protected with Digital Rights Management (DRM) systems [43, 44], paying for parking services [45] or for accessing premium content on mobile television platforms [46], etc. We can also consider, at the beginning of the process, the use of coupons obtained through participatory-based systems such as in the Let's Meet mobile marketing framework [47] where a set of users can form a group in order to enjoy an offer (it could be a discount or a better service). Once the group is formed the coupon would be included at the beginning of the initiation process to enjoy the offer.

3. Requirements for SIP payments

In this section we describe the requirements that a SIP extension for micro and macro payment must satisfy. These requirements are extracted from the analysis of the use case scenarios described in the previous section and the study of other payment protocols such as SIMPA [9], LP-SIP [10] or SIP Enhanced SEMOPS Protocol [6]. We have also taken into account general considerations to facilitate the use of multiple payment protocols and payment interoperability (whether Web or not) described in some recent surveys [15, 16].

- *Association of payment information with multimedia session description.* The merchant should be able to describe the multimedia session with the associated payment information. Usually, session description is provided by means of SDP. Ideally, payment information should be incorporated with SDP.
- *Prices depending on the quality.* Along with the payment the merchant should be able to express different prices for streams of different quality. This is because, as explained in the previous section, a multimedia session can be composed of different streams with different quality.
- *Supporting different payment protocols.* Both merchant and customer should be able to use different payment protocols: both micropayment and macropayment protocols. They should be able even to support payments based on a third party (such as in Jennings et al.s proposal [12]) that receives the payment and generates receipts for accessing the services.
- *Different prices for different payment options.* A merchant should be able to support multiple payment service providers, multiple payment protocols and multiple prices for the different combinations of payment service providers and payment protocols.
- *Avoiding additional connections.* The negotiation process needed to select the payment protocol and method must be carried out within the same connection used to request the service.
- *Negotiation prices and payment options.* Both merchant and customer should be able to negotiate the price and the payment options to be used.
- *Receipts and loyalty information.* The merchant should be able to send the customer the payment receipts once the payment has been made. The merchant and the client should also be able to exchange loyalty information.
- *Additional payments.* The vendor should be able to process incremental or additional payments. These payments are made while the session is still in progress. Usually, customers make an initial payment that covers only an initial portion of the session and then, just before that

portion ends, the customer request an extension by paying an additional amount. This later payment should not interrupt the current transmission.

- *Time notification.* The merchant should be able to notify the customer that the time he paid for is finishing.
- *Different payment models.* The merchant should support different payment models: flat rate, volume, and session-based charging. It is also advisable to support others models such as the pay-as-you-watch model [34] in which the session is initiated without any payment. In this model, the payment is requested after some content has been sent. The models supported will be indicated together with the payment information.
- *Payments based on extensibility mechanisms.* The support of payments in SIP should be based on the extensibility mechanisms of SIP in order to facilitate the incorporation of payments to the current developments of SIP.

4. Extending SIP and SDP to support payments

The main goal of our proposal is to support the payment of multimedia sessions established through SIP. We have also tried to satisfy the requirements defined in the previous section. In this section we describe the SIP and SDP extensions we propose and how they can handle all the features and requirements previously mentioned.

Firstly, we describe the basic payment mechanism. It is in charge of carry out the payment, and provides the exchange of supported payment mechanisms from each peer. After that, we describe the rest of valued-added functionality, that is, price negotiation, the support of receipts and exchange of loyalty information, and the support of additional payments.

4.1. Basic payment

In general a payment consists of a set of steps. First, the vendor provides the payment options he supports. Then, the client chooses one payment option and the payment process starts. This process generally involves the exchange of payment messages between the client and the vendor. Most micropayment and e-cash protocols only require one or two messages [35, 36, 48, 49], depending on whether the protocol can generate e-coins of the exact amount to be paid or not. However, a macropayment protocol could require the exchange of more messages such as Bitcoin [37, 38] (Bitcoin can be used for both micropayment and macropayment), Certified Bitcoin [39], MSET [40] or PA-SET [41].

As described in Section 2.2, a SIP session is established between the peers by means of a three way handshake protocol. This exchange can allow us to map the payment directly with protocols based on micropayments, e-cash that can generate an e-coin of the exact amount or with payment systems based on a third party that issues a receipt of the payment as an access token.

To do that we have extended the SIP methods involved in the session invitation so that the *INVITE* message sent by the customer contains an SDP description in the body in which the client could send the payment options he supports. Additionally, with these options, the customer could send some vouchers, coupons or other loyalty information that owns in order to get a better price or service from the vendor. To support these features, we have extended SDP with an attribute that allows the association of payment information to the whole session or to a particular stream with a specific quality. Thus, in the body of the *200 (OK)* response code, the vendor would send an extended SDP that represents a payment request by indicating the supported and desired payment options and prices for a concrete service asked for by the customer. The set of payment options sent by the vendor is the subset of the customer's payment options that the vendor supports. Finally, in the *ACK* message the customer would send a receipt or the e-coin to make the payment requested.

The receipt or the e-coin as well as the payment information (payment options) contained in the extended SDP are sent in a structure called *PaymentInformation* (when we refer to this *PamentInformation* structure, actually, we are referring to an instance of an XML element, which is a *PaymentInformationType* as is described in Section 5.4). This information is sent in the body as a new type of content. The different uses of this structure are commented throughout this section and in more detail in the following section.

With the exchange we have just described (shown in Figure 1 in the left side), the payment can be made at the same time that the multimedia session is established without any additional message. Therefore, the basic payment functionality is added and there is not any message overhead.

In the event of the payment protocol chosen requiring the exchange of more messages, we would make use of the *MESSAGE* [4] method and *200 (OK)* response code that have been also extended to exchange the *PaymentInformation* structure in their body. Thus, after the first payment message sent by the customer in the *ACK* message, the vendor sends an answer in the *MESSAGE* method. As a response to *MESSAGE*, the customer sends the *200 (OK)* response code to confirm the reception of the message. This message, if needed, can also contain a payment message of the client. This pair of messages *MESSAGE-200 (OK)* can be used as many times as needed until completing the exchange of the messages of the payment protocol chosen. Additionally, when a third party, such as a payment gateway, has to take part in the payment we can use the same messages. Exchanging as many pairs of messages as required we avoid establishing an additional session with the third party.

The vendor, with his last payment message could optionally send some loyalty information or some voucher or coupon that allows the customer to obtain some benefit (such as a discount) in his/her next payment. The vendor could also send a receipt (see Section 4.3). Once the payment is made, the session starts and the vendor provides the contents/services paid for in the session. In the payment we explained above, if the client had also requested a receipt, he would receive it likewise in the message we have just described.

Furthermore, in some circumstances, it could be interesting that the different payment options supported by the vendor can be known in advance by customers. Thus, the number of payment options that client sends in the *INVITE* message is smaller. We have made this exchange possible by means of the extension of the standard *OPTIONS* message and its response code so that the *PaymentInformation* structure can be sent in the body. This is a simple, standard and efficient way to perform it.

4.2. Basic payment with negotiation

In the previous section, we suppose that the price is established by the vendor and cannot be negotiated. Thus, when the customer receives the payment options with the prices in the SDP description contained in the *200 (OK)* response code, the customer has two options: either making the payment or aborting the session.

However, there are vendors that want to support richer business scenarios and advanced features such as the negotiation of the price or the use of coupons obtained through participatory-based systems [47]. In fact, in B2C e-commerce trade is a fundamental step and valued by the end user [26, 15].

It might happen that the customer does not consider that the price requested by the vendor in the *200 (OK)* response code is adequate for the demanded service. It might also happen that the vendor could offer a discount to the client whether he has a voucher or he is able to prove that belongs to a particular user group that has recognized a specific discount [47]. Therefore, in these situations the customer wants to negotiate the price with the vendor.

The negotiation process consists of an exchange of offers/counter-offers between the client and the vendor. Thus, we have defined a new option tag called *negotiation* for the *INVITE* message so that the client can indicate he wants to negotiate.

The support of negotiation is not mandatory and the vendor might be unwilling to accept it. In this case, the vendor does not include, in the *200 (OK)* response code, the negotiation tag and it also indicates it in the extended SDP that contains the information related to payment in the *PaymentInformation* structure (this is explained in more detail in Sections 5.1 and 5.4).

We have provided the support of negotiation between the customer and the vendor by following the same exchange of messages as in the Reliability of Provisional Responses in SIP extension [23] (see also Section 2.3). Thus, we have decided to extend the three messages that compose this extension (*183 - Session in Progress*, *PRACK* and *200 (OK)* to the *PRACK*) in order to include the extended SDP with the *PaymentInformation* structure. In these messages, this structure contains the payment options supported as well as the price associated. Additionally, we have defined a new option-tag that can be included in the *Required* header by the client to inform the server about his interest in initiating the negotiation cycle. The indication that the customer wants to negotiate is also included in the *PaymentInformation* structure. If this indication was not included, it would indicate that this is his last offer.

The negotiation process (see Figure 1 right side) consists of a series of exchanges of *PRACK* and *200 (OK)* messages containing an SDP description with payment information for the whole session or for each stream representing offers and counter-offers. Furthermore, in this structure, both entities are able to indicate whether the offer/count-offer they made is their last offer. This mechanism is used to indicate that they are not willing to continue with the negotiation. Thus, the recipient should accept this offer or finishing the negotiation. Thus, the negotiation finishes when either the customer or the vendor answers with an empty message or when one of them decides to

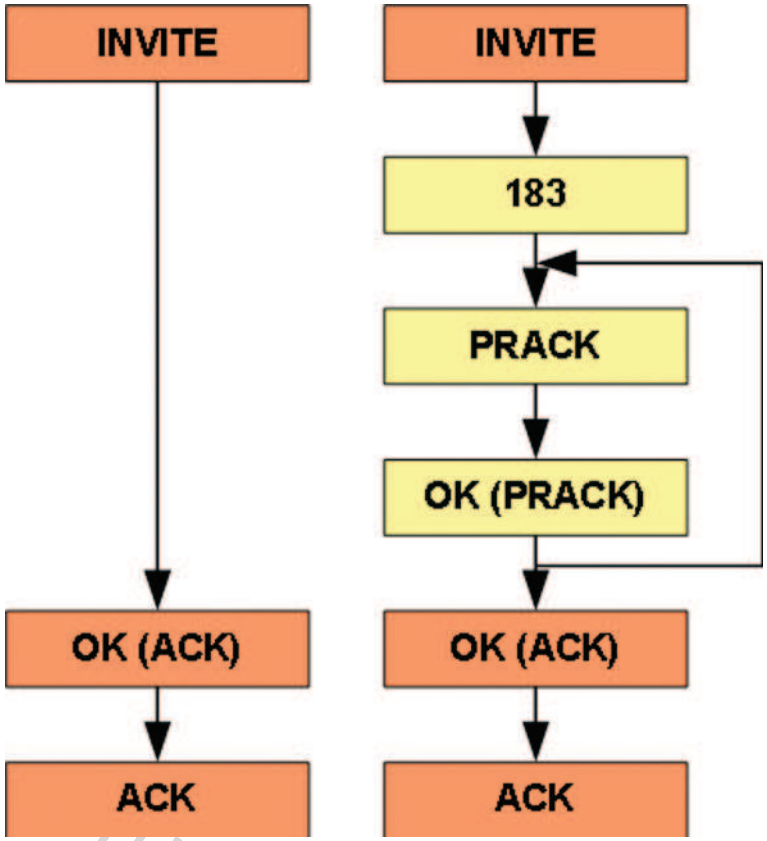


Figure 1: Session setup without (left) and with (right) negotiation

abort it. If the customer accepts the last vendor's offer, he must issue a new *PRACK* message empty. On the contrary, if the vendor accepts the last customer's offer, it issues the last *200 (OK)* response also including nothing.

When the negotiation has finished, the last stage of the already commented basic payment mechanism is performed. Thus, the *200 (OK)*, the *ACK* and the *MESSAGE* (if needed) messages are used to exchange the *PaymentInformation* structure containing the payment messages of the protocol agreed in the negotiation.

4.3. Receipts and Loyalty Information

After making a payment for a service, sometimes, customers want to receive a receipt to justify the payment made [15, 50]. It is important to point out that this receipt is not used for the same purpose as in the Jennings et al.'s proposal. In their proposal the receipt is used to provide access to the multimedia session and it can also be used as proof of payment. In our proposal, the receipt is provided only as a proof of payment.

The request of the receipt can be sent with the last payment message in the body with a *PaymentInformation* structure. We also make use of this kind of body for the *MESSAGE* method in order to support the issuing of receipts. Thus, apart from being used to send payment messages, it can also be used to deliver the receipt.

The information of the receipt is encapsulated in a structure called *PaymentReceipt* that is contained in the *PaymentInformation* structure. In that way, the vendor sends a message to the customer including the receipt after receiving a *PaymentInformation* indicating the customer's desire for having it. The vendor can also include, with the receipt, some additional information related to the transaction such as loyalty information, a voucher or a ticket/coupon with a discount, in order that the client obtains some benefits in the following purchases. The vouchers, tickets or coupons can also be obtained from other sources such as participatory-based systems such as the Let's Meet mobile marketing framework [47] where a set of users can form a group for taking advantage of an offer about a product or service. In this case, the system provides a coupon that would be sent in the *INVITE* message with the payment options that are supported. This information can be sent both during the process of payment without negotiation and in the process of payment with negotiation.

4.4. Additional payments

In Section 4.1, where we introduced the payment mechanism for SIP, it is supposed that the payment made by the customer is for the whole multimedia session. Depending on the kind of multimedia session (quality, duration, type of multimedia content, etc), the amount to pay could vary from a few cents to a lot of Euros/Dollars. This approach is useful for vendors because they obtain revenues regardless of whether the user receives all the content of the session or not. For example, after some time, the user can realize that he does not like the content/service or that, for any other reason he has to leave the session. On the other hand, customers usually prefer to pay only for what they obtain. In multimedia sessions this involves making a payment for a part of the duration of the session instead of making a payment for the whole session regardless of the time spent.

Our proposal is to divide the price of the session in small units of time or data that can be paid as the user is obtaining them. Thus, after each payment, an interval of time is guaranteed. In this situation, the customer has to be alerted when the time for what he has previously paid is about to finish. Upon receiving such an alert, the customer can decide whether he makes another payment or not. Obviously, not making another additional payment means the end of the session. This model in which the user can make payments along the session is what we have named *the support of additional payments*.

The support of additional payments in our proposal is made by using the *MESSAGE* method. In this case this message contains, in the body, an extended SDP description where we can include the information about the time/data/... left to finish the session. For this purpose we have defined an element called StopInfo that can be included inside the *PaymentInformation* element.

4.5. Security and privacy

In this section we analyse the security and privacy issues related to our proposal. The main goal of our proposal is to provide a framework that supports the exchange of the messages of different payment protocols in a generic way. Thus, the security of the proposal is based on two key elements: on the one hand, in the security of SIP protocol. On the other hand, on the security of the payment protocol chosen. The different risks and countermeasures associated to SIP can be found in [51, 52, 53, 54, 55]. We will not make a detailed analysis since it is out of the scope of this article. As for the security of the payment, it resides on the payment protocol chosen. The payment protocol supported should have some features:

- It must avoid the attacker from obtaining a payment as well as client (or vendor) information such as credit card accounts, e-coins, etc.
- Additionally, it must prevent client and vendor from performing double spending or generating bogus e-coins. Finally, the payment protocol must keep payment information hidden from attackers.

The security of the payment in our proposal resides on the security of the payment protocol chosen for an specific payment transaction. However, as our proposal is based on the SIP extensibility mechanisms it is compatible with the different SIP security mechanisms. These mechanisms could be used to enhance the security provided by a payment protocol. Thus, in our proposal HTTP digest could be used to provide message authentication and identity validity, IP Security (IPSec) can be used for protecting IP packets where SIP is transported (and by extension any mechanism that improves IPSec), SIP over TLS (SIPS) could be used to indicate that the access to resources should be secured by means of SSL/TLS which, additionally, allows us to guarantee authentication, confidentiality and integrity, S/MIME can be used in the event that the authentication, message integrity, non-repudiation and the encryption of message body is needed, SRTP and MIKEY can be used for protecting the media, etc. More details on SIP vulnerabilities and how to improve its security can be found in [51, 1, 3].

Another important issue to be taken into account is the privacy of SIP communications. Mainly, when a customer is using an anonymous payment protocol because he/she wants to protect his/her identity. To protect privacy in SIP we have to protect SIP user identities (information about customer and vendor that SIP reveals in headers), content of the body of SIP messages and hide IP addresses of communicating parties [56]. Furthermore, at the same time we protect user identities, we should support user authentication [56].

Among the different solutions that protect user identity and support user authentication in SIP we can point out PrivaSIP [57, 58] since these schemes guarantee user ID protection even the SIP messages are sent through untrusted domains, they do not require that SIP proxy servers have to maintain state information and they support standard SIP authentication mechanism [56]. Regarding the protection of the body part of SIP messages, we can use of some of the previous mentioned mechanisms such as S/MIME, TLS (DTLS), SIPS or IPSec. A discussion on the advantages and disadvantages on the use of these mechanisms can be found in [56]. For protecting IP addresses of the communicating parties there are several options such as SIPS, RFC 5767 [59], IPSec or Tor. A discussion on the use of these techniques can be found in [56]. By combining PrivaSIP with Tor we could obtain a solution that maximizes the level of privacy in SIP. Therefore, the main advantage of this combination would be a high level of privacy. However, its limitation is the latency that Tor would introduce in the communications and its applicability to current VoIP solutions since many SIP VoIP applications are using only UDP and Tor only supports TCP. More details and discussion can be found in [56, 60].

As a conclusion regarding security and privacy issues, we can mention that there are several solutions that can enhance the security and privacy of the SIP standard protocol and, therefore, the security and privacy of our SIP payment framework. Between the above-mentioned solutions we could point out (from weaker to deeper security and privacy): SIP (with or without authentication), PrivaSIP or PrivaSIP with Tor. The use of these options to protect security and/or privacy will depend on the payment scenario and the customer's and/or vendor's needs.

5. Detailed description of the extensions

In this section, we provide a detailed explanation of the different elements that we described in the previous section in order to extend SIP for the support of payments. Namely, we explain the SDP extension, the new option-tags defined, the new content defined for the body of a SIP message, the *PaymentInformation* structure that is used both in SDP and SIP messages to exchange payment information and, finally, the different extensions defined for the SIP messages can be found above.

5.1. SDP extension

The current specification of SDP is designed to characterize multimedia sessions and the streams composing them by means of several attributes. However, until now, none of these attributes can express information about payment. It is important to establish the association between the payment information and multimedia contents in order to facilitate discovering and comparison processes as recommended in DRM and e-commerce. This would also facilitate the exchange of this information with real time information in a more efficient way because all the information could be sent together. For this purpose, we propose to include payment information in the descriptions of multimedia content. This payment information is used to describe the payment options supported as well as the amount to pay with each payment option.

The information could be expressed at session level, or it could be expressed at stream level depending on whether the session is paid as a whole or whether it is possible to choose the different streams that compose the session. Furthermore, we have taken into account that each stream could be transmitted with different codecs and qualities. Thus, we should be able to specify different amounts for the different media formats. For this purpose we have defined a new SDP attribute called *payment-info*. This new attribute could be specified both at stream level and at session level. In this attribute, we do not specify a codec number if it is issued at session level.

```
a=payment-info:codec:schema:PaymentInformation
```

or

```
a=payment-info:codec:uri:PaymentInformation
```

The *codec* element represents the number of codec indicated with the m attribute. Thus, we are associating payment information to a stream. If this attribute is used at session level, there is no codec number.

The following element is a tag to indicate whether the payment information is attached to this attribute (tag *schema*) or whether the payment information is described in another document that is referenced by means of a URI (tag *uri*). This URI could point to an external document in the server or in any other server or it could point to a document that is attached to this SDP in the body of a SIP message.

In general, it is preferred that payment information is included with the a attribute (with *schema* tag) in order to avoid the establishment of an additional connection to obtain the description.

In both options to express payment information, the user obtains a *PaymentInformation* structure that describes the payment information we have just mentioned.

The *PaymentInformation* is an XML element defined according to the XML schema that we have defined for this extension. More details of this element are provided in Section 5.4.

Furthermore, in SDP, with the *PaymentInformation* element, we are able to express the different payment options without indicating the amount to pay. Thus, we provide flexibility to the vendor so that, depending on the client, he could offer different prices. For example, the user could belong to a specific group of users that has the right to some discounts. A scenario where this example is shown is in the Let's Meet! mobile marketing framework [47].

5.2. New option-tags

SIP supports the inclusion of different message headers to convey different kinds of information needed during the session. Some headers are composed by a list of option-tags indicating features supported, or required. Specifically, it uses the *Required* header to indicate the features that peers have to support mandatorily, the *Supported* header, to indicate that some additional features are supported, and, finally, the *Accept* header which is used to indicate the kind of information that can be supported. In this section, we describe the set of new option-tags that we have defined in order to support the different exchanges of payment information included in our proposal.

- *payment*. It is defined as mandatory for the *Supported* header and it indicates that the client supports our proposal. When a payment is required to access a multimedia session, if the customer does not specify this tag in the *INVITE* message, the vendor returns the *402 (payment required)* response code with a SDP description. This description contains payment information in the attribute *payment-info*, using the *PaymentInformation* structure requesting a payment.

- *additionalpayments*. Its use is optional. This new option-tag is included in the *Supported* header when the client wants to indicate that he supports making additional payments in every moment required along the session. With this tag, the client also wants to indicate the vendor that he desires to be informed when the time he has paid for is about to end. When the vendor includes this option-tag, he indicates that he could request additional payments before the period of time already paid finishes.

- *negotiation*. It is optional. This option-tag has been defined for the *Required* and *Supported* headers. This specifies that the client supports and/or wants to negotiate the price and the quality of the streams following an offer/answer model as proposed in [23, 24, 25] and by using the *PRACK* mechanism as proposed in Section 4.2. Specifically, in the *Required* header, he indicates he wants to negotiate. In the *Supported* header he specifies he might be interested in the negotiation whether the vendor supports it.

- *application/sippayment*. This option tag, which is defined for the *Accept* header, is used to signal the other peer that he can exchange payment information according to the extension we have defined. This tag also defines a content-type whose content is inserted in the body of a message. This content is used to receive the different payment options or to send payment protocol messages with the *PaymentInformation* structure.

5.3. application/sippayment content

We have defined a new content to be included in the body of a SIP message to convey payment information. This content is referenced with the content-type *application/sipppayment* as mentioned in the previous section. Its content is mainly used to exchange payment messages and it is defined from an XML structure named *PaymentInformation*. This structure is a generic XML structure composed of several elements that we describe in the following section.

5.4. PaymentInformation structure

In this section we describe the XML element defined to convey payment information such as payment options, receipts, requests of additional payment, etc. This element is named *PaymentInformation* and it represents an element of the *PaymentInformation*Type type which is defined in a XML schema since it provides a detailed description of data types, which can help with optimizations and filtering/managing data [61]. This schema has been created for this proposal. The instances of this element are included in the SDP and SIP messages.

Specifically, there are two possibilities to exchange this structure. First, in the messages that contain an extended SDP. This element is included with the *payment-info* attribute. This SDP is sent in the body of the messages and its content-type is *application/sdp*. Second, this structure, in some messages, is exchanged on its own, for example, for exchanging payment messages. In this case, this structure is sent in the body of the message and its content-type is *application/sippayment*.

The *PaymentInformation* element has been defined in a generic way and is extensible with the purpose of being able to use it in other payment scenarios such as the payment of Web services [14] or the payment of per-fee-links [62, 16]. This element is composed by several sub-elements that allow us to express different types of payment information. These elements are described next. The terminology used to describe them is the same as the introduced in Internet Open Trading Protocol (IOTP) [63, 64].

These structures are defined according to XML Schemas and, for the sake of simplicity in its description, we only describe the top-level structures. This structure is depicted in the Figure 2.

- *PaymentProtocols*. It includes the description of payment protocols supported by a peer.
- *Brands*. It enumerates the brands a peer can work with.
- *PaymentServiceProviders*. This enumerates the Payment Services Providers (PSPs) that a peer works with. This role could be played by a bank, a broker or even by the peer acting as vendor (for example, in micro-payments). The contact with a PSP could be established by either the client or the vendor, depending on the payment protocol. For example, in PA-SET, the client exchanges payment messages with the vendor and the vendor is the one who exchanges payment messages with the Payment Service Provider, in this case a payment gateway.
- *Prices*. An element that can contain different prices in different currencies.
- *Credentials*. This element is defined to convey credentials from the peers.
- *LoyaltyInformations*. This element allows the peers to exchange loyalty information.
- *PaymentDescriptions*. It represents a set of instances of the *Payment-Description* element, which is used to join the previous sets of elements in one single element. This latter element represents a specific payment description establishing the conditions for a particular transaction. Specifically, it represents the prices to pay using a set of protocols, brands and PSPs, and satisfying a set of conditions such as presenting a set of credentials or providing some loyalty information. Then, in a payment description, any tuple that can be formed with a combination of one element of the set of protocols, one element of the set of brands and one element of the set PSPs, represents the same price. If we want to express different conditions for the different combinations we should create different payment descriptions simplifying the number of elements of each set. Therefore, this element is used to compare prices and conditions. Furthermore, this element defines an attribute called negotiable that indicates whether the price is negotiable or not. Therefore, the *PaymentDescriptions* element introduces the different payment options available for a transaction.

- *PaymentProtocolMessage*. This element is defined to convey the messages of the protocol chosen to make the payment.
- *Receipts*. This structure is used to provide one or more receipts of the payment made to the client. This receipt is provided as long as the vendor supports it. This structure contains information about the payment made and the own receipt expressed according to the language supported by vendor.
- *Expiration*. This element is used to indicate the validity of a payment offer.
- *AdditionalInformation*. This element has been defined for extensibility purposes and can contain future elements that could be needed to express payment information. This can also be used to include some additional information that could be not strictly related to the payment, such as the time to stop the reproduction, as we can see in the following element.
- *StopInformation*. This structure contains information about the time when the user has to make a new payment before the session finishes. This structure is contained inside the *AdditionalInformation* element.
- *Signature*. This element represents an electronic signature of previous information. This signature agrees with the format proposed by W3C [65] in the XML Signature Syntax and Processing specification that is the standard for e-signature in XML. It is also important to point out that SIP could also use the security provided by S/MIME. This security is provided for the whole message, whereas this field is only used to secure payment information. The level of security chosen will depend on the transaction and the level of security required.

As we can see in the Figure 2, all the elements previously defined are optional. The inclusion of a particular element in this structure depends on the kind of message that is being exchanged: payment offer, payment message, receipt and so on.

5.5. SIP extension messages

In this section we specify in detail the way we have extended the different messages and what payment information is included in each message of the

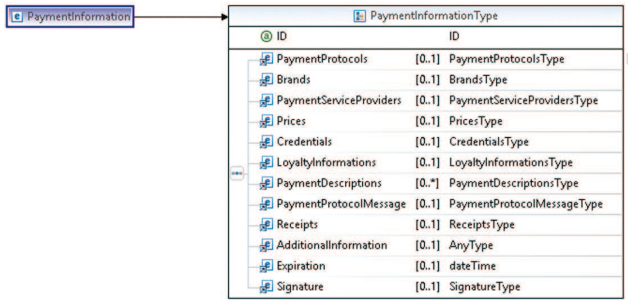


Figure 2: *PaymentInformation* structure

extension. The different messages that can be exchanged in the flow of an execution are depicted in the following figure. The messages are commented according to the order they appear in the Figure 3.

The two first messages are optional and are exchanged when the client wants to know in advance the payment options supported. Thus, the *OPTIONS* message (message 1) is sent to query the payment options supported. In the *Accept* header of this message the option tag *application/sippayment* must appear. Optionally, the client can send the payment options he supports in the body with the *PaymentInformation* structure with content-type *application/sippayment*. In this case, the elements involved in this structure are: *PaymentProtocols*, *Brands*, *PaymentServiceProviders* and *PaymentDescriptions*. In this case the *PaymentDescriptions* do not contain any price.

As a response to this query, the vendor issues a message (message 2) that contains a content-type with the tag *application/sippayment* and in the body an XML structure *PaymentInformation*. The set of elements involved in this structure are the same as the previous message. In this response, the vendor provides the information requested. In the event of the client providing, in the previous message, his information, then, the vendor provides only the options he supports from that set of options.

The following exchanges of messages (messages 3 to 10) show the process of session initiation making a simple negotiation and a payment by using a protocol that requires the exchange of several payment messages between the client and the vendor. This is the more general case. The different processes involved in this flow are described separately and in a more generic way in Section 4.

In order to initiate a session to access some multimedia contents that require payment, the client sends an *INVITE* message (message 3). In this

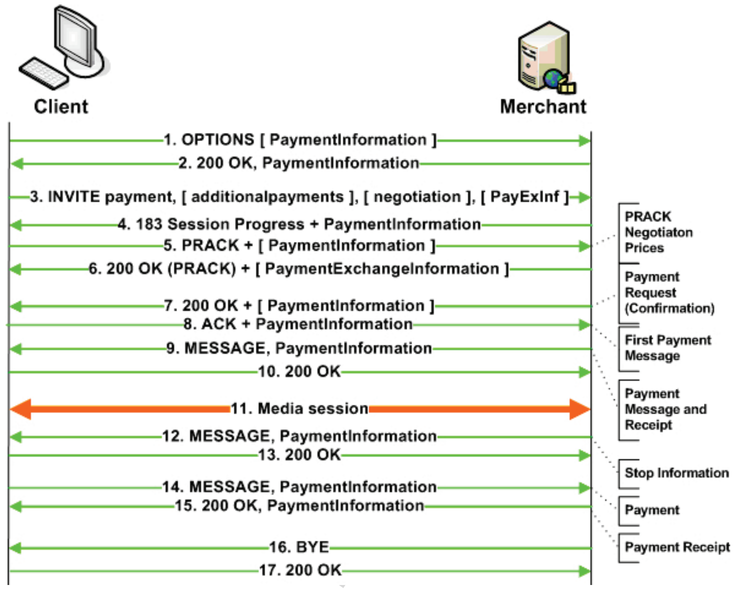


Figure 3: SIP extensions to support payments

message, he must include, in the *Supported* header, the tag payment to indicate that the client supports basic payments. Optionally, the client can include, in the *Supported* header, the tag *additionalpayment* to indicate that he supports payments along the session. Additionally, he can also specify the tag negotiation in the *Required* header in order to indicate that he supports and wants the negotiation of payment options and prices of the streams/session.

Instead of including the negotiation tag in the *Required* header we can put it in the *Supported* header. In this case, the semantic indicates that the client supports and might be interested in the negotiation.

In the *INVITE* message, the client could send the payment options he supports in the SDP with the *PaymentInformation* structure. The client can also send any loyalty information, tickets or coupons she has. The fields that are completed in this structure are the same as the explained ones in the message 1.

As a response, the vendor issues the *183 (Session Progress)* response code (message 4) whose body contains an SDP description. This description contains, for the whole stream or for each stream, the new attribute named *payment-info* including a payment request in the *PaymentInforma-*

tion structure (see Section 5.4). In this case the *PaymentDescription* field contains references to the prices for using each payment option. From this moment the price negotiation starts.

The client, in order to make a counter-offer to the offer received, issues the *PRACK* message (message 5). The offer is sent in the body as a SDP description with the extension defined to include payment information (*payment-info* attribute). The vendor can optionally send a counter-offer to the client's offer with the *200 (OK)* (answer to the *PRACK* message) response code (message 6). This counter-offer is contained in the body of the message as an extended SDP.

Otherwise, the vendor would send the message without a counter-offer. Messages 5 and 6 can be used by client and vendor as many times as they want until one of the parties decides to finish the negotiation. Whether the price is agreed, the vendor sends the message 7 to request the payment. Otherwise, the session finishes with messages 16 and 17.

The *200 (OK)* response code (message 7) is sent with the SDP description that contains the final information about making the payment for either the whole session or for each stream. This description is sent in the body of the message.

The following set of messages (messages from 8 to 10) shows the exchange of payment information needed to allow the access to the session (payment, credentials, loyalty information, etc) and the finishing of the initialization of the session.

The *ACK* message (message 8) is used to send that payment information. In order to convey that information in the body of this message, with the content-type *application/sippayment*, we make use of the *PaymentInformation* structure. In this structure, the client uses the field *PaymentProtocolMessage* to send a payment message or the *Receipts* field to send a receipt of the payment to other entity.

In general, the payment will be made to/through the vendor. Depending on the payment protocol, message 8 could be enough to make the payment. Then, the vendor would send the streams of the session (message 13). Furthermore, the *Receipts* field (with a *Receipt* structure and the attribute *requestreceipt*) can be included to indicate that the client wants a receipt for the payment transaction. If the receipt is requested and the vendor supports it, then, the messages 9 and 10 are sent as soon as possible. In any case, the session begins (message 11).

But it could happen that the payment protocol requires the exchange of more than a message to make the payment. In this case messages 9 and 10 are used to exchange the messages of the payment protocol chosen. These messages can be used as many times as needed to complete the payment protocol. The receipt could also be sent at the same time as the last payment message. In this general flow we are explaining, let us suppose the payment protocol to use needs two messages and that the client wants a receipt. These messages (9 and 10) can also be used, in any moment during the payment phase (message from 8 to 10), by both, the client and the vendor, to communicate with a third party involved in the protocol. In this latter case, there is no need of a session establishment since only payment messages are exchanged.

The vendor sends the *MESSAGE* method (message 9) to convey the second payment protocol message with the receipt for the user. This information is included in the body of the message using the *PaymentInformation* structure with content-type *application/sippayment*.

The *200 (OK)* response code (message 10) to the previous message does not need any extension. In case a new payment protocol message was needed, the message would include it, in the body, with the *application/sippayment* content-type and the *PaymentInformation* structure.

After the payment is made, the session starts (step 11). In this step the different streams are exchanged. We do not show any flow of messages because it depends on the protocol and media agreed.

This exchange of media information continues until either the session finishes (with messages 16 and 17) or the time paid for by the user has finished.

When the time is about to finish, if the user supports the mechanism defined as additional payments (see Section 4.4) the followings messages are exchanged. Otherwise, the session finishes with messages 16 and 17.

If the client indicates that he supports additional payment, when the time he paid for is about to finish, the vendor sends him a *MESSAGE* method indicating it (message 12). This information is sent in the *TimeToStop* structure contained in the

AdditionalInformation field of the *PaymentInformation* structure. This structure is sent in the body of this message with extended SDP that also contains the prices to be paid.

As a response, the client issues the *200 (OK)* response code (message 13), which does not need any extension.

If the client finally decides to continue with the session and make a new payment, he uses the *MESSAGE* method (message 14) to start the exchange of the messages of the payment protocol chosen. The payment information is sent in the body with the *PaymentInformation* structure. Optionally, he could also indicate he wants a receipt. Here, we suppose, as before, that the protocol is only composed of two messages. This payment could be for some part of the rest of the session or for the whole rest. It depends on the client's decision.

The vendor, as a response, issues the *200 (OK)* response code (message 15) with the payment protocol message contained in the body with the *PaymentInformation* structure. Additionally, the vendor could also provide the receipt and some loyalty information. If the payment protocol needed the exchange of more messages, the client and the vendor would use messages 14 and 15 as many times as they needed.

The session finishes at the end of the time or when the client is not willing to make more payments. For this purpose, we use the same messages as is specified in the standard SIP protocol. A *BYE* message and its response code (messages 16 and 17) are interchanged to finish the multimedia session. For our framework these messages have not been extended.

6. Related Work

At present, there are two kinds of approaches for charging SIP sessions based on payment: based on accounting by means of Authentication, Authorization and Accounting (AAA) infrastructures such as those based on RADIUS [66], Diameter [67] or SIPA/SIPA+ [68], or based on SIP-based payment mechanisms (SIMPA [9], LP-SIP [10] or SIP Enhanced SEMOPS Protocol [6]).

In AAA infrastructures, the way vendors usually charge for a session is based on the interaction between the vendor and payment provider (a client-server architecture) and these infrastructures provide an efficient way subscription-based service access type, which is useful for services based on subscription or for those services where there is a trust-relationship between the service provider and each one of its customers [5]. Therefore, if we suppose that any client could use the pay-per-use services of any service provider, this solution is not suitable for the scenarios such as those described in Section 2.4 where the services could be offered by a wide range of service providers (either with a trust-relationship with or not).

There are two additional reasons because these AAA infrastructures are not suitable for pay-per-use services: First, in AAA architectures, the credit-control server must to participate in each transaction [5], which is not suitable for scenarios where micropayments are required [48, 49, 10, 35, 36] (e.g. in the scenarios 1, 2 and 3 described in Section 2.4). Second, the credit-control client or the credit-contol server calculate the service unlike (micro)payment proposals where the service provider calculates it [5]. Therefore, this solution is not suitable for paying in and efficient and scalable way for real-time services (a more detailed analysis can be found in [5]) such as those described in Section 2.4.

The second approach is the proposal of specific payment protocols that extend SIP to make payments with it such as [69, 70, 9, 5, 10, 6]. The main problem of these solutions is that are limited to a particular kind of payment (micropayment or macropayment) and using a particular kind of payment model (credit or debit).

In order to solve the previous drawbacks and with the purpose of integrating payments in SIP in a more generic solution, Jennings et al. Jennings et al. proposed in [12] a solution that is based on integration with third party systems. This solution proposes to make the payment, instead of the vendor, to a third party acting as a Payment Service Provider (PSP) making a bank transfer from client's account to vendor's account. Once the payment is made, the PSP, as an acknowledgment of the payment, generates a payment receipt based on SAML [71]. This receipt is used, later, by the customer to access the vendor's service.

Jennings et al's proposal introduces several interesting features. Firstly, allows the client to choose his preferred brand (payment provider according to the terminology they use) and at the same time the vendor is independent of this choice and does not have to support any particular payment protocol. Secondly, the vendor receives, instead of a payment, a receipt integrated in the execution of SIP. Thirdly, the vendor can offer the clients different billing models such as flat rate, per unit time and per unit data as well as supporting multiple currencies and multiple payment providers. Finally, the client can extend a session by sending a new receipt for a payment.

In spite of the interesting features we have just mentioned, this proposal also presents several limitations. Although the participation of a third party (the payment provider) in order to carry out the payment process is interesting for maintaining the independence of the vendor as regards the payment systems, it means that the payment provider has to participate in every payment. This can be useful when the amounts to pay are medium or high (more than 5 Euro/Dollars) because this entity provides trust to the system. However, as mentioned in [48, 49, 10, 35, 36], the participation of a third party is not suitable for low payment transactions or in micropayments because this involves high transaction costs. The goal of micropayments is to support very low payment. Even some of them do not guarantee fairness because, as discussed in [48, 49, 10, 35, 36], in these schemes the loss is very low and the most important feature is efficiency. For this purpose, the microcoins should be verified with fast cryptographic operations and without the participation of a third party such as a PSP or a bank.

Other problem related to the participation of a third party is that it could know the details of the transaction (for example, identities of the parties, amounts transferred, frequency, etc), which may negatively impact the privacy of the customer. Indeed, this prevents that clients can use payment protocols that guarantee interesting features such as anonymity, the use of e-cash (as Bitcoin, Certified Bitcoins, Ripple, etc), the employment of smart cards [72] that provide an additional security level, etc. Furthermore, if the relationship between the client and the vendor is sporadic and the amount is low value, the transfer between banks could be more costly than using a specific payment protocol for this kind of transactions [10, 35, 36]. Thus, this solution is not flexible and cannot be adapted to different security requirements. Other important limitation of this proposal is the fact that it does not take into account that streams with different quality could have different prices to pay in a session. Finally, they do not allow the specification of different amounts to pay to the different payment service providers to use. We should take into account that each payment service provider could charge different commissions or fees to a vendor in order to provide his payment service.

The Internet Open Trading Protocol (IOTP) [63, 64] is a general framework that was designed to trade and make payments on the Internet. It is based on a payment-independent scheme and could encapsulate different payment systems. However, its use would involve making the payment in an outband way (that is, it would not integrated in the SIP flow) that would require the generation of a receipt. This ticket would be sent later in a similar way to the previous proposal. The same kind of deficiencies could be mentioned as for the Extended Payment Protocol (EPP) [62] or the new initiative launched by the W3C for a Web payment framework [15, 18, 16]. Even though, we can considered the main ideas regarding to interoperability and merchant competition when we have considered the requirements for a SIP framework.

A first approach to solve the problems we have just mentioned is the solution presented in [14]. This solution extends SIP and supports the payment with micropayment protocols. However, it does not support the use of macropayment protocols or protocols that require the exchange of more than two messages. It even does not define how to make the exchange of payment messages with other entities such as a payment gateway. The main reason for this limited support was that initially the solutions based on SIP did not consider macropayment protocols. However, during the last years and with the adoption and evolution of SIP technologies and the appearance of new scenarios (as mentioned in [5, 10, 6]) in the last years, as well as, the recent initiative launches by the W3C in the Web payments arena for promoting interoperability and vendor competence [6] had made us to consider the support of these new features. The solution we present extends and enhances this previous solution and offers a generic solution to cover both micro and macropayment protocols. We could also integrate the payment messages defined in other payment systems previously mentioned. Furthermore, our solution simplifies the extensions to make to SIP and shows payment information in a generic way that can be easily extended.

7. Comparison with previous work

In this section we compare our proposal with the only existing proposal to make payments in SIP in a generic way, that is, Jenning et al.'s proposal [12]. Their goal is also to support the payment under SIP.

As we previously commented in the Related Work section, Jenning et al.'s proposal is not suitable for micropayments as mentioned in the payment literature [48, 49, 10, 35, 36] because it includes the participation of a third party, which makes a bank transfer between the client and the vendor and increases the transaction costs as well as it may negatively impact the privacy of the client.

They do not support the use of different payment protocols either since, as we have just mentioned, it is based on the transfer of funds between entities. This could mean that both entities have to trust this party. Furthermore, this party knows the relationships between the clients and the vendors, and clients cannot use anonymous payment protocols.

Despite the drawbacks we have mentioned, we are going to compare that solution with our proposal regarding the number of messages to exchange with each protocol to make the payment and access the multimedia content/services. We have also supposed that Jennings et al.'s proposal would be modified to support the use of other payment protocols. This is the other approach we could have followed to support payment in a generic way. However, as we prove later, our approach is more efficient regarding the number of messages to exchange. In this comparison, first, we also analyse the basic payment mechanism, where there is only a payment at the beginning of the session, later a payment where both basic and additional payments are included. We do have taken into account the messages to access the multimedia content/services once the payment is made as well as the messages to exchange with the payment gateway because in both cases the number of messages would be the same.

We have based the comparison on the scenarios explained in Sections 2.4.3 and 2.4.4 since they represent the use of both a micropayment protocol such as PayStar and a macropayment protocol such as PA-SET [41]. We have also taken into account both basic payment mechanism and additional payments. With our proposal the messages exchanged for scenarios described in Section 2.4.3 and 2.4.4 are shown in Figures 4 and 5, respectively.

Finally, apart from the features we have commented that the previous proposal does not suppose, in case we had decided to extend it to support different payment protocols, the number of messages to exchange would be higher as we can see in the Table 1. In this basic mode, Jennings et al.'s proposal needs four messages more in PayStar and six for PA-SET protocol. In the advanced mode, that is, with additional payments during the session, in PayStar four messages more are needed, and four for PA-SET protocol.

Furthermore, we can also mention that our proposal could support the use of the receipts generated for the third party used in that proposal and could be sent in ours to make the payment. For micropayments, if we used that proposal as is, in the basic mode, that proposal would need nine messages, whereas our proposal would only need five. In the mode with additional payment that proposal would need thirteen messages, whereas we would need nine.

In this comparison, we do not compare the number of cryptographic operations used in these proposals since in both cases the number of these operations will be almost equal for the same protocol. The only difference would be that Jenning et al.'s proposal would need one more signature gen-

		PayStar	*PA-SET protocol*
Basic Payment	Jennings et al.	9	15
	Our proposal	5	9
With Additional Payments	Jennings et al.	13	19
	Our proposal	9	15

Table 1: Comparison based on the number of messages

eration and verification operation than our proposal due to the receipt that is generated once the payment is made to be sent during SIP session set up.

8. Conclusions and future work

Making payments in SIP could be interesting to support the purchase of the new multimedia contents that are emerging steadily. In this paper, we have seen how SIP can be extended, based on its extensibility mechanisms, to support the payment of multimedia sessions in a generic way. Concretely, we have extended both SDP and SIP to exchange payment information linked with multimedia content requests. These extensions have been made based on extensibility mechanisms provided by SDP and SIP, which facilitates the incorporation of our proposal to current implementation of SDP and SIP.

Our extension, which is specially designed to be used with micropayments, can also work with macropayment protocols as well as allowing any other payment systems, even the proprietary ones. Moreover, it supports the exchange of payment information defined by other previous proposal such as Jennings et al.'s proposal. Additionally, our proposal offers other interesting features such as the negotiation of both the quality of the streams and the prices of the payment options associated to the different streams, and the support of additional payment during the session to avoid resuming the session. In this negotiation we have also taken into account that the cost of the transaction could vary depending on the payment option chosen. For this purpose we have supported and extended the offer/answer model, SDP description as well as some SIP methods. The workload introduced by our extension is minimal and the participation of third parties is not required but supported. The aim of our extensions to SIP is to maintain the protocol as simple as possible.

Our future research will be centered on the study of more complex business scenarios as well as the extension of our proposal with additional pay-

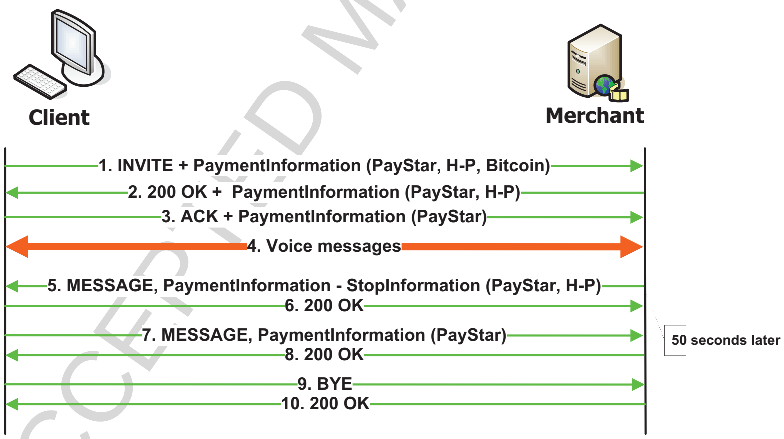


Figure 4: Payment for voice messages with micropayment protocol (PayStar)

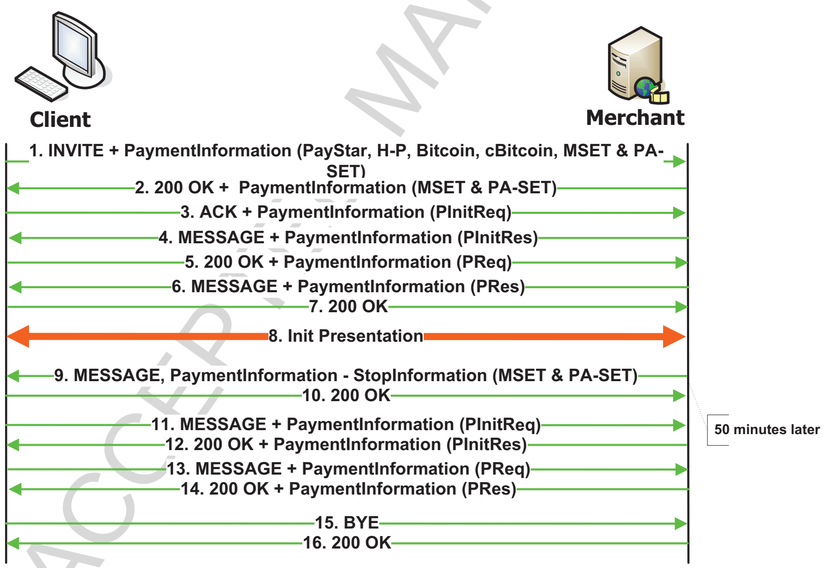


Figure 5: Payment for a virtual conference with macropayment PA-SET protocol

ment services.

9. Acknowledgments

We would like to thank the anonymous reviewers for their comments and suggestions. They have contributed to improve the quality of this paper.

This work has been partially funded by the following project: "TIN2014-52099-R Inteligencia Dinámica Emergente para Ciudades Inteligentes basada en el Internet de las Cosas" - Ministerio de Economía y Competitividad (Spain).

References

[1] M. Poikselka, G. Mayer, The IMS: IP Multimedia Concepts and Services, John Wiley & Sons, ISBN 978-1-118-69116-8, 2013.

[2] A. Ansari, M. Nehal, M. Qadeer, SIP-based Interactive Voice Response System using FreeSwitch EPBX, in: 2013 Tenth International Conference on Wireless and Optical Communications Networks (WOCN),

[3] A. B. Johnston, SIP: Understanding the Session Initiation Protocol, Artech House, ISBN 978-1-60783-996-5, 2014.

[4] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, SIP: Session Initiation Protocol, no. 3261 in Request For Comments (RFC), URL http://www.rfc-editor.org/rfc/rfc3261.txt, 2002.

[5] A. Ahmed, S. Khattab, K. Mostafa, S. El-Gamal, Comparison of Online Charging Mechanisms for SIP Services, International Journal of Electrical & Computer Sciences IJECS-IJENS 10 (2) (2010)

[6] A. Abdel-Hamid, O. Badawy, M. Aboud, SEMOPS+SIP+ECC: Enhanced secure mobile payments, in: Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2013,

[7] M. Sánchez, G. López, O. Cánovas, J. A. Sánchez, A. F. Gómez-Skarmeta, An access control system for multimedia content distribution,

in: Proceedings of the Third European conference on Public Key Infrastructure: theory and Practice, EuroPKI 2006, Springer-Verlag, ISBN 3-540-35151-5,

[8] W.-Z. Yang, L.-C. Yu, P.-C. Chen, T.-L. Chen, The Design of Multimedia Web-based Phone and Billing System with Freeware over the VoIP Network, in: Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing -Vol 1 (SUTC06) - Volume 01, IEEE Computer Society, ISBN 0-7695-2553-9-01,

[9] G. Zhang, F. Cheng, C. Meinel, SIMPA: A SIP-Based Mobile Payment Architecture, in: Proceedings of the Seventh IEEE/ACIS International Conference on Computer and Information Science (icis 2008) - Volume 00, IEEE Computer Society, ISBN 978-0-7695-3131-1,

[10] A. Ruiz-Martínez, C. I. Marín-López, A lightweight payment scheme for real-time services based on SIP, EURASIP Journal on Wireless Communications and Networking 2012 (1) (2012)

[11] J. Fischl, H. Tschofenig, Making SIP make cents, Queue 5 (2) (2007)

[12] C. Jennings, J. Fischl, H. Tschofenig, G. Jun, Payment for Services in Session Initiation Protocol (SIP), Internet-Draft, 2007.

[13] A. Aljohani, K. Al-Begain, Transaction-centric Mobile-Payment Classification Model, in: 2013 Seventh International Conference on Next Generation Mobile Apps, Services and Technologies (NGMAST),

[14] A. Ruiz-Martínez, J. A. Sánchez-Laguna, A. F. Gómez-Skarmeta, SIP extensions to support (micro)payments, in: The IEEE 21rd International Conference on. Advanced Information Networking and Applications (AINA07),

[15] J. Jaffe, S. Boyera, Now is the time for web payment standards, The Banker (2015)

[16] A. Ruiz-Martínez, Towards a web payment framework: State-of-the-art and challenges, Electronic Commerce Research and Applications 14 (5) (2015)

[17] A. Ruiz-Martínez, O. C. Reverte, A. F. Gómez-Skarmeta, Payment frameworks for the purchase of electronic products and services, Computer Standards & Interfaces 34 (1) (2012)

[18] W3C, *Web Payment Interest Group*, http://www.w3.org/Payments/IG/, 2015.

[19] W3C, *Web Payment Interest Group Charter*, http://www.w3.org/2014/04/payments/webpayments$_c$*harter.html*, 2014.

[20] H. Sinnreich, A. B. Johnston, Internet Communications Using SIP: Delivering VoIP and Multimedia Services with Session Initiation Protocol, Wiley, 2 edn., ISBN 0-471-77657-2, 2006.

[21] J. S. Erickson, Fair use, DRM, and trusted computing, Commun. ACM 46 (4) (2003)

[22] E. Jeges, K. Kernyi, E-Payment and DRM for Digital Content, 2nd INDICARE Workshop, 2005.

[23] J. Rosenberg, H. Schulzrinne, Reliability of Provisional Responses in the Session Initiation Protocol (SIP), no. 3262 in Request For Comments (RFC), URL `http://www.rfc-editor.org/rfc/rfc3262.txt`, 2002.

[24] J. Rosenberg, H. Schulzrinne, An Offer/Answer Model with the Session Description Protocol (SDP), no. 3264 in Request For Comments (RFC), URL `http://www.rfc-editor.org/rfc/rfc3264.txt`, 2002.

[25] A. Johnston, R. J. Sparks, Session Description Protocol (SDP) Offer/Answer Examples, Request For Comments (RFC), URL `https://tools.ietf.org/html/rfc4317`, 00011, 2005.

[26] D. Rosaci, G. M. L. Sarn, Multi-agent technology and ontologies to support personalization in B2C E-Commerce, Electronic Commerce Research and Applications 13 (1) (2014)

[27] P. Maes, R. H. Guttman, A. G. Moukas, Agents that buy and sell, Commun. ACM 42 (3) (1999)

[28] K. Mokhtarian, M. Hefeeda, Authentication of Scalable Video Streams With Low Communication Overhead, IEEE Transactions on Multimedia 12 (7) (2010)

[29] L. Song, J. Wang, L. Wang, J. Chen, A low-cost authenticating mechanism for interactive video streaming, IEEE, ISBN 978-1-4673-5000-6,

[30] Y. Jaafar, A. Samsudin, A. Syafalni, M. A. Omar, Framework for VoIP Authentication using Session ID based on Modified Vector Quantization, International Journal of Multimedia and Ubiquitous Engineering 8 (3) (2013)

[31] M. Handley, C. Perkins, V. Jacobson, SDP: Session Description Protocol, no. 4566 in Request For Comments (RFC), URL `https://tools.ietf.org/html/rfc4566`, 03350, 2006.

[32] J. Fischl, H. Tschofenig, Making SIP make cents, Queue 5 (2) (2007)

[33] S. Khan, M. Portmann, N. Bergmann, VoIP Spam Prevention, in: 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom),

[34] J. Domingo-Ferrer, A. Martínez-Ballesté, STREAMOBILE: Pay-per-View Video Streaming to Mobile Devices Over the Internet, in: Proceedings of the 13th International Workshop on Database and Expert Systems Applications, IEEE Computer Society, ISBN 0-7695-1668-8,

[35] S.-M. Yen, H.-C. Lin, Y.-C. Chen, J.-J. Hung, J.-M. Wu, PayStar: A denomination flexible micropayment scheme, Information Sciences 259 (2014)

[36] A. Huszti, Anonymous multi-vendor micropayment scheme based on bilinear maps, in: 2014 International Conference on Information Society (i-Society),

[37] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, 2008.

[38] S. Barber, X. Boyen, E. Shi, E. Uzun, Bitter to Better How to Make Bitcoin a Better Currency, in: A. D. Keromytis (Ed.), Financial Cryptography and Data Security, Lecture Notes in Computer Science, Springer Berlin Heidelberg, ISBN 978-3-642-32945-6,

[39] G. Ateniese, A. Faonio, B. Magri, B. d. Medeiros, Certified Bitcoins, in: I. Boureanu, P. Owesarski, S. Vaudenay (Eds.), Applied Cryptography and Network Security, Lecture Notes in Computer Science, Springer International Publishing, ISBN 978-3-319-07535-8,

[40] S. M. Shedid, M. El-Hennawy, M. Kouta, Modified SET Protocol for Mobile Payment: An Empirical Analysis, IJCSNS International Journal of Computer Science and Network Security 10 (7).

[41] A. Abdel-Hamid, O. Badawy, S. Bahaa, PA-SET: Privacy-aware SET protocol, in: 2012 22nd International Conference on Computer Theory and Applications (ICCTA),

[42] S. B. R. Kumar, S. A. Rabara, J. R. Martin, MPCS: a secure Mobile Payment Consortia System for higher educational institutions, in: Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human, ACM, ISBN 978-1-60558-710-3,

[43] D.-H. Shih, M.-H. Shih, D. C. Yen, C.-L. Chang, A robust copyright and ownership protection mechanism for music, Multimedia Tools and Applications (2014)

[44] Z. Zhang, Z. Wang, D. Niu, A novel approach to rights sharing-enabling digital rights management for mobile multimedia, Multimedia Tools and Applications (2014)

[45] J. Liu, R. Chen, Y. Chen, L. Pei, L. Chen, iParking: An Intelligent Indoor Location-Based Smartphone Parking Service, Sensors 12 (11) (2012)

[46] H. Weeds, TV Wars: Exclusive Content and Platform Competition in Pay TV, The Economic Journal (2015)

[47] L. Ntalkos, G. Kambourakis, D. Damopoulos, Lets Meet! A participatory-based discovery and rendezvous mobile marketing framework, Telematics and Informatics 32 (4) (2015)

[48] M. Lesk, Micropayments: An Idea Whose Time Has Passed Twice?, IEEE Security and Privacy 2 (1) (2004)

[49] I. Papaefstathiou, C. Manifavas, Evaluation of micropayment transaction costs, Journal of Electronic Commerce Research 5 (2), URL `http://www.csulb.edu/web/journals/jecr/issues/20042/Paper3.pdf`.

[50] I. Jacobs, M. Sporny, D. Ezell, Q. Sun, D. Jackson, Web Payments Use Cases 1.0, W3C Working Draft 30 July 2015, URL `http://www.w3.org/TR/web-payments-use-cases/`, 2015.

[51] D. Geneiatakis, T. Dagiuklas, G. Kambourakis, C. Lambrinoudakis, S. Gritzalis, K. Ehlert, D. Sisalem, Survey of security vulnerabilities in session initiation protocol, IEEE Communications Surveys Tutorials 8 (3) (2006)

[52] A. Behl, K. Behl, An Analysis of Security Implications in Session Initiation Protocol (SIP), in: Modelling Symposium (AMS), 2013 7th Asia,

[53] U. U. Rehman, A. G. Abbasi, Security analysis of VoIP architecture for identifying SIP vulnerabilities, IEEE, ISBN 978-1-4799-6089-7,

[54] A. Shrestha, Security of SIP-based infrastructure against malicious message attacks, in: 2014 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA),

[55] Z. Tsiatsikas, D. Geneiatakis, G. Kambourakis, A. D. Keromytis, An efficient and easily deployable method for dealing with DoS in SIP services, Computer Communications 57 (2015)

[56] G. Kambourakis, Anonymity and closely related terms in the cyberspace: An analysis by example, Journal of Information Security and Applications 19 (1) (2014)

[57] G. Karopoulos, G. Kambourakis, S. Gritzalis, PrivaSIP: Ad-hoc identity privacy in SIP, Computer Standards & Interfaces 33 (3) (2011)

[58] G. Karopoulos, G. Kambourakis, S. Gritzalis, E. Konstantinou, A framework for identity privacy in SIP, Journal of Network and Computer Applications 33 (1) (2010)

[59] T. Ohba, S. Schubert, M. Munakata, User-Agent-Driven Privacy Mechanism for SIP, no. 5767 in Request For Comments (RFC), URL `https://tools.ietf.org/html/rfc5767`, 5767, 2010.

[60] A. Ruiz-Martínez, A survey on solutions and main free tools for privacy enhancing Web communications, Journal of Network and Computer Applications 35 (5) (2012)

[61] J. Hill, H. Sutherlandy, P. Staudingery, T. Silveriaz, D. Schmidtx, J. Slabyz, N. Visnevskiy, OASIS: an architecture for dynamic instrumentation of enterprise distributed real-time and embedded systems, International Journal of Computer Systems Science & Engineering 26 (6).

[62] A. Ruiz-Martínez, O. Cánovas, A. F. Gómez-Skarmeta, Design and implementation of a generic per-fee-link framework, Internet Research 19 (3) (2009)

[63] D. Burdett, Internet Open Trading Protocol - IOTP. Version 1.0. RFC 2801., Request For Comments (RFC), URL `http://www.ietf.org/rfc/rfc2801.txt`, 2000.

[64] T. Dulai, S. Jask, K. Tarnay, IOTP and Payment Protocols, IGI Global,

[65] M. Bartel, J. Boyer, B. Fox, B. LaMacchia, E. Simon, XML Signature Syntax and Processing Version 2.0. W3C Working Group Note 23 July 2015, URL `http://www.w3.org/TR/xmldsig-core2/`, 2015.

[66] C. Rigney, S. Willens, A. Rubens, W. Simpson, Remote Authentication Dial In User Service (RADIUS), Request for Comments, IETF, published: RFC 2865 (Draft Standard), 2000.

[67] P. Calhoun, J. Loughney, E. Guttman, G. Zorn, J. Arkko, Diameter Base Protocol, Request for Comments, IETF, published: RFC 3588 (Proposed Standard), 2003.

[68] A. Tsakountakis, G. Kambourakis, S. Gritzalis, SIPA: generic and secure accounting for SIP, Security and Communication Networks 5 (9) (2012)

[69] J. Hao, J. Zou, Y. Dai, A Real-Time Payment Scheme for SIP Service Based on Hash Chain, in: E-Business Engineering 2008, IEEE International Conference on,

[70] G. Zhang, F. Cheng, C. Meinel, Towards Secure Mobile Payment Based on SIP, in: Proceedings of the 15th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ecbs 2008) - Volume 00, IEEE Computer Society, ISBN 978-0-7695-3141-0,

[71] S. Cantor, J. Kemp, R. Philpott, E. Maler, Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, 15 March 2005, 2005.

[72] Q. Wang, J. Zhu, Study on the Electronic Payment Technology in E-Commerce, in: Y. Yang, M. Ma (Eds.), Proceedings of the 2nd International