The 4th IEEE International Symposium on Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems
20-21 September, 2018, Lviv, Ukraine

Integrity and non-repudiation of VoIP streams with TPM2.0 over Wi-Fi networks

Kai-Oliver Detken[1], Marcel Jahnke[1], Malte Humann[1], Bernd Röllgen[2]
[1] DECOIT GmbH, Fahrenheitstr. 9, D-28359 Bremen, detken/jahnke/humann@decoit.de, www.decoit.de
[2] Global IP Telecommunications Ltd., Vor der Lehmkaut 6, D-35641 Schöffengrund, roellgen@globaliptel.com, www.globaliptel.com

The complete digitization of telecommunications allows new attack scenarios, which have not been possible with legacy phone technologies before. The reason is that physical access to legacy phone technologies was necessary. Regarding internet-based communication like voice over the internet protocol (VoIP), which can be established between random nodes, eavesdropping can happen everywhere and much easier. Additionally, injection of undesirable communication like SPAM or SPIT in digital networks is simpler, too. Encryption is not sufficient because it is also necessary to know which participants are talking to each other. For that reason, the research project INTEGER has been started with the main goals of providing secure authentication and integrity of a VoIP communication by using a digital signature. The basis of this approach is the Trusted Platform Module (TPM) of the Trusted Computing Group (TCG) which works as a hardware-based trusted anchor. The TPM will be used inside of wireless IP devices with VoIP softphones. The question is if it is possible to fulfill the main goals of the project in wireless scenarios with Wi-Fi technologies. That is what this contribution aims to clarify.

Keywords: VoIP, Trusted Computing, integrity, Wi-Fi, softphones, wireless streaming

I. INTRODUCTION

The latest successful example for the ever-ongoing convergence of information technologies is internet-based telephony, transporting *voice over the internet protocol (VoIP)*. The success of VoIP will not be limited to cable networks: Convergent speech and data transmission will affect next generation mobile networks as well. However, the new technology raises some security issues. In order to eavesdrop traditional, switched analogue or digital phone calls, an attacker has to physically access the transport medium. While he might have a harder time decoding the more complex protocols of IP networks and DSL-lines after dredging landlines of home-users, digital networks are generally more vulnerable to attacks, especially when used in conjunction with insecure wireless networks [1]. Efforts to add security features to VoIP products are currently infrequently deployed, though proposals for privacy protection exist. Protocols like SRTP can provide end-to-end security to phone calls, making them independent from the security of the transport medium and the communication provider. Secure VoIP protocols, using cryptographic protection of a call, would even be at an advantage compared to traditional telephony systems. While the problem of eavesdropping is solved for digital networks – at least in theory –, hardly any effort to add non-repudiation is made.

On the other hand, voice conversations provide inherent evidentiary value as they allow forensic evaluation and analysis of the contained biometric data, e.g. as an independent means of speaker identification. Methods for analyzing recorded voice communication are advanced and provide high probative force for example in court. In comparison to other digital media, e.g. text documents, specific features of voice communication can be viewed as contribution to security. The medium of communication here consists of a linear time-based full duplex channel enabling for inter- and trans-activity. In particular, inter-activity enables partners to make further enquiries in case of insufficient understanding. Furthermore, digital voice communication offers high reliability and quality of service, generally leading to improved understandability of VoIP communication in comparison to its analogue predecessors. To some extent, the mentioned properties mitigate problems to which digital documents are usually prone, e.g. misinterpretations due to misrepresentation, lack of uniqueness of presentation, and inadvertent or malicious hiding of content.

Therefore, starting from basic security aspects of VoIP communication, conversations will here be viewed on a transactional level between callers. The top-level category of protection targets considered is non-repudiation of conversations. Based on the diploma thesis by Christian Hett *"Security and Non-Repudiation for Voice-over-IP conversations"* [2] the research project INTEGER [3] has been started in July 2017. Three key features are addressed in this project:

- *Protection of the integrity of voice conversations*: Protecting a (recorded, digital) voice conversation from falsification and tampering differs from protecting the integrity of other digital data due to the relevance of the temporal context. In particular, packet ordering and loss have to be

considered properly, and a creation time must be assigned to each conversation.

- *Authentication of speakers*: Initial authentication of callers in conjunction with inherent biometric authenticity of voice is the basic approach to this problem. While it could be resolved in principle solely on the transport layer, it is advantageous to combine it with the methods of the first bullet point, to obtain the proof that a (recorded) conversation was carried out completely from the authenticated devices and not taken over by an attacker. It has to be noted that each authentication of a speaker requires trust in the devices used by the communicating parties.
- *Digital signatures over voice conversations*: Building on the first two tasks it is possible to achieve, for voice conversations, the level of non-repudiation provided by digital signatures over digital documents, e.g. an expression of will. For this, the aforementioned tasks must be complemented by a proof of possession of a trustworthy signature token and device, and the intention to sign.

The first task can be resolved in a stand-alone way, without any change in hardware devices or transport methods. The solution concept rests on cryptographic secrets created at the initiation of a call and their perpetuation throughout the call by a cryptographic chaining method. As a key issue, stability, quality of service, and necessary security mechanism to prevent attacks must be balanced with each other.

The main application of this concept is a secured archive for VoIP conversations which yields tamper-resilience and in consequence evidentiary value by far exceeding that of traditional tape archives. On the other hand, the second and third tasks need additions on the used devices ranging from the inclusion of authentication software and the roll-out of pertinent data, to the fulfillment of high security requirements for signature terminals.

The combined benefits of the technology developed here amount to a new paradigm for non-repudiation of digital data. The combination of integrity of recorded conversations, security about the identity of dialogue partners, and finally expressions of will embodied in signatures enables legally binding verbal contracts between unacquainted persons.

A trivial way of providing signed conversations would be achieved by recording the conversation and afterwards replaying the recording to the participants. However, that solution raises various problems as the recording as well as the playback have to be protected. The practicality is also doubtful, because it requires at least twice as much time. By contrast, the presented concepts allow using natural, fully interactive, traditional conversations to create legally binding verbal contracts.

German law already accepts verbal agreement even without additional witnesses. [2]

II. MAIN REQUIREMENTS

The central requirements for achieving non-repudiation by signing VoIP calls are, of course, related to security. Of the well-known trinity (confidentiality, integrity, and availability) of information security requirements, *integrity* is the central one to achieve non-repudiation for digital, packet-based, natural language communication. It needs to be assured that the conversation was not changed at any point in time, be it during transmission or later. Furthermore, integrity contains the integrity of any relevant meta-data created or used during a call, in particular call-time and the data that authenticates the communication partners, or at least the partner who applies the signature over the communication.

Due to the special features of voice communication, e.g. a bidirectional, full-duplex interactive conversation, only both channels together provide the necessary context to fully understand the content of the conversation and to make use of the inherent security that interweaved natural language conversations provide. To ensure that parts of the talk are not exchanged with other parts, replaced by injections, or cut out, the proposed system needs to ensure what we call *cohesion*. This means that the temporal sequencing of the communication and its direction is data for which the integrity needs to be protected in a way that makes later tampering practically unfeasible, e.g. by using sufficiently strong cryptographic methods. Cohesion as a feature related to time entails a subsidiary requirement, namely the secure *assignment of a temporal context* to a conversation. Each conversation has to be reliably associated with a certain time, which must be as close as possible to the conversation's start and the initiation of the signing. Drift of the time base should be mitigated during a signed conversation. Finally, cohesion also refers to qualitative aspects of the communication channel. A signatory is well advised not to sign a document which is illegible or ambiguous. In the digital domain this relates to the presentation problem for electronically signed data [4]. In analogy, the *quality of the VoIP channel* must be maintained to a level that ensures understandability to both partners during the time span in which the conversation is signed.

Further requirements regarding the *efficiency* of the system design and implementation are the following. First, it is highly desirable, both from a security-, as well as an efficiency viewpoint, to sign and secure the VoIP conversation as "close" as possible to its transmission time, and conceptually close to the actual VoIP stream, which is realized by using the payload of the original RTP-packets as part of the signed and archived data format. Simplicity of the implementation should minimize the effect on existing systems and infrastructures. Obviously, clients need to be modified, but the existing infrastructure should be left completely unchanged. This can be

achieved, as long as a way to transport additional signing data is available, which is the case for SIP/RTP. An efficient use of memory, bandwidth, storage space, and computational resources can be achieved by conceptual design decisions. Furthermore, *scalability* of the concept to a large number of concurrent calls is a necessity in real business environments. This means that centralized signature creation infrastructures must be avoided. Finally, any architecture that copes with VoIP needs to appropriately take *packet loss* and *quality of service* into account, in particular in view of the cohesion requirement. [2]

In security-sensitive application domains like telephony brokerage, calls need to be archived to ensure non-repudiation. The archive system should also allow the secure playback of the original conversation as well as the verification of the data involved, allowing its use as evidence. Considerations of long-term archiving aspects for signed digital data can be found in [5]. A special case that needs to be considered is the support of a business-to-consumer scenario where the consumer usually has no means to securely archive such data (this might of course also be the case in a business-to-business scenario).

The proposed solution shall support various mobile devices, e.g. smartphones, laptops, and tablets. Those devices should integrate a TPM 2.0 chip [6] allowing the use of an additional security layer. The system has to also work on those devices that do not provide a TPM chip, accepting the loss of additional security. Due to the nature of such mobile devices the (aforementioned) hardware requirements are further restricted by the potentially limited Wi-Fi bandwidth, which has to be considered.

III. Proposed Solution

Though the basic idea is based on the work of Christian Hett [2], some changes had to be made to support the TPM chip from the Trusted Computing Group [7]. Additionally, some minor adaptions regarding protocol, data structures, etc. were introduced in order to be better suited to the project's needs. Due to provider restrictions and complexity, the additional RTP application stream used by Christian Hett to exchange application data, such as signatures, is no longer an option and had to be worked around. The necessary changes allowed the optional addition of end-to-end encryption – already provided by one of the INTEGER project partners – making the archive proxy obsolete.

A. Protocol

The core concept ensures integrity, cohesion, and non-repudiation of a conversation by creating a signed hash chain covering all individual RTP-packets exchanged between the two communication parties. Fig. 1 shows such a bidirectional interactive conversation between the two parties *A* and *B*. *A* and *B* want to sign the conversation, because they want to make a legally binding contract via VoIP without requiring additional witnesses. The communication process involves the following steps as depicted in Fig. 1. The participants *A* and *B* collect all data packets of the channel in a buffer (1). Whenever a predefined chunk size is reached, *A* adds the chunk to the hash chain and generates a signature over the current hash sum (2). The signature is then sent to *B* (3a). Simultaneously, *A* forwards the chunk and the signature to the archive (3b). After receiving the signature from *A*, participant *B* checks the signature by emulating the chunk with the data packets he has sent (4). If the verification of participant *A*'s signature was successful, participant *B* adds his own chunk to the hash chain, signs the new hash value, and sends the signature to *A* implicitly confirming that the signature received from *A* was correct (5a). In parallel to this process, participant *B* forwards both chunks including their signatures to his archive (5b). Participant *A* in turn checks the signature of *B*. If the signature is correct, participant *A* sends the chunk and corresponding signature to his archive (6).

After this first "chunk-cycle" has been completed, the next cycle begins, which includes the next chunk pair. Again, both participants generate their own chunks and form the corresponding signatures. Because it is participant *A*'s turn again, he sends his signature to participant *B*. If the signature is correct, participant *B* checks it again and sends his own signature to participant *A*. This described procedure repeats itself continuously until *A* initiates the final handshake and *B* agrees to it.

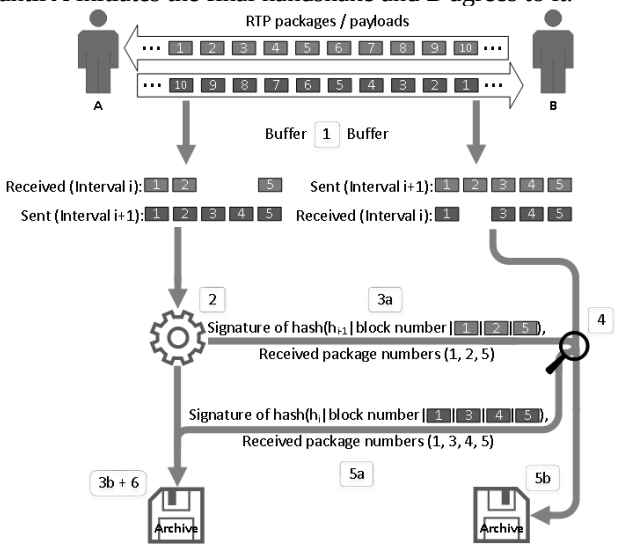

Figure 1. Flow diagram of communication between parties A and B

A timestamp authority secures the exact start of the call. In contrast to traditional long-term archives, this solution is based on streaming and securing ongoing conversations. It uses the timestamp to pinpoint the exact start time of a conversation and not the moment of archiving. Theoretically, the archive could be placed at the site of either of the parties or anywhere in between. It could be under the control of any party including third parties or installed centrally in a corporate environment. Although it seems useful to apply such a trusted timestamp at the end of the call to cover the whole conversation, it is more important to secure the start. In case the connection drops

for some reason or the session is aborted due to falling below the quality of service requirements there would be no trusted timestamp. If on the other hand the trusted timestamp was already placed at the beginning of the call, the integrity is secured until the connection was lost. As generally any (qualified) trusted timestamp entails some costs, the number of timestamps required by the protocol should be as low as possible; one in this case.

Besides the "intermediate" chunks described in the previous paragraph, dedicated start- and end-handshake-messages are part of the protocol.

The start handshake allows the negotiation of protocol-relevant parameters, e.g. the desired chunk size and quality of service requirements. Further, both parties exchange their certificates used in the signing process. The certificate of the party initiating the session has to be a qualified certificate [11], whereas the second party is allowed to use a self-signed certificate. The major reason to weaken the requirements of the second certificate is the business-to-consumer use case, as it is unlikely for a consumer to have a qualified certificate. However, it is encouraged to store a copy of that self-signed certificate at a trusted third party, e.g. the VoIP provider, to prevent someone else from pretending to be the user by applying an arbitrary self-signed certificate.

The end-handshake allows the protocol to properly finish a signing session. As mentioned before, the integrity of a lost connection is still secured, but without a proper end-handshake it is technically unclear whether both parties agreed to end the session yet, leaving the decision to an expert witness in case of a law suit.

The data included in both handshakes is also, at least partially, part of the hash chain. Additionally, there are special steps in both handshakes, allowing extensions to the actual protocol. One of those extensions is the TPM extension, described in Subsection C. The first hash value h_1 is based on the data exchanged during the start handshake:

$$h_1 = hash(meta|ext_{pre}|timestamp|ext_{post})$$

Where $hash$ is the hash function used, $meta$ the metadata exchanged during the handshake (chunk size m, quality of service thresholds, audio codec, sample rate, SIP IDs of the participants, certificates of both participants, and supported extensions), ext_{pre} the extension specific data that is secured by the trusted timestamp, $timestamp$ the trusted timestamp, and ext_{post} the extension specific data that is not secured by the trusted timestamp. The hash value for an intermediate chunk h_i includes the previous hash value, thus creating the chain, and the chunk related data: overall block number, bn, and RTP payloads RTP_j where j denotes the overall payload count.

$$h_i = hash(h_{i-1}|bn|RTP_{n+1}|RTP_{n+2}|\dots|RTP_{n+m})$$

The end handshake hash values, h_{sa} and h_{sb}, just include the extension data exchanged during the end handshake ext_{stop} and finalize the hash chain.

$$h_{sa} = hash(h_{i-1}|ext_{stop})$$
$$h_{sb} = hash(h_{sa})$$

Every time the hash chain is updated, the corresponding signature is sent to the communication partner. The actual hash value is not transmitted, only the information needed to calculate it, i.e. which packets were lost. The hash value would have to be re-calculated anyway to verify the integrity of the hash chain instead of just the signature of a, possibly arbitrary, signed hash value. Fig. 2 visualizes how the hash chain is created from the individual parts and how the trusted timestamp as well as the signatures of both parties are included.

B. Data Format

Both parties require the same data representation of RTP payloads to be able to update the hash chain and verify the signature of the other party. VoIP carrier gateways, however, usually are allowed to convert this data between various codecs, preventing the participants from verifying each other's signatures. As this problem also affects other protocols, such as ISDN, the Clearmode pseudo-codec was introduced [10]. The payload of RTP streams using this codec may not be altered by any intermediate parties transmitting the data. Additionally, the parallel RTP application stream used by Christian Hett [2] in 2006 to transmit various non-audio protocol information, such as signatures, is unfortunately not an option anymore, as those streams are not always supported by different VoIP providers or can get lost. Therefore, the proposed solution uses the *Clearmode* packets to transmit audio as well as additional protocol data in the same RTP stream. Depending on the audio codec that is used for voice data compression, a fix number of bytes are reserved for the actual audio data. The opus codec requires 60 bytes at a sample rate of 8kHz or 16kHz and 160 bytes at a

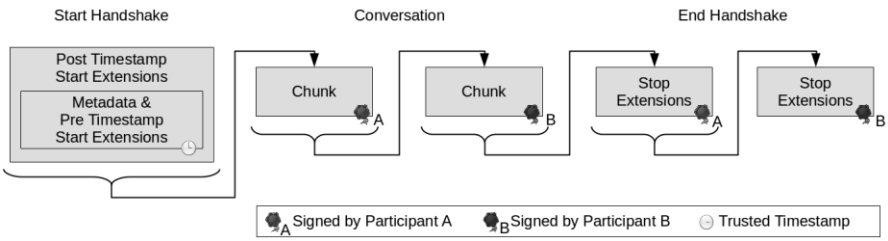

Figure 2. INTEGER hash chain

24kHz sample rate for compressed audio, for example. As the maximum transmission unit (MTU) is usually about 1 500 bytes, there is "some" room for the protocol data after taking the RTP-packet's requirements into account. For performance reasons the number of additional protocol bytes should still be kept as low as possible, reducing the bandwidth requirements. The protocol designed in the INTEGER project addresses this problem by specifying the necessary data fields on the bit level instead of the byte level, i.e. storing as much information as possible in a single byte.

In addition to the data exchanged as part of the conversation, the audio- and relevant protocol data needs be transmitted to the archive as well. As the archive data comprises the data received from the communication partner as well as the data sent to the partner, the up- and downstream bandwidth requirements for the conversation combined are roughly added on top of the conversation upstream to obtain the actual upstream bandwidth requirements. To reduce the size of the packets sent to the archive, especially in a wireless environment, the *Concise Binary Object Representation (CBOR)* [9] is used in the prototypical implementation, with the option to switch to a more sophisticated design if necessary. CBOR is based on a JSON data model and is encoded in binary. This saves bandwidth and allows for faster processing. One of the main goals for the development of CBOR was the Internet of Things (IoT), which includes very simple, inexpensive nodes. Therefore, CBOR is also very useful in wireless low-bandwidth environments.

C. TPM Integration

As non-repudiation of calls is only meaningful if the party who is interested in using a conversation as evidence can possess it as evidence, the signed conversation must be recorded. Special emphasis must be put on the format in which the calls are stored, i.e. the final outcome of the signing protocol. All intervals of a call are simply stored continuously by the archive software of the recording party.

In general, additional timestamps (as can be seen in the start chunk in Fig. 2) help pinpointing the exact start and duration of the call, thus providing additional security.

Regarding protection of secrets, especially in insecure environments like *Wi-Fi communications*, an additional security mechanism is necessary to obtain the desired security level. Therefore, a hardware trust anchor should be involved like the Trusted Platform Module (TPM). TPM chips are able to create, store, and handle keys for users in a secure way. A TPM chip can help determining that the used hardware or software has not been manipulated. For this purpose, the TPM collects the required information and stores it as hash values in the so-called Platform Configuration Registers (PCR). These values can be exported as a "TPM quote" that is signed by the TPM. The exported values can then be compared against a set of external reference values. The problem here lies in the initial acquisition of proper reference values. In the INTEGER project those reference values are collected as part of the registration process of the softphone application and, therefore, are stored by the softphone vendor who can later provide those reference values as evidence if necessary.

To ensure the state of the hardware and software used throughout the call, the TPM extension of the INTEGER protocol is used at three different steps in the protocol. During the first part of the start handshake, the initial value of the PCR used to store the value of the INTEGER hash chain and the certificate used by TPM to sign the corresponding TPM quotes are exchanged. This information is included in the data secured by the timestamp authority. In the second part of the start handshake, this data is added to the selected PCR (via the

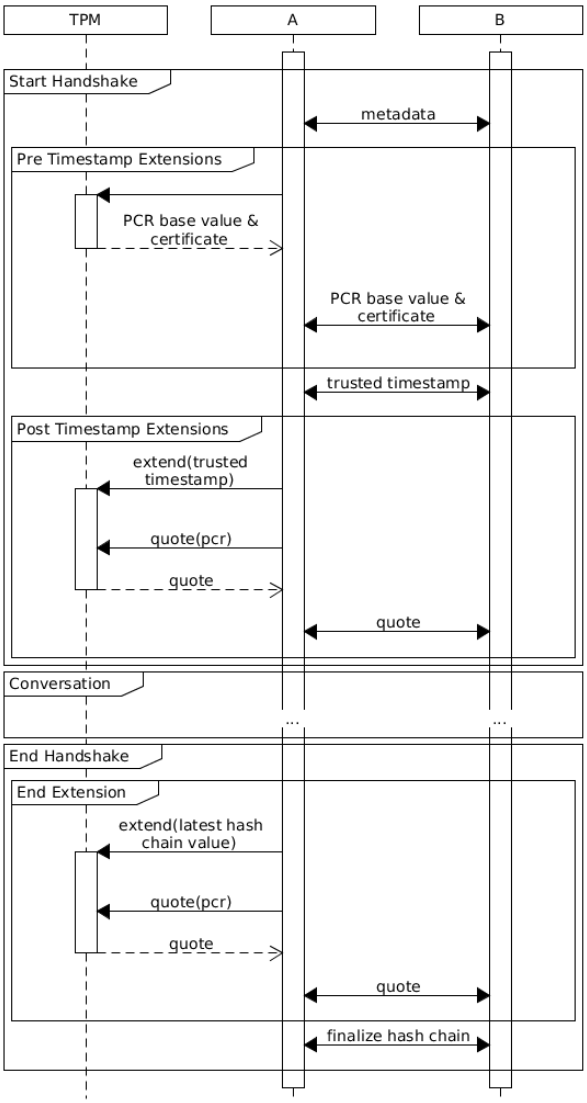

Figure 3. TPM integration via protocol extension

TPM extend command) and a TPM quote is generated by both parties. This TPM quotes are then exchanged (as part of the start handshake extension), verified and stored in the archive. Though the data added to the PCR is the same for

both participants, the initial value of the PCR might be different. Therefore, the initial values had to be exchanged to allow the other party to verify the first TPM quote. After a successful start handshake, the conversation continues as previously described by exchanging the current hash chain values and corresponding signatures. During this part of the protocol the PCR is not updated. It would be possible to add an "extension step" for each of these exchanges to the protocol, but this would – especially in case of the TPM operations – delay the exchange of said data. The second (and final) TPM quote is generated during the end handshake. First the latest value of the (conversation) hash chain (which is available at this point in the end handshake) is added to the PCR and then the TPM quote is generated. The quotes of both parties are exchanged again and stored in the archive. Fig. 3 shows these three TPM integration steps implemented as INTEGER protocol extensions.

The TPM extension of the protocol adds additional security to the conversation, but the TPM can also be used to secure the private key used to sign the conversation. It is possible to generate a new key pair with the TPM, where the private key will not leave the TPM. As this feature only affects the corresponding participant locally, this option can always be used if a TPM 2.0 chip is present, regardless if the other party supports it or not.

The INTEGER project uses the new TPM 2.0 specification of the TCG, which was published in 2016 and is used by market-leading vendor equipment, e.g. from Infineon. The TPM 2.0 has a CC EAL4+ certification, which includes secure algorithms like SHA256, SHA512 and elliptic curves (ECC). The following additional features are available in comparison to TPM 1.2:

- Definition of an interface that allows variability of underlying cryptographic algorithm
- Unification and expansion of authorization methods
- Dedicated BIOS support
- Simplified control model

The TCG defines schemes for establishing trust in a platform that are based on identifying its hardware and software components. The Trusted Platform Module (TPM) provides methods for collecting and reporting these identities. A TPM used in a computer system reports on the hardware and software in a way that allows determination of expected behavior and, from that expectation, establishment of trust. [6]

IV. CONCLUSION

One goal of the INTEGER research project is providing integrity and non-repudiation of internet-based multimedia communication for IP-based telephony. Especially in the business-to-business and business-to-consumer sectors, current solutions lack proof of confidentiality and reliability of communication. The INTEGER application is focused on the protection of integrity of a communication and secure authentication of the communication partners, by digital signing of communication. For this purpose, a hardware trust anchor – a TPM 2.0 chip of the Trusted Computing Group – is integrated into IP telephony via softphones. The project aims at establishing non-repudiation of verbal communication, which enables fundamentally more efficient business processes (among other things acting as proof of verbal contracts) thereby responding to market needs that have not yet been covered.

As future work there are multi-party conferences and proxy approaches within the archiving scenario conceivable. But first of all, INTEGER has to show that TPM deployment is useful in wireless softphone scenarios. If not, the prototype of INTEGER is still able to work without a TPM thereby slightly decreasing security features.

ACKNOWLEDGEMENT

The authors give thanks to the BMWi [8] for the financial support as well as all other partners involved in the research project INTEGER for their great collaboration. The project consists of the industrial partners DECOIT® GmbH, Global IP Telecommunications Ltd., reventix GmbH, and the research partner University of Applied Sciences of Bremen. A special appreciation goes to the associated partner Fraunhofer SIT for its support by pre-development of the described non-repudiation VoIP approach. The consortium used the patent description of Fraunhofer to bring the softphone basis of Global IP Telecommunications to a higher security level.

REFERENCES

[1] Rainer Baumann, Stephane Cavin, Stefan Schmid: *Voice Over IP - Security and SPIT*. September 2006

[2] Christian Hett: *Security and Non-Repudiation for Voice-over-IP conversations*. Diploma Thesis, Technical University of Darmstadt, Fraunhofer SIT, Darmstadt 2006

[3] Website of research project INTEGER: https://www.integer-project.de

[4] Peter Landrock, Torben Pedersen: *WYSIWYS? - What you see is what you sign?* Information Security Technical Report, 3 (1998) page 55–61

[5] Dimitrios Lekkas, Dimitris Gritzalis: *Cumulative notarization for long-term preservation of digital signatures*. Computers & Security, 23(5): 413-424, 2004

[6] Trusted Computing Group: *Trusted Platform Module Library Specification, Family "2.0"*. Level 00, Revision 01.38 - September 2016 Part 1: Architecture, 2016

[7] Website of the industrial consortium Trusted Computing Group: https://trustedcomputinggroup.org

[8] Federal Ministry for Economic Affairs and Energy: https://www.bmwi.de/Navigation/EN/Home/home.html

[9] C. Bormann, P. Hoffman: *Concise Binary Object Representation (CBOR).* RFC 7049, October 2013

[10] R. Kreuter: *RTP Payload Format for a 64 kbit/s Transparent Call.* RFC 4040, April 2005

[11] REGULATION (EU) No 910/2014 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL, 23 July 2014, Brussels 2014