Transaction-based authentication and key agreement protocol for inter-domain VoIP

Patrick Battistello[†a,b], Joaquin Garcia-Alfaro[b], Cyril Delétré[a]

[a]Orange Labs, 2 av. Pierre Marzin, 22307 Lannion Cedex, France
[b]Institut Télécom, Télécom Bretagne, 35576 Césson-Sévigné, France

[†]Patrick Battistello is no longer affiliated with Orange Labs nor Institut Télécom.

Abstract

We present an authentication and key agreement protocol to secure Voice over IP call establishment between interconnection proxies of different domains. The protocol operates on a transaction basis. Each transaction is defined as the set of operations and data required to send an authenticated message from a sender to a responder. A transaction allows a sender to either transmit a cryptographically protected stand-alone message; or a key-agreement message required to establish a secure session. The protocol handles transaction synchronisation loss and guarantees the use of a single transaction window in the general (inter-domain) context with multiple originating servers.

Key words: VoIP Security, Inter-domain VoIP, Authentication, Synchronisation.

1. Introduction

This paper presents a security protocol called DRCEP (Denial-of-service Resistant Call Establishment Protocol) that was designed in the VoIP (Voice over IP) context but which can be extended to other applications. Although some rationale of this section is specific to VoIP problematic, a large part of it applies to other applications as well.

The first aim of DRCEP is to minimise the DoS (Denial of Service) risk for servers, proxies or any service equipment facing the public Internet. As soon as a proxy is reachable from anywhere on Internet it may be subject to DoS or, even worse, Distributed DoS (DDoS) attacks which are difficult to mitigate [40, 26, 14]. This risk becomes higher when security protocols are used, because of their performance impact [47, 6, 49]. Since Internet has become the underlying layer for critical operations, security protocols are being adopted by an increasing number of applications (Web transactions, e-mail, VoIP). It should be noted that the (D)DoS risk on security protocols is more specific to inter-domain context because authenticating valid remote peers is resource consuming. As a consequence, the protocol design was focusing primarily on this scenario, considering that the (D)DoS risk in intra-domain is lower. The protocol can also be used in the intra-domain context, both in the access network and in the core network.

The DoS resistance of security protocols like TLS (Transport Layer Security) can be improved, but only by a relative small factor if public-key mechanisms are used. The best performances for authentication and key agreement are obtained with symmetric cryptography algorithms which require shared-secrets. Since installing shared-secrets between a large number of endpoints is not scalable, servers acting as TTP (Trusted Third Party) are required to enable meshed communications between any number of endpoints. In the proposed DRCEP protocol architecture, we assume that network domains exchanging large amounts of traffic are able to establish and maintain a shared-secret. On the other hand, domains having sporadic communications should go through a TTP to authenticate and obtain the required keying material. In this context, the TTP can be seen as a trust enabler proxy between parties having no previous relationship. We also assume that endpoints within each domain have a shared-secret with a responsible TTP inside the domain, at least for authentication.

These assumptions require the TTP to be on-line which may be seen as a major constraint. Nevertheless, it should be recognised that most Internet services today require on-line TTP(s). For example, in e-mail services, the message crafted by the sending endpoint is relayed through several servers prior to reaching its destination. Same applies to VoIP where proxies are necessary for routing and establishing call signalling. Considering OTT (Over The Top) services, on-line servers are required for endpoint authentication and data flow establishment between users. Even within P2P (Peer-to-Peer) networks, a responsible TTP has to set the initial configuration of endpoints (e.g., public parameters and certificates). Since new endpoints may join the P2P network at any-time this requires the TTP server to be on-line. Practical solutions are implemented by service or network operators to ensure the availability of TTP components (active

Email addresses: patrick.battistello@orange.fr (Patrick Battistello[†]), joaquin.garcia-alfaro@acm.org (Joaquin Garcia-Alfaro), cyril.deletre@orange-ftgroup.com (Cyril Delétré)

monitoring, mirroring, clustering) so that the server on-line requirement is not a real operational constraint.

1.1. Background

The reference protocol exchange for VoIP consists in a caller endpoint A (the originator) reaching a callee endpoint B (the responder) through one or several proxy servers Si. Once the call signalling channel has been established, media flows can be exchanged between endpoints. In most networks today, the SIP (Session Initiation Protocol) protocol [43] is being used for call establishment in association with SDP (Session Description Protocol) [4] for session description and RTP (Real-time Transport Protocol) [8] for conveying the media flows. Within the SIP protocol, the SIP-INVITE request plays a key role since it initiates the call establishment and may carry security parameters. Several studies have pointed out VoIP vulnerabilities, but the associated risks mainly depend on the underlying architecture [9, 2, 19, 25].

In the *intra-domain architecture*, VoIP communications remain confined in the same administrative domain, user endpoints are authenticated (usually with shared-secrets) and call signalling is routed through operator proxies. Because of these characteristics, if an endpoint inside the domain is compromised and launches DoS, SPIT (SPam over Ip Telephony) or vishing attacks [19] it can be detected and blocked by the network operator with mechanisms like [28, 31]. Further on, the vishing threat is merged with the SPIT threat because it combines social engineering and fraudulent calls.

The threats become higher in the *inter-domains architecture* because it combines the threats that may exist in each single domain and those resulting from the chosen interconnection mode. An exhaustive taxonomy is provided in [33] and exhibits several threats (DoS, call hijacking or mis-routing, message tampering) with risk levels depending on the interconnection mode.

A first mode, called *open model* hereafter, assumes that IP connectivity between proxies (or domains) and DNS lookups are sufficient to establish multimedia communications, just like in the e-mail architecture. The first issue is that, because of the PSTN predominance, current VoIP identifiers follow the E.164 [54] phone number standard and lack the domain part. Consequently, inter-domain calls are tricky to route and vice versa to verify. The second major issue is the accumulation of DoS and SPIT risks over the interconnection points which is called the *pinhole problem* in [45] and which is very similar to the risk found in e-mail architectures.

To solve these issues, a second mode called *closed model* or *private federations* hereafter consists in contractual agreements between a set of operators to establish a secure interconnection architecture. The IMS (IP Multimedia Subsystem) standards [1] define such an architecture with secure links based on IPSec (Internet Protocol Security) between domains and network topology hiding to protect interconnection proxies from DoS attacks. The phone number problem is solved by sharing securely (private) E.164 information between the operators.

1.2. Problem statement

It is clear that VoIP services are evolving from intra-domain architecture with PSTN (Public Switched Telephone Ntwork) interconnection to full IP inter-domains with end-to-end VoIP signalling. The private federations (or closed) model reproduces the PSTN principles, provides reliable routing, and is expected to reach a comparable security level. Its architecture implies that secure call signalling crosses all the intermediary domains and direct calls can be established only between adjacent VoIP domains. Furthermore, this model may have performance and cost impacts as explained in [52]. On the other hand, the open model provides a lot of flexibility but presents two major issues which have blocked its adoption (E.164 phone numbers routing and security risks on the interconnection proxies). Between these two models, a recent approach called VIPR (Verification Involving PSTN Reachability) [45] proposes an hybrid architecture combining VoIP, P2P and PSTN components. In brief, it relies on PSTN call information to build secure routing and authentication information which are then used to place direct VoIP calls.

Current solutions for open and hybrid models have some drawbacks and the private federations model may lack some service flexibility. Consequently, we identity the need for a call establishment process which addresses the inter-domains problematic and fits in the open interconnection model. Because VoIP calls (or application data) may be routed through the Internet, security is a key requirement: the protocol shall ensure authentication of involved party and key establishment. Other usual security requirements shall also be covered [11]: access control, privacy, anti-replay, DoS protection and session key freshness. This last requirement is taken in its broader scope meaning that a new session key shall be established for each call. On the other hand, the adversary is assumed to have maximal capabilities to interact with the protocol: as an external party (an outsider) or as a protocol participant (an insider) he may intercept, tamper, replay, forge or delete any protocol message.

Furthermore, the process shall manage VoIP E.164 phone identifiers routing and take into account VoIP specificities, especially its real time nature and the related regulatory constraints. As explained in [21], legal requirements may have strong impacts on VoIP deployment. In [18], the authors detail the impacts of the key disclosure legal requirement on secure call protocols.

1.3. Technical novelty

The DRCEP protocol limits the risk of (D)DoS attacks by inserting dynamic filtering values in each protocol message which can be checked straightaway by the responder. Although this principle is not new, it has been enhanced in such a way that the responder can cope with potential loss or disorder in the received messages, even when it is contacted concurrently by a large number of originator endpoints (or proxies). Authentication and session key establishment is achieved within a single message from the responder perspective and does not require session key transportation. This means that the specific

DRCEP protocol payload is short and it is compatible with an underlying UDP transport.

To reduce the performance impact of security operations (and thus the vulnerability to DoS attacks), the protocol uses only shared key cryptography. As opposed to protocols of the same class, DRCEP achieves the PFS (Perfect Forward Secrecy) security property both in case of passive off-line attacks (like key brute-force guessing) and active attacks (endpoint or controlled server revealing long-term secrets). Also, the protocol includes an efficient key renewing scheme that creates new encryption and MAC (Message Authentication Code) keys at each protocol run. Consequently, this increases the difficulty for an adversary to mount cryptanalysis attacks because it can not obtain several ciphertexts or MAC under the same key.

Considering the network architecture, the protocol may operate in a two-party setting or in a multi-party setting. The two-party setting assumes that the two principals (or entities) have a shared-secret. In the multi-party setting, intermediate servers (acting as TTP) are responsible for providing the required keying material to the originator. The intermediate servers may be organised in a hierarchical way to minimise the number of shared-secrets. Although involved in the achievement of each call, the intermediate servers do not need to remain in the application signalling path, which offers more flexibility than the traditional VoIP or IMS architecture [1]. Furthermore, the (D)Dos-resistance and PFS security properties are preserved for each intermediate server by applying the same principles as for the sender and responder entities.

Finally, the protocol operates on a transaction basis. Each transaction is defined as the set of operations and data required to send an Authenticated Message (AM) from a sender to a responder. A transaction allows a sender to either transmit a cryptographically protected stand-alone message; or a key-agreement message required to establish a secure session. In the first case, the DRCEP protocol interacts with the application and offers a fine-grained protection (on a per-message basis). This means that no underlying secure link (e.g., TLS or IPSec) is required to establish a secure call. In the second case, the DRCEP protocol is application agnostic and has the same purpose as a TLS handshake. In both cases it can be used to secure other applications than VoIP (e.g., email, Instant Messaging and web browsing).

Paper organisation — Section 2 presents and analyses the related work. Section 3 provides the DRCEP protocol specifications. Section 4 addresses the protocol security and the implementation of the core security functions. Section 5 measures the atomic performance of the protocol and establishes the simulation model for a complete network. Section 6 closes the paper with some conclusions.

2. Related work

2.1. VoIP secure call establishment protocols

2.1.1. Application independent mechanisms

The SIP (Session Initiation Protocol) VoIP protocol [43] specifies the use of TLS (Transport Layer Security) [16] for integrity and confidentiality protection of the signalling flows, in addition to peers authentication. It covers both the intra and inter-domain contexts; it can be used on a hop-by-hop basis along a chain of peers. Once the TLS transport channel is established, the SIP-INVITE request is sent securely from the sender to the next hop. Protection of the media flow requires the establishment of a session key between the caller and the callee. Following the SDES (Security DEScription) standard [4], the session key is generated by the caller and inserted in the SDP [20] part of the SIP-INVITE request. From this perspective, SDES is not a security protocol per se, since it requires another protocol to protect the SIP-INVITE request.

The same principles apply to DTLS (Datagram Transport Layer Security) [42] where UDP transport replaces TCP. Similarly, IPSec (Internet Protocol Security) [24] may be used to secure the VoIP signalling at the network level and thus enables the transport of a session key from the call originator to the responder. As explained in several papers [47, 6, 13, 52] there is a performance issue with these protocols when authentication is based on public key cryptography. This performance issue increases the operational costs and also the vulnerability to DoS (Denial of Service) attacks (Section 2.2 examines some solutions to reduce these risks).

Another vulnerability of these protocols is that the disclosure of a private key enables the recovery of past session keys. For example, if an adversary gets a TLS private key and has recorded some previous sessions, he can decrypt the PMS (Pre-Master Secret) values and obtains the corresponding session keys. Hence, the PFS (Perfect Forward Secrecy) property is not achieved.

2.1.2. Security protocols for VoIP signalling

The SIP standard [43] specifies the use of S/MIME [41] for end-to-end protection of the SIP payload. More precisely, the calling endpoint signs the SIP payload after inclusion of its certificate. Confidentiality may be partly supported (only the SIP payload is encrypted), provided the caller knows the callee certificate. To this purpose, a recent standard [23] defines a certificate management service intended to facilitate S/MIME deployment. The S/MIME standard has a negative impact on transport (because of the certificate inclusion) and it is subject to DoS attacks (because it uses public key algorithms).

A comparable approach is found in the SIP-Identity protocol [39] where the SIP-INVITE request is partially signed by the sending domain. As opposed to S/MIME, the signing is done at the domain level (by a trusted proxy), rather than at the endpoint level, and no certificate transport is required. Instead, the receiving domain recovers the certificate of the sender proxy through a (signed) URL in the SIP-INVITE request. This protocol offers no confidentiality protection for conveying a session key and, as for S/MIME, requires that network intermediaries do not alter the message content. It is also subject to DoS attacks because of signature processing and certificate recovering by the responder, as explained in [57].

The MIKEY (Multimedia Internet KEYing) protocol [5] is also designed to fit into VoIP call signalling and does not require

prior establishment of a secure link. Each protocol instance establishes a session key between the two parties. The security analysis made in [18] explains that the PKI (Public Key Infrastructure) and DH (Diffie-Hellman) modes of this protocol are vulnerable to DoS attacks if the receiver is flooded with fake security requests.

2.1.3. Mechanisms operating in the media plane

The SRTP (Secure Real-Time Transport Protocol) standard [8] ensures authentication of each media datagram in addition to its integrity and confidentiality protection. It requires a single master key from which integrity and confidentiality keys are derived. Establishing this master key between endpoints may be achieved at the VoIP signalling level with TLS/SDES, MIKEY or S/MIME protocols.

Alternatively, this may be achieved at the media plane level with ZRTP (Media Path Key Agreement for Unicast Secure RTP) [56] or DTLS-SRTP (Datagram Transport Layer Security Extension to Establish Keys for the Secure Real-time Transport Protocol) [27] protocols. These two protocols operate as follows: once the media connection information is extracted from the SIP SDP offer, one endpoint initiates a key establishment exchange with the remote peer. This exchange applies either PSK (Pre-Shared Key) or DH (Diffie-Hellman) methods. Several endpoints authentication mechanisms are available: digital signatures, PSK or authentication fingerprint passed through a secure signalling channel. A last option is supported with ZRTP which is a SAS (Short Authentication String) to be spoken and verified by each peer.

As explained in [18], deploying a PKI or shared-secrets at the end-user level raises respectively operational complexity and scalability issues. On the other hand, the ZRTP SAS method may be used only for sessions that involve humans at both ends of the communication. For these reasons, and because the SDP offer also has to be protected, a security protocol at the VoIP signalling level is required most of the time. Finally, [18] highlights the incompatibility between this end-to-end media key establishment model and some legal interception constraints.

2.1.4. Routing and verifying E.164 phone numbers

VoIP identifiers create a routability and verification issue because most of them are formatted according to the E.164 standard [17] which lacks the domain part. The ENUM (E.164 NUmber Mapping) protocol [17], build on the DNS principles, offers a theoretical solution but it has not been widely deployed. In fact, publishing end-user routing information would create a SPIT risk for the responder. Consequently, the routing of E.164 VoIP identifiers is mainly done on a hop-by-hop basis, relying on trusted interconnection operators which keep the routing information private.

Regarding the verification of the received E.164 caller identifier, a proposal was made in [54] to perform an RRC (Return Routability Check). The protocol is as follows: when the called domain receives a SIP-INVITE request, it extracts the E.164 calling number, generates a random token including call information and sends a verification request to the claimed calling number. The calling domain needs to intercept this request and respond with the extracted token, signed by the calling domain according to [39]. The verification phase relies itself on secure E.164 routing.

A recent proposal called VIPR (Verification Involving PSTN Reachability) [45] proposes to solve both the E.164 routing and verification issues. It is an hybrid approach combining VoIP, P2P and PSTN components which uses PSTN reliability to build secure routing information. It requires that each domain has a PSTN and a VoIP connection, joins a cross-operators P2P network and publishes in the DHT (Distributed Hash Table) the list of its PSTN phone identifiers and one of its VoIP proxy. Once a PSTN inter-domain call is completed, if the called number is found in the DHT, the calling domain contacts the called domain and obtains a *cryptographic call token* bounded to the specific called number and to the specific calling domain, along with the SIP routing information required to place direct VoIP calls in the future. While this mechanism offers an incremental approach, we foresee some limitations: the called endpoint can not authenticate the calling number, each domain has to store potentially a large number of tokens and besides all it requires PSTN endlessly. Actually, when the validity period of a token has expired or when a new destination is being called, a PSTN verification is required and a new token has to be validated (and stored). If the signature key of a domain is compromised, the PSTN verification shall be repeated for each token previously issued by that domain.

2.1.5. Protection against SPIT threat

A parallel was quickly drawn between SPIT and SPAM and several counter-measures are proposed in [44]: use of white lists or black lists, reputation techniques creating circles of trust, challenge of the caller with CAPTCHA or mathematical puzzles, payment at risk, call rating (and filtering) based on statistical analysis. To be efficient, most of these techniques require that the caller (or at least the calling domain) is authenticated. Whereas this is usually the case in intra-domain, caller authentication in the inter-domain context is a tricky issue, especially when the interconnection proxies are reachable from anywhere on the public Internet. Consequently, the approach we propose in this paper first aims at authenticating the caller (or originating domain) in an efficient way.

Nevertheless, even if the caller (or the calling domain) has been successfully authenticated, the first call creates an exception because the caller is not part of a white or black list. Several consent based approaches are discussed in [44] where the callee receives a call notification and later decides how to handle it. As an alternative approach, the authors of [37] assume that call participants usually establish cross-media relations (via e-mail, web or business card exchange) before placing a VoIP call. Weak secret information may be obtained to reach a specific contact and then inserted in the SIP-INVITE request as a (weak) authentication token. As for VIPR [45], this approach requires the storage of a large number of tokens.

2.2. DoS Protection

The exposure of secure call mechanisms based on public key cryptography to DoS (Denial of Service) risks may be reduced with one of the following approaches. The first one consists in increasing the processing performance of the host so that it can handle more cryptographic operations, as proposed in [51, 15]. Another solution is to choose the cryptographic parameters so as to reduce the computation cost for the responder. In the RSA based TLS handshake, this is achieved by setting a relatively small private exponent within the limit defined in [10].

A third approach is to delay the heavy computation tasks required by the responder by first challenging the originator. This principle is retained in several proposals [22, 3, 53] where the responder B, for each new connection attempt, first returns a cookie to the originator A. The cookie usually includes an ephemeral Diffie-Hellman (DH) value which requires one exponentiation for B. In [22] the authors claim that this value may be pre-computed but this does not work in case of (D)DoS attack because the responder would have to compute new values almost continuously. In [3] it is recommended that B keeps the same DH value as long as it is under a heavy load. However, even if B is not required to compute an exponentiation at each new connection request, it is easy for A to acknowledge the received cookie and then to force B to verify a wrong signature at the next step. The same analysis applies to [53] where B is protected at the very first step (because of the cookie principle) but needs to perform a public decryption on the second message received, whereas returning the cookie consumes almost no resource for the attacker A.

A fourth approach for the responder consists in caching previous (successful) connection state. This is described in the TLS standard [16] under the *session resumption* mode. This mode is useless in case of (D)DoS because the cache may rapidly be overloaded with attacker sessions replacing those of legitimate users. The counter-measure where the session state is stored by the client [48, 46] certainly preserves the server resources but it can not be enforced for attacking endpoints.

A more promising approach, called *client-aided RSA* is described in [12] for TLS. It consists in transferring a significant part of the computation load to the originator A. When it receives a connection request, the responder B returns a vector, along with its certificate, to A. Then A encrypts the PMS (Pre-Master Secret) with B public key and performs additional computations with the exponents received in the vector. Then B performs the remaining exponentiations which are less consuming than the complete PMS decryption. Furthermore, B may force A to pass a hash computation challenge before performing any exponentiation. Although this last measure does not guarantee that A is a legitimate user, it increases the computation load.

The above solutions, and others analysed in [50], improve the resistance of the responder to (D)DoS attacks but, even for [12], the gain is limited to a factor below 20. Even if these approaches are combined for greater efficiency, we anticipate that the (D)DoS resistance will remain much lower than with shared key cryptography.

2.3. Protocols using shared key cryptography

The application independent security protocols listed in Section 2.1.1 do all support a PSK (Pre-Shared Key) mode which increases the authentication and key establishment performances. The PFS security property is, however, not achieved: if the adversary obtains the shared secret, then all the previous session keys are compromised.

The same statement holds for the various authentication and key establishment protocols analysed in [11], whatever the number of parties involved and the method used for key establishment (agreement versus transportation). Furthermore, the responder usually needs to perform (at least) one cryptographic operation for checking the validity of the received message. Recently, a new MIKEY mode denoted KMS was proposed in [29]. It is a ticket-based approach with a trusted third party, inspired from Kerberos [32], which can also be integrated into the VoIP call establishment. Since the KMS protocol performs key transportation, it does not support the PFS property, it has a negative impact on transport and forces the responder to maintain a connection state before it can check the validity of the received message.

An alternative approach is described in IPACF (Identity-Based Privacy-Protected Access Control Filter) [55], where each single frame conveys an access filter value requiring only a comparison operation for the responder. This filter value is updated at each new frame and it depends on a shared secret key established between the server and each client. When the server receives a frame with a valid filter value, it responds to the client with a new responder filter value and updates the client filter value for the next frame; the same process holds for the client. If the received filter value does not match the expected one, the message is discarded without requiring any cryptographic processing. This mechanism also provides user privacy by sending a pseudo-ID which is user specific and changes at each frame. It is, however, unclear how the protocol behaves if some messages are lost or disordered, and whether strictly bidirectional exchange has to be maintained between the server and each client.

The DRCEP protocol described in Section 3 retains the IPACF principle of dynamic filtering values while supporting message loss or disorder, as well as unidirectional flow only. It also supports the PFS property and optionally a hierarchical architecture to simplify the management of shared secrets.

3. Protocol specifications

3.1. General overview

The protocol runs between entities A and B which can be the users endpoints themselves or proxies acting on behalf of them. For performance consideration, and DoS protection, the protocol uses only symmetric cryptography, which requires A and B to have a shared-secret. Since installing shared-secrets between each pair of entities is not scalable, an intermediary server S may be involved. Entity S has one shared-secret with A and one with B; it is responsible for authenticating those entities and for providing to A the material required to contact B.

$\|$ or ,	:	Concatenation operator.
$\oplus$	:	XOR operator, equivalent to binary modulo addition.
$\{\mathrm{V}\}_{\mathrm{K}}$	:	Encryption of V with symmetric key K.
δt	:	Time precision period used for time related operations.
AM	:	Authenticated Message sent from A to B .
$\mathrm{ATI_E}$	:	One of the Acceptable Transaction Index for entity E.
$\mathrm{BTI_{E,p}}$	:	p^{th} secret Base Transaction Index of entity E. The initial value is noted $\mathrm{BTI_{E,0}}$.
$\mathrm{CK(TI_E, X)}$	:	Confidentiality Key used for encryption operations.
$\mathrm{C_X}$ or $\mathrm{C'_X}$	:	Public constant.
$\mathrm{FK(TI_E, X)}$	:	Filtering Key matching transaction index $\mathrm{TI_E}$ of entity E and issued by entity X.
$\mathrm{FV(TI_E, X)}$	:	Filtering Value matching transaction index $\mathrm{TI_E}$ of entity E and issued by entity X.
$\mathrm{H(V)}$	:	The result of an one-way hash function applied to V.
$\mathrm{IK(TI_E, X)}$	:	Integrity Key used for MAC operations.
$\mathrm{K_{AB}}$	:	Secret master key shared between entities A and B.
$\mathrm{len(V)}$	:	Binary length of value V.
A mod B	:	The remainder of the integer division of A by B.
$\mathrm{MAC_K(V)}$	:	Message Authentication Code applied to V with key K. When $\mathrm{V} = \Sigma$, the MAC applies to all the fields contained in the message.
OM	:	Original MESsage to transmit from A to B.
$\mathrm{OTPR_E}$	:	One-Time Pad Result targeted to entity E.
$\mathrm{SK(TI_E, X)}$	:	Session Key matching transaction index $\mathrm{TI_E}$ of entity E and issued by entity X.
$\mathrm{TI_E}$ or $\mathrm{TI_E}(t)$	:	Secret (current) Transaction Index of entity E.
$\mathrm{TM_E}$	:	Transaction Material intended to entity E.
$\mathrm{TRID(TI_E)}$	:	TRansaction IDentifier corresponding to transaction index $\mathrm{TI_E}$ of entity E.

Table 1: Notations used in this paper

Consequently, S needs to be on-line and may be duplicated for redundancy purpose. This is not seen as a real constraint, since most communication protocols today require an on-line TTP (Trusted Third Party) — at least for phone number routing or for periodical verification. In Section 3.6, we explain how this architecture can be extended with several intermediary servers Si to meet the classical hop-by-hop inter-domain model or to reduce the number of shared secrets maintained by each Si. Although not explicitly described further on, the protocol applies to the intra-domain context as well or to a mixed scenario where S may be in a third domain.

The protocol operates on a *transaction* basis and several transactions can be triggered concurrently by the originator with support of transaction loss or disordering. Disordering means that a first transaction message is received by the responder after a second message triggered later on (this may result from an underlying UDP transport). A transaction is defined as the set of DRCEP operations and messages required to send an Authenticated Message (AM) from A to B. The AM message includes specific protocol information and the Original Message (OM) that A wants to deliver to B. The AM message offers integrity protection and OM may be encrypted for confidentiality purpose. When the protocol runs directly between A and B, a single AM message is required to perform entities authentication, key establishment and to deliver the original OM message. When n intermediary Si servers are involved, this requires $2 \times \mathrm{n}$ additional messages which corresponds to n sub-transaction legs.

This architectural setting is similar to that of Kerberos, but the protocol is different and it brings additional properties: PFS is achieved even if an entity or a long-term secret is compromised. The protocol also offers DoS resistance and the size of the final message AM is close to that of the Original Message (OM). This last property is obtained because DRCEP requires no key transportation in the AM message (as opposed to Kerberos). Finally, the encryption and MAC keys are renewed at each transaction; this increases the security of the overall protocol.

The DoS resistant property comes from the exclusive use of symmetric cryptographic algorithms and from the insertion of a dynamic Filtering Value (FV) in each transaction message. This filtering value is derived from the TRansaction IDentifier (TRID) that enables the responder to infer the corresponding Transaction Index (TI). In this respect, the TRID value corresponds to the public image of the Transaction Index (TI) that shall be kept secret. The responder entity (B or any intermediary server Si) can check straight away the received FV value by comparing it to the pre-computed (expected) one. In the general case where the responder receives concurrently protocol messages from a large number of originators, this filtering value also serves as an authenticator of each single originator. As opposed to several protocols (SIP Identity [39], Kerberos [32], MIKEY [5]), the responder is not engaged in any resource consuming operation (cryptography, context handling or message

generation) before a valid FV value is detected. This ensures to the responder maximum protection against DoS attacks. Additionally, because the FV value changes at each transaction in an unpredictable way, it serves also as a protection against blind or replay attacks.

From this perspective, DRCEP protocol follows the same filtering principle as the IPACF protocol [55] by using a dynamic value which is updated at each message. Since the FV value is derived from the TRansaction IDentifier (TRID), it also enables the responder to detect which Transaction Index (TI) has been issued by the originator. The set of Acceptable Transaction Index (ATI) maintained by the responder at a given time is called a *transaction window*. The transaction window accommodates for possible transactions loss or disordering because, as opposed to IPACF, there is not a single transaction index (and thus a single FV value) acceptable by the responder at a given time. More precisely, the Transaction Index (TI) is part of the shared-secret and it is iterated independently by both entities at each transaction. It serves as a pre-image to a secure one-way hash function to form the TRID value that is the public image of TI.

In the general case where the responder B receives AM transaction messages concurrently from several distinct entities Ai, it can not maintain a separate transaction index (and the corresponding transaction window) with each Ai. Therefore, the above principles are adapted so that B only has to maintain its own $\mathrm{TI_B}$ value and the corresponding transaction window for all the possible Ai entities. The $\mathrm{TI_B}$ secret is shared with all Ai entities and iterated independently by each Ai through a secure one-way function to achieve the PFS (Perfect Forward Secrecy) property.

From the shared-secret $\mathrm{TI_B}$ and the master key $\mathrm{K_{AiB}}$ shared between Ai and B, each Ai entity derives the corresponding transaction material, among which the $\mathrm{FV(TI_B, Ai)}$ filtering value and the session key. The key derivation function shall satisfy the PFS property, meaning that if a derived key is compromised, the adversary cannot infer the master key or any other derived key. Furthermore, to ensure cryptographic separation between the different parts of the protocol, the master key $\mathrm{K_{AiB}}$ serves only for keys derivation purposes. The same principles apply to each sub-transaction leg in the general case where there is no direct relation (i.e., shared-secret) between Ai and B, and intermediary servers Sj are required.

There are several approaches to integrate VoIP signalling with the DRCEP protocol. In the first approach, each VoIP message is encapsulated into a single Authenticated Message (AM) corresponding to a unique DRCEP transaction. Alternatively, the DRCEP protocol can also be used as an application independent protocol to establish a secure channel at the network or transport level. In that case, the single AM message received by B is sufficient to perform authentication and session key establishment with Ai. The third approach differs from the previous one by encapsulating the `SIP-INVITE` request into the AM message that also serves in establishing the secure channel.

3.2. Notations

The notations used in this paper are summarised in Table 1.

3.3. Case 1: protocol run between two entities A and B

We assume entity A is the initiator and entity B is the responder. They share a secret consisting of two pieces of information: the master key $\mathrm{K_{AB}}$ and $\mathrm{TI_B}$ which is the current transaction index of the responder entity B. This index is iterated at each new transaction through a public algorithm.

From the (current) $\mathrm{TI_B}$ value, entities A and B compute independently the transaction identifier $\mathrm{TRID(TI_B)}$ which is the public image of the transaction index. This identifier shall depend on the $\mathrm{TI_B}$ value and be the result of a cryptographic function such that observing any number of previous identifiers does not reveal any information on the current $\mathrm{TI_B}$ value nor on its future values. Hence we propose to apply a one-way hash function H to the concatenation of the transaction index $\mathrm{TI_B}$ and a public constant:

$$\mathrm{TRID(TI_B)} \leftarrow \mathrm{H(C_1 \| TI_B)} \tag{1}$$

Then, the dynamic filtering value issued by A and expected by B for this transaction is computed from the transaction identifier. For now, a strict equality is enforced:

$$\mathrm{FV(TI_B, A)} \leftarrow \mathrm{TRID(TI_B)} \tag{2}$$

The session key $\mathrm{SK(TI_B, A)}$ established between A and B for this transaction is computed independently by both entities from (at least) the master key $\mathrm{K_{AB}}$ and the (current) transaction index $\mathrm{TI_B}$ of the responder. The session key shall be the result of a cryptographic function such that knowing past session keys and the corresponding transaction indexes does not enable the adversary to determine the master key $\mathrm{K_{AB}}$. In this context, and further on, impossible means *computationally non-feasible with non-negligible probability*. Hence we propose to encrypt the concatenation of the transaction index $\mathrm{TI_B}$ and a public constant under the master key $\mathrm{K_{AB}}$:

$$\mathrm{SK(TI_B, A)} \leftarrow \mathrm{\{C_2 \| TI_B\}_{K_{AB}}} \tag{3}$$

From this material, A is able to send to B one of the two following Authenticated Messages (AM). The first message (4) supports integrity protection only while the second (5) adds confidentiality protection of the original message OM:

$$\mathrm{AM : A \rightarrow B : FV(TI_B, A), OM, MAC_{IK(TI_B, A)}(\Sigma)} \tag{4}$$

$$\mathrm{AM : A \rightarrow B : FV(TI_B, A), \{OM\}_{CK(TI_B, A)}, MAC_{IK(TI_B, A)}(\Sigma)} \tag{5}$$

In message (4), the $\mathrm{IK(TI_B, A)}$ key used for integrity protection may be the session key itself or a key derived from it. In message (5) the $\mathrm{CK(TI_B, A)}$ and $\mathrm{IK(TI_B, A)}$ keys may be derived from the session key. Alternatively, we suggest that these keys are obtained from $\mathrm{K_{AB}}$ and $\mathrm{TI_B}$ in the same way as the session key:

$$\mathrm{IK}(\mathrm{TI_B}, \mathrm{A}) \leftarrow \{\mathrm{C_3} \| \mathrm{TI_B}\}_{\mathrm{K_{AB}}} \quad (6)$$

$$\mathrm{CK}(\mathrm{TI_B}, \mathrm{A}) \leftarrow \{\mathrm{C_4} \| \mathrm{TI_B}\}_{\mathrm{K_{AB}}} \quad (7)$$

In a simplified mode, $\mathrm{CK}(\mathrm{TI_B}, \mathrm{A})$ and $\mathrm{IK}(\mathrm{TI_B}, \mathrm{A})$ do not depend on the (current) transaction index $\mathrm{TI_B}$ and thus are the same for all the transactions. Since changing these keys at each transaction only requires two symmetric encryptions and increases the overall security, the simplified mode is not further retained.

When B receives one of the two messages (4, 5), it checks the received transaction identifier $\mathrm{TRID}(\mathrm{TI_B})$ (that equals the $\mathrm{FV}(\mathrm{TI_B}, \mathrm{A})$ value), against the expected one for its current transaction index $\mathrm{TI_B}$. If this check fails the AM message is silently ignored, else B infers the corresponding transaction material $\mathrm{SK}(\mathrm{TI_B}, \mathrm{A})$, $\mathrm{IK}(\mathrm{TI_B}, \mathrm{A})$ and $\mathrm{CK}(\mathrm{TI_B}, \mathrm{A})$. From this material B can check the AM message and decrypt OM. For performance optimisation, B *can pre-compute all the transaction material*, from the current transaction index $\mathrm{TI_B}$ and the master key $\mathrm{K_{AB}}$, prior to receiving the AM message.

Once the current transaction is completed, A and B iterate independently the transaction index $\mathrm{TI_B}$. The simplest option is to apply a linear function, like adding a constant to the previous $\mathrm{TI_B}$ value. When multiple originators are considered, it is desirable to link the transaction index with the current time value t:

$$\mathrm{TI_B}(t) \leftarrow \mathrm{BTI_{B,0}} + t \quad (8)$$

In Equation (8) we only consider discrete values for time thus t is rounded with the time precision δt. The $\mathrm{BTI_{B,0}}$ value corresponds to the Base Transaction Index set up by B at time $t = 0$ and shared with A. From the initial shared-secret $\mathrm{BTI_{B,0}}$ and the public parameter δt, A and B can simultaneously compute the current transaction index $\mathrm{TI_B}(t)$, assuming their respective clocks are synchronised. From this setting, if entity A or B is compromised, the attacker obtains ($\mathrm{K_{AB}}$, $\mathrm{BTI_{B,0}}$) and thus can recover all the previous session keys. To solve this limitation and achieve the PFS (Perfect Forward Secrecy) property, we propose to iterate the base transaction index of B every Θ seconds with a one way function:

$$\mathrm{TI_B}(t) \leftarrow \mathrm{BTI_{B,p}} + t \quad (9)$$

$$\mathrm{BTI_{B,p}} \leftarrow \mathrm{H}(\mathrm{C_0} \| \mathrm{BTI_{B,p-1}}) \text{ every } \Theta \text{ seconds with } \mathrm{BTI_{B,1}} \leftarrow \mathrm{H}(\mathrm{C_0} \| \mathrm{BTI_{B,0}}) \quad (10)$$

From equation (9), the current transaction index of B is the sum of the p^{th} base transaction index $\mathrm{BTI_{B,p}}$ and the current time t (rounded with precision δt). The value $\mathrm{BTI_{B,p}}$ is obtained recursively from $\mathrm{BTI_{B,0}}$ by applying a one-way function every Θ seconds. Assuming each entity can erase $\mathrm{BTI_{B,p-1}}$ after the iteration and H is a secure one-way function, if A or B is compromised at time t_C the attacker obtains $\mathrm{BTI_{B,p}}$ but can not recover $\mathrm{BTI_{B,p-1}}$. This means the PFS property is achieved except for the last (t_C mod Θ) seconds.

From this setting, it is clear that the index $\mathrm{TI_B}(t)$ computed by A when creating the transaction is not equal to the one computed by B when receiving the AM message (except if the transmission delay is lower than the time precision δt). For this reason, and also to accommodate for possible transactions loss or disordering, B shall maintain a *transaction window* of several Acceptable Transaction Indexes $\mathrm{ATI_{B,k}}(t)$ at time t:

$$\mathrm{ATI_{B,k}}(t) \leftarrow \mathrm{BTI_{B,p}} + t + \mathrm{k} \times \delta t \quad (11)$$

In Equation (11), t is the current time value rounded with precision δt, $\mathrm{BTI_{B,p}}$ is the current base transaction index of B and k is a relative integer in the range $[\mathrm{k_{min}}, \mathrm{k_{max}}]$ with $\mathrm{k_{min}} < 0 < \mathrm{k_{max}}$. The value $\Delta = \mathrm{k_{max}} - \mathrm{k_{min}} + 1$ is defined as the window size and corresponds to the (constant) number of acceptable transaction indexes maintained by B. The $\mathrm{BTI_{B,p}}$ value is iterated every Θ seconds according to (10). In addition to the $\mathrm{ATI_{B,k}}$ values, the transaction window shall also include the Δ corresponding transaction identifiers (defined by Equation (1)):

$$\mathrm{TRID}(\mathrm{ATI_{B,k}}) \leftarrow \mathrm{H}(\mathrm{C_1} \| \mathrm{ATI_{B,k}}) \quad (12)$$

When B receives the AM message, it compares the value of $\mathrm{TRID}(\mathrm{TI_B})$ issued by A ($\mathrm{TRID}(\mathrm{TI_B}) = \mathrm{FV}(\mathrm{TI_B}, \mathrm{A})$) to the Δ values $\mathrm{TRID}(\mathrm{ATI_{B,k}})$ contained in the transaction window at the current time t. If a match is found, B infers the corresponding transaction index $\mathrm{TI_B}$ used by A and the remaining transaction material $\mathrm{SK}(\mathrm{TI_B}, \mathrm{A})$, $\mathrm{IK}(\mathrm{TI_B}, \mathrm{A})$, $\mathrm{CK}(\mathrm{TI_B}, \mathrm{A})$. If no match is found, then AM is silently ignored.

Since the $\mathrm{ATI_{B,k}}$ transaction indexes depend on t, the transaction window is a sliding window those values are changing along the time. Every δt seconds, the value t in Equation (11) changes which implies that the previous $\mathrm{ATI_{B,k_{min}}}$ transaction index is no longer valid and that a new transaction index $\mathrm{ATI_{B,k_{max}}}$ is computed. The range of k (from $\mathrm{k_{min}}$ to $\mathrm{k_{max}}$ with $\mathrm{k_{min}} < 0 < \mathrm{k_{max}}$) should be chosen to accommodate network delays and possible clock drift between A and B. In practise, this means that the transaction window shall contain more past transaction indexes than future transaction indexes ($|\mathrm{k_{min}}| > |\mathrm{k_{max}}|$). For example: $\delta t = 10\mathrm{ms}$, $\Delta = 1000$, $\mathrm{k_{min}} = -700$ and $\mathrm{k_{max}} = 299$ accommodates for possible clock drift and transmission delay of several seconds between A and B. On its side, the originator entity A does not have to maintain a transaction window. It just has to know the public values δt and Θ of B and to maintain a reasonable synchronisation with B clock. Assuming these hypothesis are met, A has the assurance that the transaction index $\mathrm{TI_B}(t)$ (cf. Equation 9) matches one of the Δ acceptable $\mathrm{ATI_{B,k}}$ values maintained by B when it receives the AM message.

In the other way round, B can send to A an Authenticated Message (AM) by performing the same operations, assuming B knows the current Base Transaction Index $\mathrm{BTI_{A,p}}$ of A.

3.4. Case 2: involvement of a single server S

In this section, we assume that A has no association with B but that is has an association with S and that S has an association with B. This means that A shares the secret ($\mathrm{K_{AS}}$, $\mathrm{BTI_{S,p}}$) with S; entity A also knows the public parameters (δt, Θ) of

S. Similarly, S shares the secrets ($\mathrm{K_{SB}}$, $\mathrm{BTI_{B,p'}}$) with B and knows the public parameters of B. For simplicity purpose, we assume that the public parameters of S and B (δt, Θ) are the same and that $\mathrm{p} = \mathrm{p'}$, but this assumption does not restrict the protocol scope.

As a first step, A sends an authorisation query (AuthQ message) to S to authenticate and ask for transaction material to contact B. If S accepts the query it returns an authorisation response (AuthR message) with the requested transaction material and then A can send the corresponding AM message to B. The first part of the transaction (messages between A and S) follows the same security principles as in Section 3.3 by using dynamic filtering values and the second part (AM message from A to B) is identical to the operations described previously.

To achieve the first part of the transaction, entity A derives from the current transaction index $\mathrm{TI_S}(t)$ of S two transaction identifiers $\mathrm{TRID}(\mathrm{TI_S})$, $\mathrm{TRID'}(\mathrm{TI_S})$ and the corresponding filtering values. This information is computed according to Equations (9, 1, 2):

$$\mathrm{TI_S}(t) \leftarrow \mathrm{BTI_{S,p}} + t \tag{13}$$

$$\mathrm{TRID}(\mathrm{TI_S}) \leftarrow \mathrm{H}(\mathrm{C_1} \| \mathrm{TI_S}), \mathrm{TRID'}(\mathrm{TI_S}) \leftarrow \mathrm{H}(\mathrm{C'_1} \| \mathrm{TI_S}) \tag{14}$$

$$\mathrm{FV}(\mathrm{TI_S}, \mathrm{A}) \leftarrow \mathrm{TRID}(\mathrm{TI_S}), \mathrm{FV}(\mathrm{TI_S}, \mathrm{S}) \leftarrow \mathrm{TRID'}(\mathrm{TI_S}) \tag{15}$$

The base transaction index $\mathrm{BTI_{S,p}}$ of S in Equation (13) is iterated every Θ seconds according to Equation (10) to achieve the PFS property. Then, from $\mathrm{TI_S}$ and $\mathrm{K_{AS}}$, entity A derives the keys $\mathrm{SK}(\mathrm{TI_S}, \mathrm{A})$, $\mathrm{CK}(\mathrm{TI_S}, \mathrm{A})$, $\mathrm{IK}(\mathrm{TI_S}, \mathrm{A})$ and $\mathrm{IK}(\mathrm{TI_S}, \mathrm{S})$ by following the same principles as in Equations (3, 6, 7):

$$\mathrm{SK}(\mathrm{TI_S}, \mathrm{A}) \leftarrow \{\mathrm{C_2} \| \mathrm{TI_S}\}_{\mathrm{K_{AS}}} \tag{16}$$

$$\mathrm{IK}(\mathrm{TI_S}, \mathrm{A}) \leftarrow \{\mathrm{C_3} \| \mathrm{TI_S}\}_{\mathrm{K_{AS}}}, \mathrm{IK}(\mathrm{TI_S}, \mathrm{S}) \leftarrow \mathrm{IK}(\mathrm{TI_S}, \mathrm{A}) \oplus \mathrm{C'_3} \tag{17}$$

$$\mathrm{CK}(\mathrm{TI_S}, \mathrm{A}) \leftarrow \{\mathrm{C_4} \| \mathrm{TI_S}\}_{\mathrm{K_{AS}}} \tag{18}$$

Then A sends the AuthQ message (see Equation (19)) to S. In this message, Ident contains the transaction information, especially the identity of the target entity B and a nonce. For confidentiality purpose, Ident is encrypted with the symmetric key $\mathrm{CK}(\mathrm{TI_S}, \mathrm{A})$:

$$\mathrm{AuthQ} : \mathrm{A} \rightarrow \mathrm{S} : \mathrm{FV}(\mathrm{TI_S}, \mathrm{A}), \{\mathrm{Ident}\}_{\mathrm{CK}(\mathrm{TI_S}, \mathrm{A})}, \mathrm{MAC}_{\mathrm{IK}(\mathrm{TI_S}, \mathrm{A})}(\Sigma) \tag{19}$$

On its side, S maintains at time t a transaction window of Δ consecutive acceptable transaction indexes $\mathrm{ATI_{S,k}}(t)$, computed in the same way as (11). In Equation (20) below, $\mathrm{k} \in [\mathrm{k_{min}}, \mathrm{k_{max}}]$ and the base transaction index $\mathrm{BTI_{S,p}}$ is iterated every Θ seconds according to (10). The transaction window also contains the transaction identifiers and filtering values for each $\mathrm{ATI_{S,k}}(t)$:

$$\mathrm{ATI_{S,k}}(t) \leftarrow \mathrm{BTI_{S,p}} + t + \mathrm{k} \times \delta t \tag{20}$$

$$\mathrm{TRID_k}(\mathrm{TI_S}) \leftarrow \mathrm{H}(\mathrm{C_1} \| \mathrm{ATI_{S,k}}), \mathrm{TRID'_k}(\mathrm{TI_S}) \leftarrow \mathrm{H}(\mathrm{C'_1} \| \mathrm{ATI_{S,k}}) \tag{21}$$

$$\mathrm{FV_k}(\mathrm{TI_S}, \mathrm{A}) \leftarrow \mathrm{TRID_k}(\mathrm{TI_S}), \mathrm{FV_k}(\mathrm{TI_S}, \mathrm{S}) \leftarrow \mathrm{TRID'_k}(\mathrm{TI_S}) \tag{22}$$

Upon reception of the AuthQ message, S compares the received $\mathrm{FV}(\mathrm{TI_S}, \mathrm{A})$ to the Δ acceptable values $\mathrm{FV_k}(\mathrm{TI_S}, \mathrm{A})$ pre-computed in the current transaction window. If no match is found, the AuthQ message is silently ignored else S infers the transaction index $\mathrm{TI_S}$ used by A and computes the related key $\mathrm{IK}(\mathrm{TI_S}, \mathrm{A})$ (according to Equation (17)) from which it can check the MAC of the AuthQ request.

Assuming the MAC check is correct, then S computes the $\mathrm{CK}(\mathrm{TI_S}, \mathrm{A})$ key (according to Equation (18)) from which it can decrypt the Ident information and obtains the receiver identity B. Assuming S shares the secrets ($\mathrm{BTI_{B,p}}$, $\mathrm{K_{SB}}$) with B it computes the transaction material $\mathrm{FV}(\mathrm{TI_B}, \mathrm{S})$, $\mathrm{SK}(\mathrm{TI_B}, \mathrm{S})$ related to the current transaction index $\mathrm{TI_B}(t)$ of B (according to Equations (9, 10, 1, 2, 3)). Then S returns this Transaction Material $\mathrm{TM_A}$ to A through the AuthR message below. Prior to crafting the AuthR message, S needs to compute the remaining keys $\mathrm{IK}(\mathrm{TI_S}, \mathrm{S})$, $\mathrm{SK}(\mathrm{TI_S}, \mathrm{A})$ (according to Equations (17, 16)) for the sub-transaction leg with A.

$$\mathrm{AuthR} : \mathrm{S} \rightarrow \mathrm{A} : \mathrm{FV}(\mathrm{TI_S}, \mathrm{S}), \mathrm{TM_A}, \mathrm{MAC}_{\mathrm{IK}(\mathrm{TI_S}, \mathrm{S})}(\Sigma) \tag{23}$$

$$\mathrm{TM_A} \leftarrow \mathrm{SK}(\mathrm{TI_S}, \mathrm{A}) \oplus (\mathrm{FV}(\mathrm{TI_B}, \mathrm{S}) \| \mathrm{SK}(\mathrm{TI_B}, \mathrm{S})) \tag{24}$$

In Equation (24), a one-time pad operation is performed with the key $\mathrm{SK}(\mathrm{TI_S}, \mathrm{A})$ that is shared between A and S for this specific transaction and which is renewed at each transaction. The value of the $\mathrm{SK}(\mathrm{TI_S}, \mathrm{A})$ key is added (modulo 2) to the concatenation of information required by A to contact B, that is the Filtering Value $\mathrm{FV}(\mathrm{TI_B}, \mathrm{S})$ and the Session Key $\mathrm{SK}(\mathrm{TI_B}, \mathrm{S})$). For Equation (24) to be valid, we assume that $\mathrm{len}(\mathrm{SK}(\mathrm{TI_S}, \mathrm{A})) = \mathrm{len}(\mathrm{FV}(\mathrm{TI_B}, \mathrm{S})) + \mathrm{len}(\mathrm{SK}(\mathrm{TI_B}, \mathrm{S}))$.

When entity A receives the AuthR message, it first verifies that the received $\mathrm{FV}(\mathrm{TI_S}, \mathrm{S})$ value matches the expected one for the transaction of index $\mathrm{TI_S}$ it has initiated with S (cf. Equations (14, 15)). If this is the case, A uses the pre-computed $\mathrm{IK}(\mathrm{TI_S}, \mathrm{S})$ key to check the MAC of AuthR. If the MAC is valid, A extracts the required transaction material to contact B, by applying the specific $\mathrm{SK}(\mathrm{TI_S}, \mathrm{A})$ session key to the received $\mathrm{TM_A}$ value:

$$\mathrm{FV}(\mathrm{TI_B}, \mathrm{S}) \| \mathrm{SK}(\mathrm{TI_B}, \mathrm{S}) \leftarrow \mathrm{SK}(\mathrm{TI_S}, \mathrm{A}) \oplus \mathrm{TM_A} \tag{25}$$

From the $\mathrm{SK}(\mathrm{TI_B}, \mathrm{S})$ session key, A derives two specific keys $\mathrm{CK}(\mathrm{TI_B}, \mathrm{A})$, $\mathrm{IK}(\mathrm{TI_B}, \mathrm{A})$ and sends the following AM message to B:

$$\mathrm{AM} : \mathrm{A} \rightarrow \mathrm{B} : \mathrm{FV}(\mathrm{TI_B}, \mathrm{S}), \{\mathrm{OM}\}_{\mathrm{CK}(\mathrm{TI_B}, \mathrm{A})}, \mathrm{MAC}_{\mathrm{IK}(\mathrm{TI_B}, \mathrm{A})}(\Sigma) \tag{26}$$

When entity B receives the AM message, it first checks that the received $\mathrm{TRID}(\mathrm{TI_B})$ value ($\mathrm{TRID}(\mathrm{TI_B}) = \mathrm{FV}(\mathrm{TI_B}, \mathrm{S})$) matches one of the pre-computed transaction identifiers of the current transaction window. At time t, the transaction window of B contains the Δ acceptable transaction indexes $\mathrm{ATI_{B,k}}(t)$ and the corresponding transaction identifiers (cf. Equations 11, 12). If a match is found, B computes (or recovers) the corresponding session key $\mathrm{SK}(\mathrm{TI_B}, \mathrm{S})$ from which it derives the

$\mathrm{CK}(\mathrm{TI_B}, \mathrm{A})$ and $\mathrm{IK}(\mathrm{TI_B}, \mathrm{A})$ keys for completing the AM verification.

Note: it is possible to include in the AuthR message an authenticator $\mathrm{TRCheck}(\mathrm{TI_B}, \mathrm{S})$ generated by S to prove to B that S has authenticated and allowed the transaction of index $\mathrm{TI_B}$ for entity A. This authenticator shall be constructed in such a way that it can not be manipulated by A and also that it can not be used by an adversary to impersonate A. For this purpose, the authenticator shall include at least a MAC based on a symmetric key shared between S and B. We propose to compute this authenticator in the following way:

$$\mathrm{TRCheck}(\mathrm{TI_B}, \mathrm{S}) \leftarrow \mathrm{MAC}_{\mathrm{VK}(\mathrm{TI_B}, \mathrm{S})}(\mathrm{ID_A}, @\mathrm{IP_A}, \mathrm{ID_B}, @\mathrm{IP_B}, \mathrm{TI_B}) \quad (27)$$

In the above equation, $\mathrm{ID_E}$ and $@\mathrm{IP_E}$ are respectively the identity and IP address of entity E and $\mathrm{VK}(\mathrm{TI_B}, \mathrm{S})$ is a key specific to this transaction, derived as the other transaction keys (i.e., by encrypting the $\mathrm{TI_B}$ value and a public constant under the master key $\mathrm{K_{SB}}$). Since the use of the authenticator TRCheck is already described in [7], it is not repeated here (and further on) for concision purpose. Actually, the authenticator can be uncorrelated from the remaining protocol operation.

3.5. Case 3: multiple originators Ai

In Section 3.3, we assumed that B only receives AM transaction messages from A. The same assumption was made in Section 3.4 between A and S and between S and B. In practical scenario, B may be in relation with several Ai entities and receives AM messages from any Ai entity at any time. Following the protocol described in Section 3.3, this requires that B shares a master key $\mathrm{K_{AiB}}$ with each Ai. This would also require that B shares a specific transaction index with each Ai and maintains a separate transaction window but this scheme is not scalable.

To avoid this issue, we assume that each Ai knows the current base transaction index $\mathrm{BTI_{B,p}}$ of B from which it can compute the current $\mathrm{TI_B}(t)$ index (cf. Equations (9, 10)). Thus, the $\mathrm{BTI_{B,p}}$ value constitutes a shared-secret among all Ai entities. When a new entity is added to the group, it does not need to get the initial $\mathrm{BTI_{B,0}}$ value but only the current $\mathrm{BTI_{B,p}}$ value. From $\mathrm{TI_B}(t)$, Ai computes the current transaction identifier $\mathrm{TRID}(\mathrm{TI_B})$ (cf. Equation (1)). Since $\mathrm{BTI_{B,p}}$ is a shared-secret and the current time t is rounded with precision δt, entities Ai and Aj may generate the same $\mathrm{TRID}(\mathrm{TI_B})$ value within a close time interval. This means that the transaction identifier can still be used as a filtering value but does not enable to authenticate the originating entity.

To obtain distinct filtering values for two (distinct) originating entities generating a transaction towards B within the same time interval δt, it is necessary that the filtering value $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$ does not only depend on $\mathrm{TRID}(\mathrm{TI_B})$ but also on the $\mathrm{K_{AiB}}$ key. It is also necessary that B learns the identity of Ai to select the right $\mathrm{K_{AiB}}$ key. Therefore we assume that each Ai shares its identity $\mathrm{ID_{Ai}}$ with B; this identity shall be kept secret. From $\mathrm{TRID}(\mathrm{TI_B}, \mathrm{Ai})$, $\mathrm{K_{AiB}}$ and $\mathrm{ID_{Ai}}$ there are several possible schemes to produce the filtering value; we propose the following one:

$$\mathrm{TRID}(\mathrm{TI_B}) = \mathrm{P_1} \| \mathrm{P_2} \| \mathrm{P_3} \text{ with } \mathrm{len}(\mathrm{P_2}) = \mathrm{len}(\mathrm{ID_{Ai}}) \quad (28)$$

$$\mathrm{FK}(\mathrm{TI_B}, \mathrm{Ai}) \leftarrow \{\mathrm{C_5} \| \mathrm{TI_B}\}_{\mathrm{K_{AiB}}} \quad (29)$$

$$\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai}) \leftarrow \mathrm{P_1} \| \mathrm{P_2} \oplus \mathrm{ID_{Ai}} \| \mathrm{P_3} \oplus \mathrm{MAC}_{\mathrm{FK}(\mathrm{TI_B}, \mathrm{Ai})}(\mathrm{P_1} \| \mathrm{P_2} \oplus \mathrm{ID_{Ai}} \| \mathrm{TI_B}) \quad (30)$$

In Equation (28), the value $\mathrm{TRID}(\mathrm{TI_B})$ (obtained from Equation (1)) is logically split in three parts. Then, Ai computes the symmetric key $\mathrm{FK}(\mathrm{TI_B}, \mathrm{Ai})$ that it shares with B for this specific transaction of index $\mathrm{TI_B}$ (cf. Equation (29)). The filtering value represented by $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$ is obtained from $\mathrm{TRID}(\mathrm{TI_B})$ by performing the following operations: the first part of $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$ is equal to the first part of $\mathrm{TRID}(\mathrm{TI_B})$ ($\mathrm{P_1}$). The second part of $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$ is the result of the modulo 2 addition of $\mathrm{P_2}$ and $\mathrm{ID_{Ai}}$. Since the value of $\mathrm{P_2}$ changes at each transaction this ensures that the $\mathrm{ID_{Ai}}$ value is masked to a protocol eavesdropper. The third part of $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$ is the result of the modulo 2 addition of $\mathrm{P_3}$ and a MAC which depends on the $\mathrm{FK}(\mathrm{TI_B}, \mathrm{Ai})$ key. The MAC input is obtained by concatenating the first two parts of $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$ with the transaction index value $\mathrm{TI_B}$ related to this transaction.

Then, the value of the session key $\mathrm{SK}(\mathrm{TI_B}, \mathrm{Ai})$ and the keys $\mathrm{CK}(\mathrm{TI_B}, \mathrm{Ai})$, $\mathrm{IK}(\mathrm{TI_B}, \mathrm{Ai})$ are computed according to Equations (3, 6 and 7) with the key $\mathrm{K_{AiB}}$. From this security material, Ai forms the AM message (cf. Equation 5) and sends it to B.

On its side, B maintains a transaction window containing the Δ acceptable transaction indexes $\mathrm{ATI_{B,k}}$ and the corresponding transaction identifiers (cf. Equations 11, 12). When receiving a AM message, B first verifies the $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$ value by performing successively the three checks below; if one of them fails the message is silently ignored.

As a first step, B compares the $\mathrm{P_1}$ part of the received filtering value with the $\mathrm{len}(\mathrm{P_1})$ first bits of the Δ identifiers $\mathrm{TRID}(\mathrm{ATI_{B,k}})$ contained in the transaction window. If a match is found, B infers the corresponding $\mathrm{TRID}(\mathrm{ATI_{B,k}})$ value and thus the $\mathrm{TI_B}$ transaction index issued by Ai. In the second step, B performs a modulo 2 addition of $\mathrm{TRID}(\mathrm{ATI_{B,k}})$ and the received $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$ value to extract the received $\mathrm{ID_{Ai}}$ and MAC values. If the received $\mathrm{ID_{Ai}}$ value matches an existing entity identifier, B infers the corresponding $\mathrm{FK}(\mathrm{TI_B}, \mathrm{Ai})$ key. In the last step, B checks the third part of $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$ by computing the expected value $\mathrm{MAC}_{\mathrm{FK}(\mathrm{TI_B}, \mathrm{Ai})}(\mathrm{P_1} \| \mathrm{P_2} \oplus \mathrm{ID_{Ai}} \| \mathrm{TI_B})$ and by comparing it to the received one.

If the checks are successful, then the received $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$ value is valid. In other words, B has authenticated Ai and found the base parameters $\mathrm{K_{AiB}}$, $\mathrm{TI_B}$. Therefore, B can infer the corresponding transaction material $\mathrm{SK}(\mathrm{TI_B}, \mathrm{Ai})$, $\mathrm{IK}(\mathrm{TI_B}, \mathrm{Ai})$, $\mathrm{CK}(\mathrm{TI_B}, \mathrm{Ai})$ according to Equations (3, 6, 7) and eventually process the AM message, as explained in Section 3.3.

3.6. Case 4: extension to a hierarchical architecture

In the previous section we explained how a responder B can handle transactions from several originators Ai while maintaining a single transaction window for all the possible Ai entities.

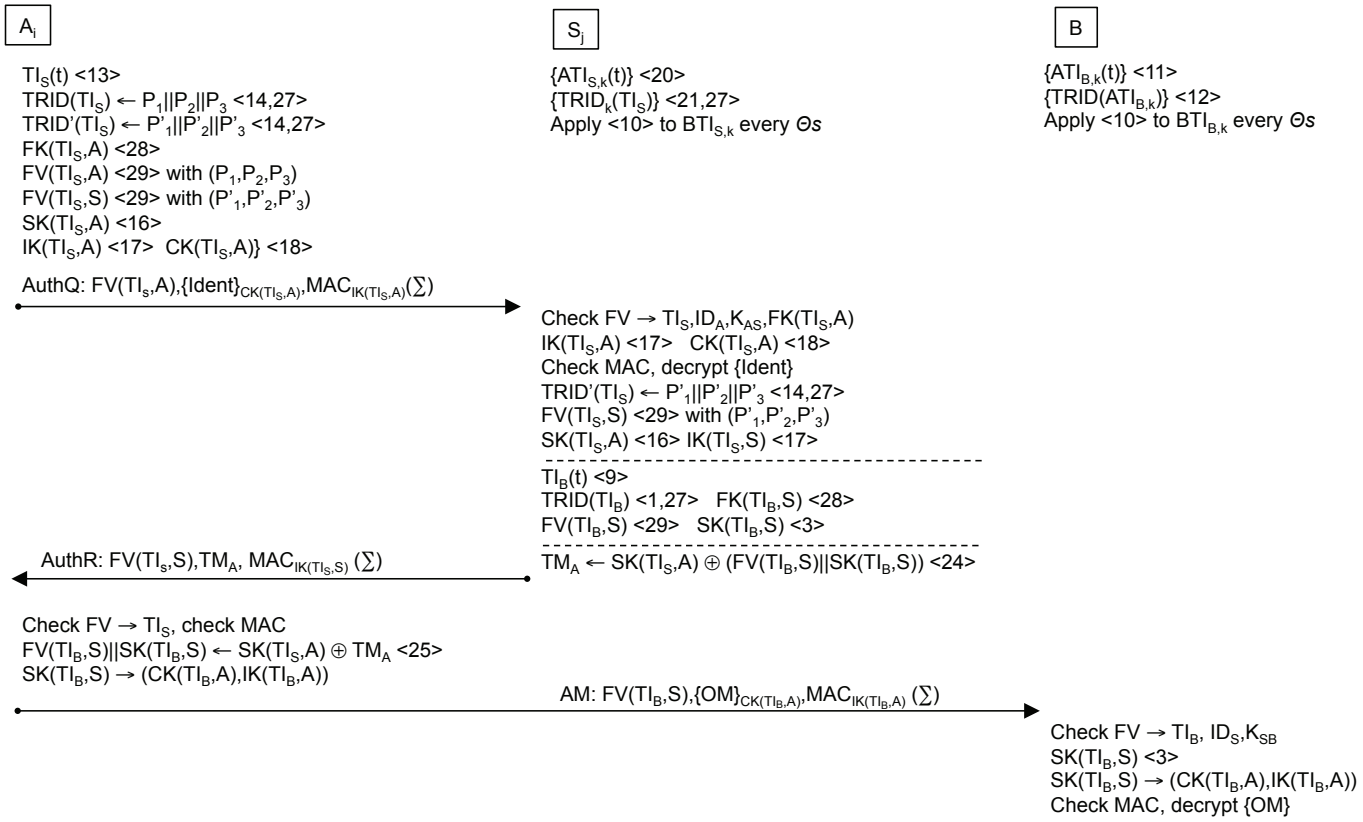

Figure 1: Description of the architecture and operations of the protocol in the three-party case. Arrows represent the exchange of messages. Remainder represent the operations computed by each principal for the generation and verification of data and keys. The corresponding equations are noted <equation number>. For simplification purpose Ai is noted A and Sj is noted S.

This requires that each Ai knows the current $\mathrm{BTI_{B,p}}$ shared-secret and has an association ($\mathrm{ID_{Ai}}$, $\mathrm{K_{AiB}}$) with B. These principles can be mixed with those of Section 3.4 resulting in a architecture where B is contacted through several authorisation servers Sj and each Sj is in relation with several originators Ai.

The resulting protocol and operations are described in Figure 1. In this architecture, B maintains a single transaction window of indexes $\mathrm{ATI_{B,k}}(t)$ computed from $\mathrm{BTI_{B,p}}$ and t (cf. Equations 11, 10). Similarly, Sj maintains a single transaction window of indexes $\mathrm{ATI_{Sj,k}}(t)$. For simplicity purpose, we assume that B and Sj share the same values t, δt, Θ, p, $\mathrm{k_{min}}$, $\mathrm{k_{max}}$.

Assuming entity Ai has a shared-secret with Sj and wants to contact B, it computes the $\mathrm{FV}(\mathrm{TI_{Sj}}, \mathrm{Ai})$ value and the corresponding AuthQ message by combining the principles of Section 3.4 and Section 3.5. If Sj recognises the AuthQ message as valid and authorises the transaction with B, it computes the transaction material ($\mathrm{FV}(\mathrm{TI_B}, \mathrm{Sj})$, $\mathrm{SK}(\mathrm{TI_B}, \mathrm{Sj})$) for the current transaction index $\mathrm{TI_B}(t)$ of B. This material is returned to A through the AuthR message where the $\mathrm{SK}(\mathrm{TI_{Sj}}, \mathrm{Ai})$ session key is used to mask the confidential information. Entity A checks the AuthR validity based on the $\mathrm{FV}(\mathrm{TI_{Sj}}, \mathrm{Sj})$ filtering value which authenticates Sj, and on the MAC code for message integrity. If the message is valid, A extracts the $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Sj})$ value and the session key $\mathrm{SK}(\mathrm{TI_B}, \mathrm{Sj})$ from which the $\mathrm{CK}(\mathrm{TI_B}, \mathrm{Ai})$ and $\mathrm{IK}(\mathrm{TI_B}, \mathrm{Ai})$ keys are derived.

This three-party architecture can be extended to a four-party architecture where the originator endpoints Ai and the outbound proxy S1 are part of network domA and the responder endpoints Bj and the inbound proxy S2 are part of network domB. Each Ai entity has an association ($\mathrm{ID_{Ai}}$, $\mathrm{K_{AiS1}}$) with S1 and knows the shared-secret $\mathrm{BTI_{S1,p}}$. Similarly, S2 has an association with each Bj and S1 has an association ($\mathrm{ID_{S1}}$, $\mathrm{K_{S1S2}}$, $\mathrm{BTI_{S2,p}}$) with S2.

The protocol presented in Table 2 describes the overall operation for one Ai endpoint of domA to contact one Bj endpoint of domB through the proxies S1 and S2. The whole transaction consists in three steps: first between Ai and S1, then between S1 and S2 and finally between Ai and Bj. Prior to sending message (32), Ai computes the transaction identifiers $\mathrm{TRID}(\mathrm{TI_{S1}})$, $\mathrm{TRID'}(\mathrm{TI_{S1}})$ based on $\mathrm{BTI_{S1,p}}$ and t (cf. Equations 13, 15). Then, the corresponding filtering values $\mathrm{FV}(\mathrm{TI_{S1}}, \mathrm{Ai})$ and $\mathrm{FV}(\mathrm{TI_{S1}}, \mathrm{S1})$ are obtained from Equations (28, 29, 30). On its side, S1 maintains a single transaction window for all Ai entities containing the Δ acceptable indexes $\mathrm{ATI_{S1,k}}(t)$ obtained from Equation (11). Upon receiving a valid $\mathrm{FV}(\mathrm{TI_{S1}}, \mathrm{Ai})$ (cf. Section 3.5), S1 infers the $\mathrm{TI_{S1}}$ transaction index issued by Ai and the corresponding transaction material: the session key $\mathrm{SK}(\mathrm{TI_{S1}}, \mathrm{Ai})$ is obtained from Equation (16) and the integrity keys $\mathrm{IK}(\mathrm{TI_{S1}}, \mathrm{Ai})$, $\mathrm{IK}(\mathrm{TI_{S1}}, \mathrm{S1})$ are obtained from Equation (17). The $\mathrm{TM_{Ai}}$ value of message (35) is computed according to Equation (24) as follows:

$$\mathrm{TM_{Ai}} \leftarrow \mathrm{SK}(\mathrm{TI_{S1}}, \mathrm{Ai}) \oplus (\mathrm{FV}(\mathrm{TI_{Bj}}, \mathrm{S2}) \| \mathrm{SK}(\mathrm{TI_{Bj}}, \mathrm{S2})) \quad (31)$$

The same principles apply between entities S1 and S2 to achieve the $\mathrm{AuthQ/AuthR}$ exchange and to generate the required values of: $\mathrm{TRID}(\mathrm{TI_{S2}})$, $\mathrm{TRID'}(\mathrm{TI_{S2}})$, $\mathrm{FV}(\mathrm{TI_{S2}}$,S1), $\mathrm{FV}(\mathrm{TI_{S2}}, \mathrm{S2})$, $\mathrm{FK}(\mathrm{TI_{S2}}, \mathrm{S1})$, $\mathrm{SK}(\mathrm{TI_{S2}}, \mathrm{S1})$, $\mathrm{IK}(\mathrm{TI_{S2}}, \mathrm{S1})$, $\mathrm{IK}(\mathrm{TI_{S2}}, \mathrm{S2})$, and $\mathrm{OTPR_{S1}}$. For these computations, the master key $\mathrm{K_{S1S2}}$ is used with the current $\mathrm{BTI_{S2,p}}$ base transaction index of S2.

When it receives a valid AuthQ message (see Equation 33), S2 computes the Bj current transaction index $\mathrm{TI_{Bj}}(t)$ from Equation (8), and the filtering value $\mathrm{FV}(\mathrm{TI_{Bj}}, \mathrm{S2})$ from Equations (1, 28, 29, 30). The final session key $\mathrm{SK}(\mathrm{TI_{Bj}}, \mathrm{S2})$ is obtained according to Equation (3). When Ai receives message (35), it extracts from $\mathrm{TM_{Ai}}$ the $\mathrm{FV}(\mathrm{TI_{Bj}}, \mathrm{S2})$, $\mathrm{SK}(\mathrm{TI_{Bj}}, \mathrm{S2})$ values and then derives $\mathrm{CK}(\mathrm{TI_{Bj}}, \mathrm{Ai})$, $\mathrm{IK}(\mathrm{TI_{Bj}}, \mathrm{Ai})$ keys from $\mathrm{SK}(\mathrm{TI_{Bj}}, \mathrm{S2})$ to craft the AM message.

From this setting, it is clear that the intermediary entities S1 and S2 know the final session key $\mathrm{SK}(\mathrm{TI_{Bj}}, \mathrm{S2})$ shared between Ai and Bj, which may be desirable for legal constraints. However, it is possible for Ai to include in the AM message a Diffie-Hellman value $\mathrm{g^{x_{Ai}}}$ where $\mathrm{x_{Ai}}$ is an ephemeral secret for this transaction. Once it has validated the received AM message, Bj completes the Diffie-Hellman exchange by computing the new session key $\mathrm{g^{x_{Ai}x_{Bj}}}$ and returning to Ai its own public value $\mathrm{g^{x_{Bj}}}$. In this way, Bj is not involved in heavy computation before having authenticated the originator Ai.

4. Security analysis

We present in this section a heuristic evaluation about the robustness of DRCEP protocol to guaranteeing the system security properties. We expect from DRCEP protocol to be robust with regard to the four main traditional security properties that any information system must guarantee [30], that are: (1) availability, (2) confidentiality, (3) integrity, and (4) authenticity. We also expect from DRCEP protocol to provide (5) PFS (Perfect Forward Secrecy) property which is less common, especially for protocols based on symmetric cryptography.

Concerning availability, the goal is to minimise the impact of (D)DoS attacks on the protocol entities, especially A and B but also on the intermediary servers (if any) which may be involved in the transaction completion.

Regarding confidentiality, the protocol must prevent any adversary from knowing the Original Message (OM) or the Session Key $\mathrm{SK}(\mathrm{TI_B}, \mathrm{A})$ established between A and B. The values of the shared-secrets ($\mathrm{TI_B}$, $\mathrm{K_{AB}}$) shall also be protected from disclosure or recovery. Additionally, the Ident information contained in the AuthQ message and the entity identifier $\mathrm{ID_E}$ incorporated in the computation of the Filtering Value (FV) shall also not be accessible to a third-party.

The integrity property concerns each message of the protocol (AuthQ, AuthR and AM). Since each message is protected by a Message Authentication Code (MAC), the underlying goal is to guarantee the confidentiality of the IK keys involved in MAC computations.

The protocol also ensures that each party (A and B) are being authenticated by their respective servers and eventually can authenticate each other along the key establishment process.

$$\mathrm{AuthQ}: \quad \mathrm{Ai} \rightarrow \mathrm{S1} \quad : \mathrm{FV}(\mathrm{TI_{S1}}, \mathrm{Ai}), \mathrm{Ident}, \mathrm{MAC}_{\mathrm{IK}(\mathrm{TI_{S1}}, \mathrm{Ai})}(\Sigma) \tag{32}$$

$$\mathrm{AuthQ}: \quad \mathrm{S1} \rightarrow \mathrm{S2} \quad : \mathrm{FV}(\mathrm{TI_{S2}}, \mathrm{S1}), \mathrm{Ident}, \mathrm{MAC}_{\mathrm{IK}(\mathrm{TI_{S2}}, \mathrm{S1})}(\Sigma) \tag{33}$$

$$\mathrm{AuthR}: \quad \mathrm{S2} \rightarrow \mathrm{S1} \quad : \mathrm{FV}(\mathrm{TI_{S2}}, \mathrm{S2}), \mathrm{TM_{S1}}, \mathrm{MAC}_{\mathrm{IK}(\mathrm{TI_{S2}}, \mathrm{S2})}(\Sigma) \tag{34}$$

$$\mathrm{AuthR}: \quad \mathrm{S1} \rightarrow \mathrm{Ai} \quad : \mathrm{FV}(\mathrm{TI_{S1}}, \mathrm{S1}), \mathrm{TM_{Ai}}, \mathrm{MAC}_{\mathrm{IK}(\mathrm{TI_{S1}}, \mathrm{S1})}(\Sigma) \tag{35}$$

$$\mathrm{AM}: \quad \mathrm{Ai} \rightarrow \mathrm{Bj} \quad : \mathrm{FV}(\mathrm{TI_{Bj}}, \mathrm{S2}), \{\mathrm{OM}\}_{\mathrm{CK}(\mathrm{TI_{Bj}}, \mathrm{Ai})}, \mathrm{MAC}_{\mathrm{IK}(\mathrm{TI_{Bj}}, \mathrm{Ai})}(\Sigma) \tag{36}$$

Table 2: Extension of the protocol to a hierarchical architecture

Finally, the PFS property concerns both passive off-line attacks (like password guessing) and entity hacking where the adversary has obtained the long term secrets (TI, $\mathrm{K_{AB}}$) of one or several parties; under these hypothesis the adversary shall still not be able to recover any of the past session keys $\mathrm{SK}(\mathrm{TI_B}, \mathrm{A})$.

4.1. Adversary capabilities

We define an adversary as a network entity with significant computational power whose primary goal is to retrieve enough information about the protocol data messages, to eventually obtain an unauthorised access to the service, impersonate a legitimate entity, decrypt and/or alter the contents of a message, or impact the service availability. In Table 3, we informally define five elementary capabilities that such an adversary may posses. These capabilities are mainly independent from each other and may be cumulative. The way the adversary can obtain these capabilities is outside the scope of this paper.

Capability CAP1 assumes an external adversary who has a complete knowledge of the protocol and is equipped with the related software. Indeed, the adversary has an instance of all the cryptographic functions required by the protocol and can invoke them at any time to craft DRCEP messages towards any entity of the protocol. We also assume that the adversary can be anywhere in the network and can spoof the network address of any legitimate entity.

Capability CAP2 assumes the adversary can read the content of any DRCEP message exchanged in the network between any pair of entities.

Capability CAP3 assumes the adversary has the entire control over all the communication channels of the DRCEP network. Consequently he can delete, delay or alter any message exchanged between any legitimate entities.

Capability CAP4 assumes the adversary has the entire control over a legitimate entity and thus can access the long-term secrets of this entity. This capability may be obtained by service subscription or by hacking a legitimate entity. In the first case, the adversary acts as an end-user entity (A or B) whereas in the second case it may also control a server entity ($\mathrm{S_i}$).

Capability	Description
CAP1	Injection of protocol messages
CAP2	Eavesdropping of messages
CAP3	Alteration of messages
CAP4	Control of protocol entities
CAP5	Obtain the values of old cryptographic keys

Table 3: Possible elementary capabilities of the adversary

Capability CAP5 assumes the adversary is able to obtain the value of a session key $\mathrm{SK}(\mathrm{TI_B}, \mathrm{A})$ used in any sufficiently old previous run of the protocol. This capability implies CAP2 (or CAP3) for the adversary to capture previous protocol information carried in the exchanged messages. By extension, this capability also includes the recovery of other transaction keys (CK, IK or FK).

As part of CAP1 (and other capabilities) the adversary can initiate communications with the other parties of the protocol, as well as replay or alter some legitimate exchanges. In this regard, the adversary is naturally exposed to failures and may eventually be detected by the authorised parties. In the sequel, we note as NA the Number of Attempts made by the adversary. An attempt is typically a call to a cryptographic function, the injection, modification, deletion of a DRCEP message, or a combination of both. We note as NM the Number of Messages observed, or intercepted by the adversary, from which it harvests the required information to launch an attack.

4.2. Selected security functions and hypothesis

The DRCEP protocol description presented in the previous section uses abstract cryptographic functions which are now instantiated for practical implementation and security evaluation. The one-way hash function $\mathrm{H}(\mathrm{V})$ is the SHA-256 function specified in [34]. The symmetric cipher $\{\mathrm{V}\}_{\mathrm{K}}$ is the AES-256 function specified in [36]. The $\mathrm{MAC_K}(\mathrm{V})$ function is based on the HMAC standard [35] with the AES-256 hash function. Given theses choices, Table 4 specifies the lengths of the keys and security parameters used in the protocol.

These lengths imply that the MAC result used in Equation (30) to compute the $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$ value is truncated to 64 bits. We assume that the MAC results incorporated in each message for integrity protection are all truncated to LMAC = 128 bits. Similarly, the results of the hash function used to compute TRID in the remainder equations are truncated to LHASH = 128 bits. Because the Transaction Identifier has a length of 120 bits, the result of the hash function in Equation (10) is truncated to LTI = 120 bits.

Concerning the cipher function, all the inputs have a length of (8 + 120) bits which is equal to the AES-256 block size and the keys have a length of 256 bits. Consequently, a slight modification is required to Equation (16) to produce the $\mathrm{SK}(\mathrm{TI_S}, \mathrm{A})$ session key which has a length of 256 bits. The modified equation is:

$$\mathrm{SK}(\mathrm{TI_S}, \mathrm{A}) \leftarrow (\{\mathrm{C_2} \| \mathrm{TI_S}\}_{\mathrm{K_{AS}}} \| \{\mathrm{C'_2} \| \mathrm{TI_S}\}_{\mathrm{K_{AS}}}) \quad (37)$$

We then assume that these three functions behave as ideal oracles, so that we can estimate the probabilities of success associated to various adversary trials. These results are then linked to the DRCEP protocol to form elementary probabilities which are used in the next section to evaluate more complex attacks. The probability results below are based from references in [30].

4.2.1. Hypothesis and possible attacks on the hash function

We assume for the considered SHA-256 function that the output $\mathrm{y} = \mathrm{H}(\mathrm{x})$ appears as random to an adversary who has no information on the preimage x.

Consequently, given an observed output TRID, the probability that a random preimage TI verifies Equation (1) is close to $2^{-\mathrm{LTI}} = 2^{-120}$. Assuming the adversary is able to observe NM distinct $\mathrm{TRID_i}$ outputs and executes NA times its own instance of the SHA-256 hash function with NA distinct and randomly chosen preimages $\mathrm{TI_j}$, the probability to guess one valid transaction index is close to $\mathrm{PE_1} = \mathrm{NM} \times \mathrm{NA} \times 2^{-120}$. If a successful $\mathrm{TI_j}$ value is found, the adversary deduces the current Base Transaction Index BTI from Equation (9).

If the previous attack is successful, the adversary knows the secret $\mathrm{BTI_{E,p}}$ of entity E and can try to invert Equation (10) to break the PFS security property. Assuming the adversary executes NA times its own instance of the SHA-256 hash function with NA distinct and randomly chosen preimages, the probability to guess $\mathrm{BTI_{E,p-1}}$ is close to $\mathrm{PE_2} = \mathrm{NA} \times 2^{-120}$.

4.2.2. Hypothesis and possible attacks on the cipher function

We assume for the considered AES-256 function that the output $\mathrm{y} = \{\mathrm{x}\}_{\mathrm{K}}$ appears as random to an adversary who has no information on the key K whatever the information he has on the plaintext x (including full knowledge of the plaintext). We denote n as the length of both ciphertext and plaintext; and k as the length of the key K ($\mathrm{n} = 128$, $\mathrm{k} = 256$ for AES-256).

Element	Length
$\mathrm{C_X}$	8
BTI	120
TI	120
TRID	128
$\mathrm{P_1}$, $\mathrm{P_2}$, $\mathrm{P_3}$	32, 32, 64
ID	32
$\mathrm{K_{AB}}$	256
FK	128
FV	128
IK	128
CK	128
$\mathrm{SK(TI_S, A)}$	256
TM	256
$\mathrm{SK(TI_B, S)}$	128

Table 4: Length of the security parameters

Consequently, if the adversary knows a set of NM plaintexts and the corresponding ciphertexts ($\mathrm{NM} \geq 3$), obtained under the same (unknown) key K, the probability to determine the ciphertext of a new plaintext (or vice versa the plaintext of a new ciphertext) is close to $\mathrm{PE_3} = 2^{-\mathrm{n}}$.

Under the same hypothesis, the probability to guess K is bounded by $\mathrm{PE_4} = \mathrm{NA} \times 2^{-\mathrm{k}}$. To that purpose, the adversary tries NA distinct and randomly chosen keys $\mathrm{K_i}$ against a given pair (x, y); if a match is found, the $(\mathrm{NM} - 1)$ remaining pairs are used to confirm the key.

4.2.3. Hypothesis and possible attacks on the MAC function

We assume for the considered MAC function that the output $\mathrm{y} = \mathrm{MAC_K(x)}$ appears as random to an adversary who has no information on the key K whatever the information he has on the preimage x (including full knowledge of the preimage). We denote respectively v, n, k as the lengths of the preimage x, the output y and the key K.

Consequently, if the adversary knows a set of NM preimages and the corresponding outputs ($\mathrm{NM} \geq 3$), obtained under the same (unknown) key K, the probability to determine the output of a new preimage is close to $\mathrm{PE_5} = 2^{-\mathrm{n}}$. Similarly, the probability to determine the preimage of a new output is close to $\mathrm{PE_6} = 2^{-\mathrm{v}}$.

Under the same hypothesis, the probability to guess K is bounded by $\mathrm{PE_7} = \mathrm{NA} \times 2^{-\mathrm{k}}$. To that purpose, the adversary tries NA distinct and randomly chosen keys $\mathrm{K_i}$ against a given pair (x, y); if a match is found, the $(\mathrm{NM} - 1)$ remaining pairs are used to confirm the key.

4.3. Security analysis of the protocol exchanges

This section provides the security analysis of the DRCEP protocol, based on the previous hypothesis for the security functions. This analysis is performed in a incremental way by following the construction steps of the protocol: (1) protocol run between two entities A and B, (2) involvement of a single server S, (3) multiple originators Ai and (4) extension to a hierarchical architecture. At each step, we consider adversaries equipped with the capabilities defined in Table 3. The analysis is heuristic rather than exhaustive in that we only consider the attacks which are the most likely to occur and have the highest probabilities of success.

4.3.1. Security of a transaction between two entities

This corresponds to the first protocol stage, described in Section 3.3, where entities A and B share the secrets ($\mathrm{K_{AB}}$, $\mathrm{BTI_{B,p}}$) and each transaction consists in a single AM message defined by Equation (5).

1. Attack to the availability property

   Attack A1 – The adversary holds capability CAP1. He tries to send AM messages to B without knowing the shared secrets; the goal is to create a (D)DoS condition on B. To this purpose, the adversary needs at least to furnish a valid $\mathrm{FV(TI_B, A)}$ value, to guess

one of the Δ acceptable $\mathrm{ATI}_{\mathrm{B,k}}$ identifiers contained in the transaction window of B. Considering NA attempts, the probability of success is bounded by $\mathrm{P}_{\mathrm{A1}} = \mathrm{NA} \times \Delta \times 2^{-120}$. If the adversary tries to craft a fully valid AM message, the P_{A1} probability is lowered by a factor of 2^{-256}.

2. Attacks to the confidentiality property

   Attack A2 – The adversary holds capability CAP2. The probability that the adversary guesses the $\mathrm{TI_B}$ secret based on the observation of NM public values $\mathrm{FV}(\mathrm{TI_B}, \mathrm{A})$, and given NA (off-line) attempts, is equal to $\mathrm{P}_{\mathrm{A2}} = \mathrm{PE}_1 = \mathrm{NM} \times \mathrm{NA} \times 2^{-120}$.

   Attack A3 – Assuming attack A2 is successful, the adversary knows the $\mathrm{BTI}_{\mathrm{B,p}}$ value at a given point in time and can infer all the future values with Equation (10). The probability in NA attempts to guess $\mathrm{BTI}_{\mathrm{B,p-1}}$ (i.e., to break the PFS property for the past Θ seconds) is equal to $\mathrm{P}_{\mathrm{A3}} = \mathrm{PE}_2 = \mathrm{NA} \times 2^{-120}$.

   Attack A4 – Assuming attack A2 is successful, the adversary knows the NM values $\mathrm{TI_B}(t)$ used in the captured AM messages. The probability in NA attempts to guess the $\mathrm{K_{AB}}$ key is equal to $\mathrm{P}_{\mathrm{A4}} = \mathrm{PE}_4 = \mathrm{NA} \times 2^{-256}$. This attack exploits Equation (6) where the plaintext $(\mathrm{C}_3 \| \mathrm{TI_B})$ is known, the key is chosen randomly, and the obtained ciphertext IK is verified by recomputing the MAC over the captured message.

3. Attack to the integrity property

   Attack A5 – The adversary holds capability CAP3. The goal of his attack is to change the OM part of an intercepted AM message which requires to have the valid $\mathrm{IK}(\mathrm{TI_B}, \mathrm{A})$ and $\mathrm{CK}(\mathrm{TI_B}, \mathrm{A})$ keys. We assume here that these keys are derived from the session key $\mathrm{SK}(\mathrm{TI_B}, \mathrm{A})$ and not from Equations (6, 7). The adversary chooses randomly NA session keys $\mathrm{SK_i}$ from which it derives the corresponding keys ($\mathrm{CK_i}$, $\mathrm{IK_i}$) and forms the $\mathrm{AM_i}$ message. Each $\mathrm{AM_i}$ message replays the $\mathrm{FV}(\mathrm{TI_B}, \mathrm{A})$ value captured in the original AM message. If the attack is repeated for NM intercepted AM messages, the probability of success is equal to $\mathrm{P}_{\mathrm{A5}} = \mathrm{NM} \times \mathrm{NA} \times 2^{-128}$.

   Notes on Attack A5:

   - Because the transaction window of B is iterated automatically along the time, the NA value can not be chosen arbitrarily high.
   - If the $\mathrm{CK_i}$ and $\mathrm{IK_i}$ keys are not derived from the session key, the P_{A5} probability is lowered by a factor of 2^{-128}.
   - To avoid sending a large amount of messages on the network, the alternative is to check each $\mathrm{SK_i}$ key against the intercepted AM message. In that case, the probability of success is lowered because an additional MAC computation is required at each trial whereas the validity period of $\mathrm{FV}(\mathrm{TI_B}, \mathrm{A})$ remains the same.

4. Attack to the authenticity property
   We assume and adversary that holds capability CAP4. If the adversary takes the control over entity A, he can send OM messages of his choice to B. However, knowing the secrets ($\mathrm{K_{AB}}$, $\mathrm{BTI}_{\mathrm{B,p}}$) does not enable him to recover the session keys established prior he has taken the control over A or B.

5. Attack to the PFS property

   Attack A6 – We assume an adversary who is holding capability CAP5 and who has recovered NM session keys $\mathrm{SK}(\mathrm{TI_B}, \mathrm{A})$ along with the corresponding filtering values $\mathrm{FV}(\mathrm{TI_B}, \mathrm{A})$. To guess simultaneously the secret key $\mathrm{K_{AB}}$ and the transaction index $\mathrm{TI_B}$, the adversary chooses randomly NA candidate keys $\mathrm{K_i}$ and for each $\mathrm{K_i}$ computes the plaintext $(\mathrm{C}_2 \| \mathrm{TI_i})$ of the first session key. If applying Equations (1, 2) to $\mathrm{TI_i}$ returns the expected FV value then a match has been found and the remaining $(\mathrm{NM} - 1)$ pairs $(\mathrm{SK}(\mathrm{TI_B}, \mathrm{A}), \mathrm{FV}(\mathrm{TI_B}, \mathrm{A}))$ are used for confirmation. Similarly to PE_4, the probability of success is $\mathrm{P}_{\mathrm{A6}} = \mathrm{NA} \times 2^{-256}$.

4.3.2. Security of a transaction between three entities

This corresponds to the second protocol stage, described in Section 3.4, involving entity S and resulting in a additional AuthQ/AuthR message exchange between A and S. Compared to the security analysis for the AM message, the main difference is that the keys $\mathrm{IK}(\mathrm{TI_B}, \mathrm{A})$ and $\mathrm{CK}(\mathrm{TI_B}, \mathrm{A})$ must be derived from the session key $\mathrm{SK}(\mathrm{TI_B}, \mathrm{S})$ (i.e., Equations 6, 7 are no longer applicable). Note: the possible attacks on B secrets or on the AM message are the same as in Section 4.3.1 and are not repeated here.

1. Attack to the availability property

   Attack A7 – We assume an adversary holding capability CAP1. The probability in NA attempts that the adversary generates a AuthQ or AuthR message with at least a valid FV value is equal to $\mathrm{P}_{\mathrm{A7}} = \mathrm{NA} \times \Delta \times 2^{-120}$. For a fully valid message, the P_{A7} probability is lowered by a factor of 2^{-256}.

2. Attacks to the confidentiality property

   Attack A8 – We assume an adversary holding capability CAP2. The probability in NA (off-line) attempts to guess the $\mathrm{TI_S}$ secret, based on the observation of NM AuthQ (or AuthR) messages, is equal to $\mathrm{P}_{\mathrm{A8}} = \mathrm{NM} \times \mathrm{NA} \times 2^{-120}$.

   Attack A9 – Assuming Attack A8 is successful, the probability to guess the previous $\mathrm{BTI}_{\mathrm{S,p-1}}$ value knowing $\mathrm{BTI}_{\mathrm{S,p}}$ is equal to $\mathrm{P}_{\mathrm{A9}} = \mathrm{NA} \times 2^{-120}$.

   Attack A10 – Assuming Attack A8 is successful, the probability to guess the $\mathrm{K_{AS}}$ key is equal to $\mathrm{P}_{\mathrm{A10}} = \mathrm{NA} \times 2^{-256}$.

Note that, based on Equation (24), if the adversary observes the $\mathrm{TM_A}$ value in the AuthR message and the corresponding $\mathrm{FV(TI_B, S)}$ value in the AM message, he can infer $\mathrm{len(FV(TI_B, S))}$ bits of the $\mathrm{SK(TI_S, A)}$ key. Since the attacker only has a partial knowledge of the key $\mathrm{SK(TI_S, A)}$, it is not possible to launch an attack similar to Attack A6 (which would have lowered the combined probabilities of Attacks A8 and A10).

3. Attack to the integrity property

   Attack A11 – The adversary holds capability CAP4. His goal is to change the Ident part of an intercepted AuthQ message, for example to modify the responder identity. It requires a valid pair of $\mathrm{IK(TI_B, A)}$ and $\mathrm{CK(TI_B, A)}$ keys; since these keys depend on both $\mathrm{TI_S}$ and $\mathrm{K_{AS}}$, the highest probability is obtained when trying a random guess independently on each key value, i.e., $\mathrm{P_{A11} = NM \times NA \times 2^{-128} \times 2^{-128}}$.

4. Attacks to the authenticity property

   Attack A12 – Assuming an adversary holding capability CAP4, and who has taken the control over A, he can initiate communications with B and obtains a set of NM valid pairs ($\mathrm{FV(TI_B, S)}$, $\mathrm{SK(TI_B, S)}$). Following the same approach as for Attack A6, the probability to guess the $\mathrm{K_{SB}}$ key in NA attempts is equal to: $\mathrm{P_{A12} = NA \times 2^{-256}}$.

   Attack A13 – We assume the adversary has now taken the control over B and has the capability CAP2 to observe the messages exchanged between A and S. If the adversary receives NM transactions from A, he can recover the corresponding keys $\mathrm{SK(TI_S, A)}$ with Equation (24). Since the adversary also knows the corresponding $\mathrm{FV(TI_S, S)}$ values, the probability to guess the $\mathrm{K_{AS}}$ key in NA attempts is equal to: $\mathrm{P_{A13} = NA \times 2^{-256}}$ (same method as for Attack A6 applied to Equation (16)).

   Note that, if the adversary has taken the control over S, he knows the shared secrets of A and B and can manipulate all the communications but he can not easily recover past session keys.

5. Attack to the PFS property
   Assuming an adversary holding capability CAP5, who has obtained a set of NM keys established between A and S (SK or IK or CK keys) along with the corresponding filtering values FV, then the adversary has the same probability of success as in Attack A13 to find the $\mathrm{K_{AS}}$ key. Depending on which key he holds (SK or IK or CK), he will use Equation (16) or (17) or (18).
   If the adversary has obtained a set of NM session keys established between S and B along with the corresponding filtering values FV this returns to Attack A12.

4.3.3. Security of concurrent access to the same transactions window

This corresponds to the third protocol stage, described in Section 3.5 where B may be contacted concurrently by several $\mathrm{A_i}$. Compared to the analysis made in Section 4.3.1, the only difference is the computation of the Filtering Value FV by each originator and its verification by B (cf. Equations (28, 29, 30)). This results in the following changes compared to the previous security analysis:

1. Attack to the availability property
   Whereas guessing a valid Transaction Index (TI) value was sufficient in the previous stages to attack the availability property, this is no longer the case. Therefore, if we assume an adversary who is holding capability CAP1, he would also need now a valid entity identifier ($\mathrm{ID_E}$) and a valid Filtering Key (FK). Hence, the best probability of success is to bypass TI and to guess the 128 bits of an acceptable FV value, which gives: $\mathrm{NA \times \Delta \times 2^{-128}}$.
2. Attacks to the confidentiality property
   Now the observed FV value is no longer equal to the TRID value. The first 32 bits remain the same (cf. part P1 in Equation 30) and can be used as a verifier for each $\mathrm{TI_i}$ value chosen randomly by an adversary with capability CAP2. The probability of success is still equal to $\mathrm{NM \times NA \times 2^{-120}}$ but the adversary would have to observe more messages because the verifier length is shorter.

   Assuming the adversary has successfully guessed the $\mathrm{TI_B}$ value (and thus the current $\mathrm{BTI_{B,p}}$ value) of a responder, he can launch the two new attacks below:

   Attack A14 – Let us assume the adversary observes one AM message sent by one legitimate entity $\mathrm{A_i}$ to B. From Equation (30) he can infer the $\mathrm{ID_{Ai}}$ identifier of $\mathrm{A_i}$; then the adversary can send AM messages to B while hijacking $\mathrm{A_i}$ identity. Although each AM message will be rejected by B (because only parts $\mathrm{P_1}$ and $\mathrm{P_2}$ of the filtering value are valid) this would force B to perform MAC verifications on the received FV values. If the adversary replicates this attack for a large number of $\mathrm{A_i}$, this may create a (D)DoS situation on B.

   Attack A15 – We assume the adversary observes NM filtering values FV for NM distinct AM messages. From the $\mathrm{P_3}$ part of each FV value, the adversary can recover the MAC result computed with the FK key (cf. Equation 30)). Because the adversary knows $\mathrm{TI_B}$, he can recover the MAC preimage. Consequently, an off-line brute force attack is possible on the master key $\mathrm{K_{AiB}}$: for each candidate key $\mathrm{K_i}$ the adversary derives the corresponding FK key and checks if it is valid. The success probability is: $\mathrm{P_{A15} = PE_7 = NA \times 2^{-256}}$.

3. Attack to the integrity property
   The probability of success remains equivalent to the result presented in Section 4.3.1, Attack A5.
4. Attack to the authenticity property
   Assuming an adversary holding capability CAP4 and who has taken the control over a legitimate entity $\mathrm{A_i}$, he then knows the shared-secret $\mathrm{TI_B}$ of B. Hence, he can launch

attacks A14 and A15 with the same probabilities of success.

5. Attack to the PFS property
   Assuming an adversary holding capability CAP5 who has recovered NM filtering keys $\mathrm{FK}(\mathrm{TI_B}, \mathrm{Ai})$ along with the corresponding filtering values $\mathrm{FV}(\mathrm{TI_B}, \mathrm{Ai})$, he can launch attack A6 with the same probability of success (by applying Equation (29) instead of Equation (3)) and then verify the obtained TI transaction index with equation (1).

4.3.4. Security in the general case

The general case corresponds to the protocol described in Section 3.6. The resulting security analysis combines the results of section 4.3.2 for the three-party setting and those of section 4.3.3 where a responder (B or S) may be in relation with several originators (S or A). In the general case, the originator A may contact a first authorisation server $\mathrm{S_1}$ which may contact another one until a $\mathrm{S_j}$ server in relation with B is found. Each time a new server is contacted, this creates an additional sub-transaction with an $\mathrm{AuthQ/AuthR}$ exchange and finally the AM message is sent from A to B. Even if multiple sub-transactions are created, the security properties of the DRCEP protocol remain the same for each individual entity because the protocol principles are repeated without being modified. As a summary of this section, the security analysis has shown that:

- Using filtering values FV in each protocol message (i.e., AuthQ, AuthR and AM) protects from blind attacks since the adversary can not initiate any transaction without knowing the expected FV value. Furthermore, even if the adversary recovers a valid FV value it becomes rapidly invalid because the transaction window changes automatically along the time.
- The PFS property is ensured because the session key SK and each transaction key (CK, IK, FK) depend on both the master key $\mathrm{K_{AB}}$ and the one-way transaction index $\mathrm{TI_B}$.
- Various attacks are possible on the protocol, like: (1) guessing the TI secret based on a set of observed identifiers, (2) inverting a valid BTI value, (3) guessing the master key $\mathrm{K_{AB}}$ when a set of valid TI values is known or vice-versa when a set of transaction keys (SK, IK, CK, FK) is known. The success probability of each attack is conversely proportional to 2^{L} where L is the size of the security parameter (key or transaction index). This means the success probability of each attack can be lowered to the required level.
- The highest risk comes from an adversary with capabilities CAP2 and CAP4 which acts as a legitimate entity and knows the TI shared-secret of a responder. Consequently he can infer the identifier values $\mathrm{ID_{Ai}}$ of other entities and flood the responder with invalid FV values requiring each one a MAC verification. Even under these hypothesis, the probability to guess the master key of a third party remains low.

5. Performance evaluation

5.1. VoIP-based implementation and atomic evaluation

We implemented the DRCEP protocol as a VoIP-based prototype based on SIP signalling, and evaluated the core functions on a Dell390 Precision server based on an Intel Core2Duo and with 2GB of RAM. The DRCEP data exchange between the protocol entities (e.g., A, S and B) is implemented through standard SIP dialogue. The AuthQ message corresponds to a SIP-OPTIONS query and the AuthR message corresponds to a SIP-200OK response. For these two messages, the DRCEP protocol fields are encapsulated into the SIP message body in text format. The original SIP-INVITE request generated by the caller corresponds to the OM message and the final AM message encapsulates the encrypted OM message plus the expected filtering value and the MAC.

For the atomic performance evaluation, we implemented all the protocol entities (A, B and S) within a single physical host, we measured the response time of each function for a large set of packets (up to 10^8 packets) and we noted down the atomic average processing time. The obtained results are shown in Table 5 and correspond to the protocol optimal performances because: 1) All the entities are implemented within a single host thus removing all network delays and OS network processing; 2) The protocol information is passed directly between processes within simplified data structures. This removes the SIP stack coding and decoding times; 3) We did not implemented the SIP signalling processing in itself. We focused on the DRCEP core functions of: packet filtering, decoding and session key establishment.

From a functional point of view, Table 5 distinguishes the low level functions from the higher level functions. Low level functions consist in TRID computation, transaction window iteration and message filtering; they can be implemented close to the hardware. Higher level functions consist in computing the DRCEP transaction keys, the $\mathrm{TM_E}$ field, in ciphering and deciphering the protected information Ident and OM; they can be implemented at the application level.

The filtering function updates the transaction window every δt seconds, verifies the filtering value FV of each message, and validates the corresponding MAC of the message. If both the filtering value and the MAC are correct, the message is eventually passed to the remainder processes at the application level. We recall (cf. section 3.5 and equation 30) that the filtering function shall validate the filtering value of each message in the following order: verification of $\mathrm{P_1}$, verification of $\mathrm{P_2} \oplus \mathrm{ID_E}$, verification of the $\mathrm{P_3}$ part (requiring a MAC computation). Therefore, we assume the existence of four different types of received messages:

- Type 1: message with an invalid $\mathrm{P_1}$ part in FV.
- Type 2: message with a valid $\mathrm{P_1}$ but an invalid $\mathrm{P_2}$ part in FV.
- Type 3: message with both valid $\mathrm{P_1}$ and $\mathrm{P_2}$ but an invalid $\mathrm{P_3}$ part in FV.

- Type 4: message with a valid filtering value FV.

Observe that, while the verification of messages of types 1, 2, or 3 will always fail, the validity of messages of type 4 depends on the validity of the MAC on the whole message. Since we assume different average lengths for each DRCEP message (respectively 100, 100 and 1000 bytes for AuthQ, AuthR and AM) the resulting processing time for messages of type 4 is either $10s$ or $20s$. Depending on the MAC check, messages of type 4 will eventually be discarded or passed to the upper modules to complete their processing. As shown in Table 5, the remaining processing times at the application level (AuthQ , AuthR and AM generation) are constant because they concern valid messages which have already been filtered.

Notice that the transaction window iteration time is the same for all the entities ($2s$) and is equal to the computation time of the transaction identifier TRID. This comes from the application of the same equations (1, 12) which consist in hashing a pre-image of the same length.

Entity	**Function**	**Processing time**
A		
	- $\mathrm{TRID}(\mathrm{TI_S})$ computation	2 s
	- AuthQ generation based on received SIP-INVITE	15 s
S		
	- Transaction window iteration	2 s
	- AuthQ filtering when Type 1	13 ns
	- AuthQ filtering when Type 2	35 ns
	- AuthQ filtering when Type 3	4,7 s
	- AuthQ filtering when Type 4	10 s
	- $\mathrm{TRID}'(\mathrm{TI_S})$ computation	2 s
	- AuthR generation based on received AuthQ	34 s
A		
	- Transaction window iteration	2 s
	- AuthR filtering when Type 1	13 ns
	- AuthR filtering when Type 2	35 ns
	- AuthR filtering when Type 3	4,7 s
	- AuthR filtering when Type 4	10 s
	- AM generation based on received AuthR	50 s
B		
	- Transaction window iteration	2 s
	- AM filtering when Type 1	13 ns
	- AM filtering when Type 2	35 ns
	- AM filtering when Type 3	4,7 s
	- AM filtering when Type 4	20 s
	- AM processing	40 s

Table 5: Processing time per message and entity

5.2. Extended evaluation over a complete network

The previous atomic performance evaluation is now extended to a complete network simulating a real deployment scenario. The goal of this extended evaluation is to measure the delays of the call set-up between entities (hereinafter denoted as CSD, Call Set-up Delay), as well as the loss rates and the filling of waiting queues (FIFO buffers) under the presence of attackers applying the adversary models presented in Section 4. This evaluation is performed with OMNET++ [38], an extremely powerful simulation framework. In the sequel, we present the details of the network, the characteristics of the attackers, and the obtained results.

5.2.1. Network architecture and traffic sources

Figure 2 depicts the network of the extended performance evaluation. It is composed of several VoIP domains and we consider only unidirectional communications from the sources to the destination network B. All those calls received by B are coming from three different VoIP domains (denoted as A1, A2 and A3). Within each of these domains, several VoIP terminals are generating calls among which some are targeted to endpoints located at network B. We only simulate the call establishment process, i.e., the reception by network B of AM messages conveying the SIP-INVITE request. Implicitly, the remaining VoIP signalling and media processing is performed by specific entities which are not included in the simulation scope.

Domains A1 and A2 do not hold a shared secret with B. Hence they require a third party server S to get connected to B. For each call generated by one of these domains, we measure the performance of the complete chain $\mathrm{net}\mathrm{A} \rightarrow \mathrm{A} \rightarrow \mathrm{S} \rightarrow \mathrm{A} \rightarrow \mathrm{B} \rightarrow \mathrm{net}\mathrm{B}$ with messages AuthQ, AuthR and AM. Within each of these domains, there are three different traffic sources. Each source is characterised by its cardinality and by its respective call rate towards network B: residential home-networks (denoted as R), medium size networks (denoted as PME) and large size networks (denoted as GE). We assume that the calls generation pattern follows a Poisson distribution characterised by its parameter λ.

On the contrary, domain A3 has a shared secret with B and, consequently, can directly send messages AM to B. This is equivalent to considering that entities A and S in domain A3 are grouped into a single entity that we denote as AS. For each call generated by domain A3, we measure the performance of the optimised chain $\mathrm{net}\mathrm{A} \rightarrow \mathrm{A} \rightarrow \mathrm{B} \rightarrow \mathrm{net}\mathrm{B}$. Within domain A3, we do not distinguish between the various traffic sources.

Table 6 shows the number of clients per domain, grouped by type of sources, with their respective average call rate towards B, the corresponding λ value and the average traffic impact on S and B (still assuming an average 100 bytes length for AuthQ and AuthR messages and 1000 bytes for AM). As shown in this table, domain A2 generates ten times more traffic than domain A1, domain A3 generates an aggregated traffic similar to A2 but with no impact on S entity since A3 has a shared secret with B. Finally, it is important to observe that the total 1,113,000 clients do generate, on average, one call towards B every $0,53ms$ which results in 1,866 calls per second and consequently 687 million calls per hour.

For each sever (A, S or B), we simulate separately the ingress and egress message flows. On the egress side, there are the

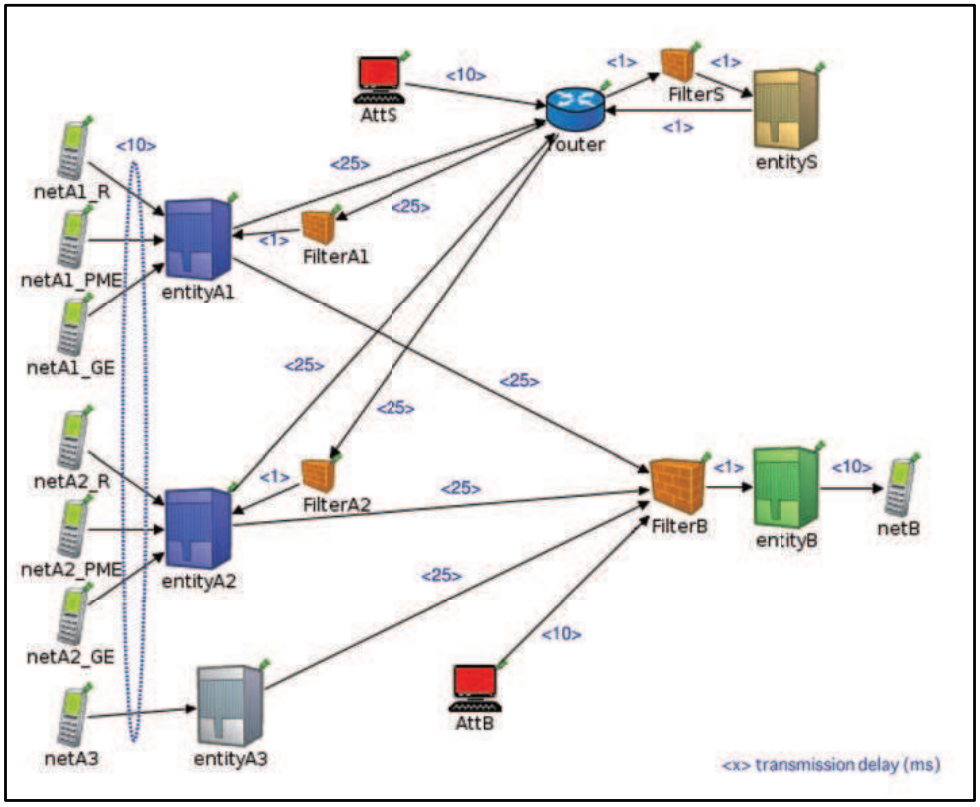

Figure 2: Network simulated for the extended performance evaluation

Domain	Network	Number of clients	Call rate	λ (ms)	Traffic on S (MBytes/s)	Traffic on B) (MBytes/s)
A1						
	netA1-R	50,000	3	24	0.004	0.041
	netA1-PME	2,500	40	36	0.003	0.028
	netA1-GE	500	140	51	0.002	0.019
	total	53,000	-	11	0.008	0.088
A2						
	netA2-R	500,000	3	2.4	0.041	0.416
	netA2-PME	25,000	40	3.6	0.027	0.277
	netA2-GE	5,000	140	5.1	0.019	0.196
	total	530,000	-	1.1	0.088	0.888
A3						
	netA3	530,000	-	1.1	-	0.888
Total		1,113,000	-	0.53	0.096	1.866

Table 6: Corresponding parameters of every traffic source

TRID computation function and the message generation function. On the ingress side there are the filtering function and the message processing function. Entity B only has an ingress chain since it does not respond to the received AM messages. As shown in Figure 2, the filtering function is simulated as a separate entity, in order to isolate its dynamic processing time which depends on the type of message received. The filtering module manages a transaction window, whose parameters are the following:

- δt: time precision period for the iteration of the transaction window, set to $10ms$.
- $\mathrm{k_{min}}$: lower bound of the transaction window, set to -500. Thus the filtering module authorises valid transactions received up to $5s$ later than the local time.
- $\mathrm{k_{max}}$: upper bound of the transaction window, set to 300. Thus the filtering module authorises valid transactions received up to $3s$ earlier than the local time.
- Δ: transaction window size: $\Delta = \mathrm{k_{max}} - \mathrm{k_{min}} + 1 = 801$.
- Θ: iteration period for the base transaction index BTI, set to $3,600s$.

Within each entity, the processing modules (both ingress and egress) are characterised by the response times indicated in Table 5, i.e., $15s$ for the generation of messages AuthQ, $34s$ for the generation of messages AuthR, $50s$ for the generation of messages AM, and $40s$ for processing the AM message on B side.

Each module includes a FIFO buffer where messages are queued following their order of arrival. Since the filtering module is, by far, the most requested module, we settle the size of its FIFO queue to 10,000 messages. On the other hand, since the processing modules (i.e., emission and reception of messages) have a response time significantly lower than the legitimate traffic inter-arrival time (some s vs. some ms), we set the size of their FIFO queues to 1,000 messages.

Finally, let us observe that the simulated network includes network delays of several ms (they appear as <delay> on Figure 2). These delays also count for the processing time which is not related to the DRCEP protocol itself: traversal of network and OS stacks, coding and parsing of SIP messages, other application delays not captured in Table 5. More specifically, the transmission delay in the MAN area (i.e., inside domains A and B) is set to $10ms$; the one in the WAN area (i.e., between entities A, S and B) is set to $25ms$ and finally, the one in the LAN area (i.e., between the filtering modules and the servers or router) is set to $1ms$.

5.2.2. Attackers and malicious traffic

Within the network there exist two groups of attackers: a first group (denoted AttS) that target their attacks against entity S thus using AuthQ messages; and a second group (denoted AttB) that target entity B thus using AM messages. We assume that both groups hold enough network and processing resources to send up to 10^6 messages per second against their respective targets. Considering the respective lengths of AuthQ and AM messages, this results in a flooding of up to 100MByte/s towards entity S and up to 1000MByte/s towards entity B.

In addition, we assume attackers can send the four different types of messages identified in Section 5.1 (i.e., messages with a valid filtering value FV, a partially valid one or a completely invalid one). While messages of types 1, 2, 3 with invalid filtering values may be generated randomly, those of type 4 with a valid filtering values imply attackers having CAP1 capability (cf. Table 3, Section 4). In the latter case, the attackers do therefore eavesdrop the legitimate traffic on the communication channel and capture valid identifiers. They can then launch (D)DoS attacks against S or B by replaying valid traffic or craft tampered messages with a valid identifier, in which case a valid MAC is required for the message to be accepted (cf. Attack A5 in Section 4.3.1).

Assuming attackers can generate such messages at high rates (up to 10^6 messages per second) means attackers are very powerful. These theoretical hypothesis are used to evaluate the DRCEP protocol under the worst case scenario. In practise, we believe that only those malicious messages denoted as type 1 are likely to appear in the network. The likelihood of messages denoted as type 2 and 3 is quite low because they require guessing respectively 32 and 64 bits of the expected filtering value. Finally, messages denoted as type 4, even if they could be generated by attackers eavesdropping the network, can be filtered with the following method: the receiving entity (S or B) adds a 2-bits counter per sender and per acceptable transaction identifier (cf. equations 11 and 12). Each time a valid filtering value is received, the related counter for the purported sender is incremented (from 0 to 3). Hence, if a valid filtering value is eavesdropped by an attacker, it can be replayed at most 2 times before being marked as invalid. Assuming the receiver transaction window contains 1,000 acceptable transaction identifiers and there are 100 possible senders, then this mechanism requires a memory of only 25,000 bytes.

Finally, we believe that attacks will combine malicious packets from various types, with packets of types 1 and 2 being the most prevalent. Hence, we define the notion of harmfulness level and, for each level, we define a different proportion of each packet type. The higher the proportion of malicious packets of type 4, the higher the harmfulness level. The proposed values are given in Table 6 and are used for simulation purpose only. Once again, it should be highlighted that these values correspond to the worst case scenario for the protocol: only packets of type 1 are likely to appear in practise, or packets of type 4 but they can be efficiently filtered.

5.3. Simulation results

We abstracted and simulated the proposed network using OMNET++ [38]. For every simulation, we measured the following output parameters:

- Number of legitimate calls generated by each source (i.e., number of SIP-INVITE requests sent by each sub-network from A1 to A3).

Harmfulness level	Packet type 1	Packet type 2	Packet type 3	Packet type 4
1	50%	50%	0%	0%
2	42%	43%	10%	5%
3	35%	35%	20%	10%
4	25%	25%	35%	15%

Table 7: Distribution of packet types per harmfulness level

- Number of legitimate calls being established (i.e., number of AM messages received at network B). From these first two parameters we infer the call loss ratio.
- Average and maximal lengths of the FIFO queues in S and B filtering modules.
- Average and maximal values of the CSD (Call Set-up Delay). The CSD is defined as the time between the SIP-INVITE is sent by one of the source sub-network and its reception by the called sub-network (netB). Thus, the Call Set-up Delay accumulates the times to traverse the various entities, FIFO queues, filtering modules and network links.

We conducted various simulations with the same legitimate traffic model (cf. Table 6) while modifying the attack traffic according to the following input parameters:

- Average malicious packet rate generated by the attackers. In a first approach, we iterated this parameter from 10^3 to 10^6 packets per second. However, we did not notice a significant impact on the legitimate traffic below 10^5 packets per second. Hence, the results presented here are truncated to the $[10^5, 10^6]$ range which means the IAT (Inter Arrival Time) varies between $1s$ and $10s$.
- Harmfulness level: as explained below, because of the atomic performance results given in Table 5, malicious packets of type 4 will have a greater performance impact than those of type 1 (by a factor of around 1000). We analyse this impact by modifying the harmfulness level and thus the proportion of packets of each type (cf. Table 7).

As a consequence, we collected 40 measurements for each observed output parameter (i.e., 10 values for IAT $\times$ 4 values for the harmfulness level). The results are plotted in the Figures 3–6 below, showing respectively: the average FIFO length for filtering module S, the maximal FIFO length for filtering module S, the average FIFO length for filtering module B, the maximal FIFO length for filtering module B, the average Call Set-up Delay (CSD), the maximal Call Set-up Delay and the call loss ratio. In each figure, the IAT parameter corresponds to the abscissa and each harmfulness level corresponds to a distinct curve. Prior to analysing these results, it should be stated that the simulations were repeated more than 50 times without showing any significant deviation in the observed results.

When the harmfulness level is equal to 1 (which corresponds to a distribution of packets types of respectively 50%, 50%, 0%, 0%), we observe no call loss up to 10^6 packets/s attack rate. This result can be explained by the atomic performance results of Table 5 and the traffic characteristics: the inter arrival time of the aggregated legitimate traffic is equal to $530s$ whereas the maximal processing time for this traffic is less than $190s$. On the other hand, the inter arrival time of the malicious packets reaches $1s$ but these packets are filtered in either $13ns$ or $35ns$ (depending on whether the P_1 part of the filtering value is valid or not). For the same reasons, the average FIFO lengths remain below 1 packet and their maximal lengths are below 100 packets. The maximal CSD (Call Set-up Delay) remains constant at $100ms$; this delay corresponds to the sum of the delays on the transmission links for the longest path ($\mathrm{netA} \rightarrow \mathrm{A} \rightarrow \mathrm{S} \rightarrow \mathrm{A} \rightarrow \mathrm{B} \rightarrow \mathrm{netB}$): $10+25+3+25+1+25+1+10 = 100ms$. It should be noted that the processing time for the longest path ($150s$) is negligible compared to delay induced by transmission links. For the shortest path ($\mathrm{netA} \rightarrow \mathrm{A} \rightarrow \mathrm{B} \rightarrow \mathrm{netB}$), the transmission delay is equal to $10 + 25 + 1 + 10 = 46ms$ and the processing time is less than $130s$. Calls from domains A1 and A2 follow the longest path, calls from domain A3 follow the shortest path, and since the respective traffic weights are 1, 10, 10 (cf. Table 6), the theoretical average CSD is equal to $(1 \times 100 + 10 \times 100 + 10 \times 46)/(1 + 10 + 10) = 74,2ms$. This value matches the ones observed in Figures 7–9 (because the processing and filtering times are negligible compared to the delay of transmission links).

When the harmfulness level is equal to 2 (which corresponds to a distribution of packets types of respectively 42%, 43%,

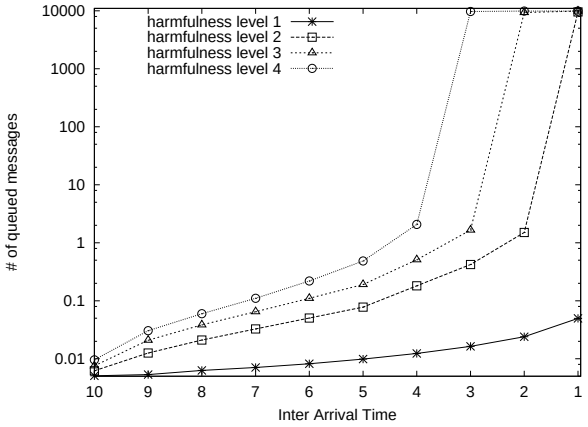

Figure 3: Average size FIFO filtering module S

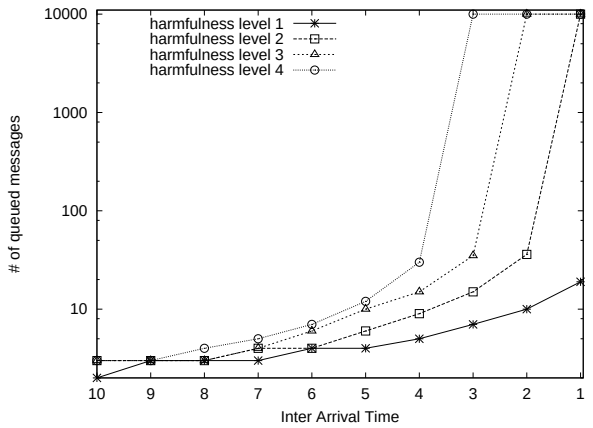

Figure 4: Maximal size FIFO filtering module S

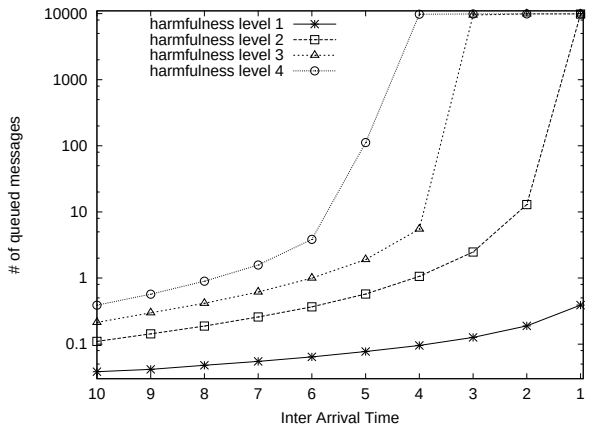

Figure 5: Average size FIFO filtering module B

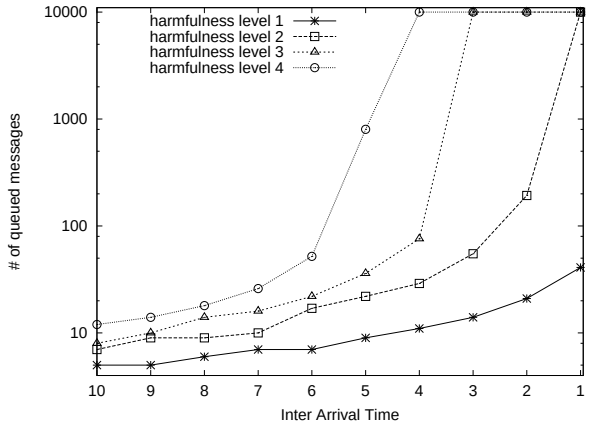

Figure 6: Maximal size FIFO filtering module B

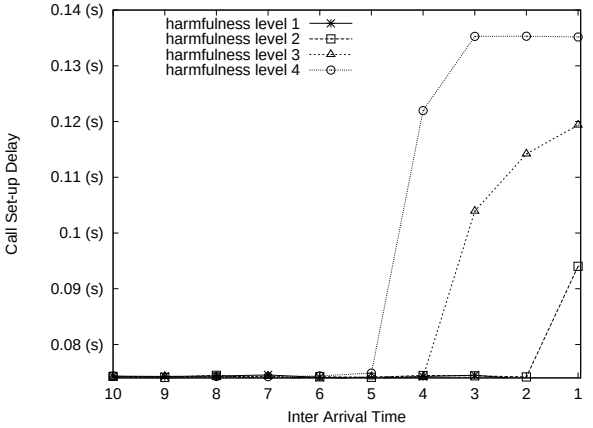

Figure 7: Average time, in (s)econds, for the establishment of calls

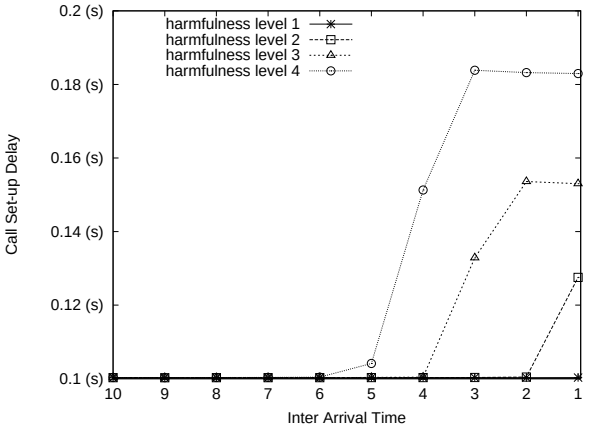

Figure 8: Maximal time, in (s)econds, for the establishment of calls

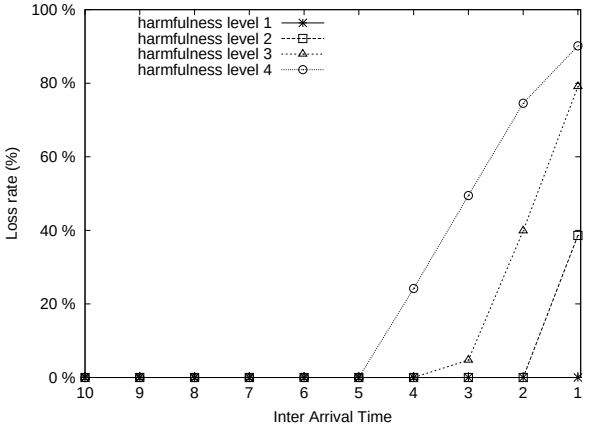

Figure 9: Loss rate at the destination network

10%, 5%), there is no call loss up to 5.10^5 packets/s attack rate and the CSD values remain unchanged, although the FIFO lengths have increased. When the attack rate reaches 10^6 packets/s, the S and B filtering FIFOs are overloaded which results in a significant loss (38%) of legitimate calls. The FIFO saturation is due to the malicious packets of type 4 (with a valid FV value): these packets are filtered in $20s$ on the B side, but since they count for 5% of the attack traffic, their inter arrival time is equal to $20s$ when the attack rate is 10^6 packets/s. In addition to that, malicious packets of type 3 require $4,7s$ to be filtered whereas their inter arrival time is equal to $10s$. This congestion in the filtering modules impacts the legitimate traffic.

When the harmfulness level is equal to 3 (which corresponds to a distribution of packets types of respectively 35%, 35%, 20%, 10%), the congestion appears at a lower rate (3.10^5 attack packets/s) resulting in legitimate calls loss and increased CSD values. This is due to the higher percentage of packets of types 3 and 4 in the attack traffic. The worst case is obtained when the harmfulness level reaches 4 (which corresponds to a distribution of packets types of respectively 25%, 25%, 35%, 15%). In that case, 24% of the legitimate calls are lost when the attack rate reaches $2,5.10^5$ packets/s. It should be noted that, even with a harmfulness level of 4, there is no impact on the legitimate traffic up to an attack rate of 2.10^5 packets/s. Besides all, we believe that such attack conditions remain theoretical and that the percentage of type 3 and type 4 malicious packets will be much lower in practice. Even if attackers where able to generate such attack conditions at the considered rates, we proposed in Section 5.2.2 an efficient counter-measure to filter malicious packets of type 4.

6. Conclusion

We have presented a cryptographic protocol that handles authentication and key agreement. The protocol aims at guaranteeing secure Voice over IP call establishment between interconnection proxies of different domains. The core base of the protocol relies on the use of transactions. A transaction is defined as the set of operations and data required to send authenticated messages from a sender to a responder. Each message can be seen as a stand-alone data exchange. This can be used, for instance, as the *preamble* of a secure session between sender and responder. Additional features of the protocol include the protection to denial of service attacks and the achievement of perfect forward secrecy. Moreover, the management of transaction synchronisation loss is guaranteed by an implicit time synchronisation mechanism. This mechanism successfully handles the use of a single transaction window in the general (inter-domain) context with multiple senders and responders. The security of the core functions of the protocol have also been evaluated. The security evaluation has been presented by dividing misbehaving entities in different adversary models, based on their capacities. The specific boundaries to bypass the security properties of the protocol have also been presented. The results show that the protocol is highly robust in terms of sample attack scenarios that could affect the security of the protocol. We have finally presented empirical measures about the atomic performance of the protocol and established a simulation model for a complete network. The simulation results prove the validity of our work, and its robustness in terms of attacks against the availability of the service.

Acknowledgements

The entire work of Patrick Battistello was produced during his contract with Orange Labs, in parallel with his doctoral studies at Institut Télécom. Orange Labs remains responsible for the technical contributions presented in this paper. The authors would like to thank to Henri Gilbert, Frédéric Cuppens, and Nora Cuppens-Boulahia for fruitful discussions on the topic of this paper. Joaquin Garcia-Alfaro graciously acknowledges support received from the Spanish Ministry of Science and Education (TSI2007-65406-C03-03 E-AEGIS, CONSOLIDER INGENIO 2010 CSD2007-0004 ARES and TIN2011-27076-C03-02 CO-PRIVACY grants).

References

[1] 3GPP. IMS Functional Architecture. 3GPP TR33.828, May 2009.

[2] H. Abdelnur, R. State, I. Chrisment, and C. Popi. Assessing the Security of VoIP Services. *10th IFIP/IEEE Symposium on Integrated Management (IM'07)*, IEEE, Pages 373–382, May 2007.

[3] W. Aiello, S.M. Bellovin, M. Blaze, J. Ioannidis, O. Reingold, R. Canetti, and A.D. Keromytis. Efficient, DoS-resistant, secure key exchange for internet protocols. *9th ACM conference on Computer and Communications Security (CCS 2002)*, ACM, Pages 48–58, 2002.

[4] F. Andreasen, M. Baugher, and D. Wing. Session Description Protocol (SDP) Security Descriptions for Media Streams. IETF RFC 4568 (Proposed Standard), July 2006.

[5] J. Arkko, E. Carrara, F. Lindholm, M. Naslund, and K. Norrman. MIKEY: Multimedia Internet KEYing. IETF RFC 3830 (Proposed Standard), August 2004.

[6] R. Barbieri, D. Bruschi, and E.Rosti. Voice over IPsec: Analysis and Solutions. *18th Annual Computer Security Applications Conference (ACSAC 2002)*, IEEE, Pages 261–270, December 2002.

[7] P. Battistello. Inter-domain and DoS-resistant call establishment protocol (IDDR-CEP): work in progress *4th International Conference Principles, Systems and Applications of IP Telecommunications (IPTComm 2010)*, ACM, Pages 22–31, August 2010.

[8] M. Baugher, D. McGrew, M. Naslund, E. Carrara, and K. Norrman. The Secure Real-time Transport Protocol (SRTP). IETF RFC 3711 (Proposed Standard), March 2004.

[9] E. A. Blake. Network Security: VoIP Security on Data Network – A Guide. *4th annual conference on Information security curriculum development (InfoSecCD '07)*, ACM, pages 1–7, 2007.

[10] D. Boneh and G. Durfee. Cryptanalysis of RSA with private key d less than $N^{0.292}$. *IEEE Transactions on Information Theory*, 46(4):1339–1349, 2000.

[11] C. Boyd and A. Mathuria. *Protocols for authentication and key establishment*. Book, Springer-Verlag, 2003.

[12] C. Castelluccia, E. Mykletun, and G. Tsudik. Improving secure server performance by re-balancing ssl/tls handshakes. *2006 ACM Symposium on Information, Computer and Communications Security (ASIACCS '06)*, ACM, Pages 26–34, 2006.

[13] C. Coarfa, P. Druschel, and D. S. Wallach. Performance Analysis of TLS Web Servers. *ACM Transactions on Computer Systems (TOCS)*, 24(1):39–69, 2006.

[14] W. Conner and K. Nahrstedt. Protecting SIP proxy servers from ringing-based denial-of-service attacks. *10th IEEE International Symposium on Multimedia, 2008. ISM 2008*, Pages 340–347, December 2008.

[15] A. Dandalis and V. K. Prasanna. An adaptive cryptographic engine for internet protocol security architectures. *ACM Trans. Des. Autom. Electron. Syst.*, 9(3):333–353, 2004.

[16] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. IETF RFC 5246 (Proposed Standard), August 2008.

[17] P. Faltstrom and M. Mealling. The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM). IETF RFC 3761 (Proposed Standard), April 2004.

[18] J. Floroiu and D. Sisalem. A comparative analysis of the security aspects of the multimedia key exchange protocols. *3rd International Conference on Principles, Systems and Applications of IP Telecommunications (IPTComm '09)*, ACM, Pages 1–10, 2009.

[19] S. E. Griffin and C. C. Rackley. Vishing. *5th Annual Conference on Information Security Curriculum Development (InfoSecCD '08)*, ACM, Pages 33–35, 2008.

[20] M. Handley, V. Jacobson, and C. Perkins. SDP: Session Description Protocol. IETF RFC 4566 (Proposed Standard), July 2006.

[21] J. Hill. The storm ahead: how calea will turn VoIP on its head. *3rd annual conference on Information security curriculum development (InfoSecCD '06)*, ACM, Pages 147–150, 2006.

[22] S. Hirose and K. Matsuura. Enhancing the resistance of a provably secure key agreement protocol to a denial-of-service attack. *2nd International Conference Information and Communication Security (ICICS'99)*, Springer-Verlag, Pages 169–182, 1999.

[23] C. Jennings and J. Fischl. Certificate Management Service for the Session Initiation Protocol (SIP). RFC 6072 (Proposed Standard), February 2011.

[24] S. Kent and K. Seo. Security Architecture for the Internet Protocol. IETF RFC 4301 (Proposed Standard), December 2005.

[25] A. D. Keromytis. A survey of voice over IP security research. *5th International Conference on Information Systems Security (ICISS '09)*, Springer-Verlag, Pages 1–17, 2009.

[26] M. Luo, T. Peng, and C. Leckie. CPU-based DoS Attacks against SIP servers. *IEEE Network Operations and Management Symposium, NOMS 2008*, IEEE, Pages 41 –48, April 2008.

[27] D. McGrew and E. Rescorla. Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP). RFC 5764 (Proposed Standard), May 2010.

[28] B. Mathieu, Y. Gourhant, and Q. Loudier. Spit mitigation by a network level anti spit entity. *Third annual security workshop (VSW'06)*. ACM, 2006.

[29] J. Mattsson and T. Tian. MIKEY-TICKET: An Additional Mode of Key Distribution in Multimedia Internet KEYing (MIKEY). IETF draft-mattsson-mikey-ticket-00, October 2009.

[30] A. Menezes, P. V. Oorschot, and S. Vanstone. Handbook of Applied Cryptography. CRC Press, August 2001.

[31] M. B. Nassar, R. State, and O. Festor. Intrusion detection mechanisms for voip applications. *CoRR*, abs/cs/0610109, 2006.

[32] C. Neuman, T. Yu, S. Hartman, and K. Raeburn. The Kerberos Network Authentication Service (V5). IETF RFC 4120 (Proposed Standard), July 2005.

[33] S. Niccolini, E. Chen, J. Seedorf, and H. Scholz. SPEERMINT Security Threats and Suggested Countermeasures. IETF draft-ietf-speermint-voipthreats-01, July 2009.

[34] NIST. Secure Hash Standard. FIPS PUB 180-2, August 2002.

[35] NIST. The Keyed-Hash Message Authentication Code (HMAC). FIPS PUB 198, March 2002.

[36] NIST. Advanced Encryption Standard (AES). FIPS PUB 197, November 2001.

[37] K. Ono and H. Schulzrinne. Have I met you before?: using cross-media relations to reduce SPIT. *3rd International Conference on Principles, Systems and Applications of IP Telecommunications (IPTComm 2009)*, ACM, Pages 1–7, 2009.

[38] Omnet++. Extensible, modular, component-based C++ simulation library and framework. http://www.omnetpp.org/, 2011.

[39] J. Peterson and C. Jennings. Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP). IETF RFC 4474 (Proposed Standard), August 2006.

[40] M. Z. Rafique, M. A. Akbar, and M. Farooq. Evaluating DoS attacks against SIP-based VoIP systems. *IEEE Global Telecommunications Conference, GLOBECOM 2009*, IEEE, Pages 1–6, November 2009.

[41] B. Ramsdell and S. Turner. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification. RFC 5751 (Proposed Standard), January 2010.

[42] E. Rescorla and N. Modadugu. Datagram Transport Layer Security. IETF RFC 4347 (Proposed Standard), April 2006.

[43] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. IETF RFC 3261 (Proposed Standard), June 2002.

[44] J. Rosenberg and C. Jennings. The Session Initiation Protocol (SIP) and Spam. IETF RFC 5039 (Informational), January 2008.

[45] J. Rosenberg and C. Jennings. Verification Involving PSTN Reachability: Requirements and Architecture Overview. IETF draft-rosenberg-dispatch-vipr-overview-42, October 2010.

[46] J. Salowey, H. Zhou, P. Eronen, and H. Tschofenig. Transport Layer Security (TLS) Session Resumption without Server-Side State. RFC 5077 (Proposed Standard), January 2008.

[47] S. Salsano, L. Veltri, and D. Papalilo. SIP security issues: the SIP authentication procedure and its processing load. *IEEE Network*, 16(6):38–44, November/December 2002.

[48] H. Shacham, D. Boneh, and E. Rescorla. Client-side caching for TLS. *ACM Transactions on Information and System Security (TISSEC)*, 7(4):553–575, 2004.

[49] C. Shen, E. Nahum, H. Schulzrinne, and C. Wright. The impact of TLS on SIP server performance. *Principles, Systems and Applications of IP Telecommunications (IPTComm 2010)*, ACM, pages 59–70, August 2010.

[50] J. Smith, S. Tritilanunt, C. Boyd, and J.M. Gonzalez-Nieto. Denial-of-service resistance in key establishment. *International Journal of Wireless and Mobile Computing*, 2(1):59–71, 2007.

[51] J. Thoguluva, A. Raghunathan, and S. T. Chakradhar. Efficient software architecture for ipsec acceleration using a programmable security processor. *Conference on Design, automation and test in Europe (DATE '08)*, ACM, Pages 1148–1153, 2008.

[52] D. S. Tonesi, L. Salgarelli, and A. Tortelli. Securing the signaling plane in beyond 3G networks: analysis of performance overheads. *Security and Communication Networks*, 2009.

[53] Z. Wan, B. Zhu, R.H. Deng, F. Bao, and A.L. Ananda. DoS-resistant access control protocol with identity confidentiality for wireless networks. *2005 IEEE Wireless Communications and Networking Conference (WCNC 2005)*, IEEE, Pages 1521–1526, 2005.

[54] D. Wing. SIP E.164 Return Routability Check (RRC). IETF draft-wing-sip-e164-rrc-01, February 2008.

[55] C.H.J. Wu, C.C.A. Huang, and J.D. Irwin. Using Identity-Based Privacy-Protected Access Control Filter (IPACF) to against denial of service attacks and protect user privacy. *2007 Spring Simulation Multiconference*, ACM, Pages 362–369, 2007.

[56] Zimmermann et al. ZRTP: Media Path Key Agreement for Unicast Secure RTP. RFC 6189 (Proposed Standard), April 2011.

[57] G. Zhang, J. Pallares, Y. Rebahi, and S. Fischer-Hübner. SIP Proxies: New Reflectors in the Internet. *Communications and Multimedia Security (CMS 2010)*, Springer-Verlag, Pages 142–153, 2010.