Providing Voice Privacy as a Service over the Public Telephone Network[#*]

Mohamad Sharif and [§]Duminda Wijesekera
[§]Center for Secure Information Systems
Department of Information and Software Engineering,
George Mason University, Fairfax, VA 22030-4444.
e-mail: {msherif|dwijesek}@gmu.edu.

ABSTRACT

Existing design of the public telephone network is susceptible to eavesdropping on its voice stream. Consequently, any wire taper can listen to supposedly private conversations. In order to prevent such eavesdropping, we propose an architecture and its implementation for end-to-end voice encryption as a service available for interested subscribers. Our proposal consists of appropriately placing servers to authenticate telephone equipment and subscribers of the service, and certificate authorities that can cross-certify over telecommunication service providers. We show how these entities and necessary signaling mechanisms between them can be implemented using the transaction capabilities application layer (TCAP) of the signal system seven (SS7) protocol and the D channel of the digital subscriber line (DSL) connecting telephone equipment to the SS7 grid. Using published specifications, we show that the duration to initiate an encrypted telephone call takes about 19 seconds including user authentication under average load conditions. That makes our design and protocols comparable to existing intelligent services provided over public telephones and only four times longer than initiating a normal telephone call.

Key Phrases: Secure telephony, telecommunication security, signal system 7, voice encryption services, privacy preserving telephones.

1 INTRODUCTION

Many people use telephone services with the belief that no one other than the identified recipient at the other end is listening to their conversations. However, as things stand today, an eavesdropper can easily monitor such supposedly private telephone conversations. Thus, telephone calls need to be protected against eavesdropping. Existing security architectures in wire-line and wireless telephone infrastructures comes short of providing end-to-end voice privacy as well as authenticating subscribers. Consequently, this paper provides an architecture and corresponding protocols to provide end-to-end voice privacy at the application layer with minimum modification to existing public telephone network infrastructures.

A telephone call made using our protocols consists of the caller dialing a * key on a telephone set equipped with encryption capabilities to obtain the service, followed by punching in identity related information and the called telephone number as well as the called subscriber ID. If the both sets of the telephones and subscribers are authenticated, then the encrypted voice streams start to flow between the two telephone sets.

Voice privacy is achieved by encrypting voice signals between two end telephones using symmetric keys and a one-time encryption key. This one-time encryption key is used to prevent replay attacks. We also propose imposing an authentication protocols for telephone subscribers and telephones that are to be used for the proposed voice privacy service. Proposed authentication protocols use public key cryptography and provide authentication center(s) the assurance that the telephone set and the user at the other end of the connection are what they claim to be. We show how to integrate our proposed voice privacy service on existing public telephone grids.

[#] Patent Pending

[*] Partly supported by NSF under grant CCR-0113515 and Center for Secure Information Systems at GMU

We also show that the privacy call setup delay under normal traffic loads is 19.7 seconds, and compare the privacy call setup delay with other intelligent services call setup delays in the PSTN. Because privacy calls requires at least seven database transaction and four subscriber inputs, it takes twice as much time than a typical calling card call initiation time that requires one database transaction and 5 times that of a normal call initiation time that does not require any database transaction.

Our proposed architecture for voice privacy can be implemented at the application layer of the *Signal System 7 (SS7)* and the *ISDN* protocol model, where exiting security architectures and other advanced intelligent network (AIN) services in the wire-line and wireless network are being implemented. Proposed architecture takes advantage of information shared between telephone companies to facilitate wire-line and wireless call processing.

The rest of the paper is organized as follows. Section 2 summarizes related work that provides some form of security for wire-line and wireless telephone networks. In order to make this paper self-contained, Section 3 provides a brief overview of SS7 and ISDN specifications. Section 4 provides a detail description of the proposed voice privacy architecture. Sections 5 describe how to integrate the proposed voice privacy protocol into the basic call model of the AIN, and detailed implementation of our protocols over existing protocols (namely over TCAP and the D-channel of the ISDN protocol stack). Section 6 describes end-to-end voice privacy call processing in PSTN. Section 7 shows the correctness of proposed protocols and section 8 shows their performance characteristics. Section 9 show the detail processing required of voice privacy messages by all relevant entities connected to SS7 in order for voice privacy protocols to work as intended. Finally, section 10 concludes the paper. Appendix A and B describes in details voice privacy protocols parameters and the messages that used to implement voice privacy design in PSTN, respectively. Appendix C describes the pseudo code for the components connected to SS7 in order for voice privacy protocols to work as proposed.

2 RELATED WORK

Telephone services have been improving from Plain old Telephone Services (POTS) to current day advance intelligent network (AIN) services. Thus far, PTSN does not provide protection against unauthorized eavesdropping. However, there are systems that provide protection from eavesdropping, but most of these systems address only the confidentiality part of the security services and not other security aspects such as authentication, authorization, and non-repudiation. These systems are discussed next.

2.1 Secure Telephone Unit: Third Generation (STU III)

Secure telephones widely used in the intelligence community, commonly known as secure telephone unit third generation (STU III), was developed by the National Security Agency (NSA) in 1987 [8]. These secure telephones are design to work only as dedicated pairs through public telephone network infrastructure and use predetermined symmetric keys to encrypt voice messages. These keys are physically stored in the handheld telephone unit. In addition, these keys are not updated frequently. STU III has an in-built key management system for customizing and downloading keys. Obtaining a STU III requires NSA's permission.

2.2 SecureLogix's TeleVPN®

TeleVPN [27] system provides voice privacy between two communicating Private Branch Exchanges (PBXs) over PSTN. TeleVPN uses triple Data Encryption System (3DES) [24,31,32] to encrypt voice signals and provides similar services to Virtual Private Network (VPN) over the Internet. Because TeleVPN is implemented at the perimeter of the voice network, it does provide a true end-to-end (telephone-to-telephone) voice privacy service. More over it does not perform

subscriber and telephone authentication. TeleVPN System was not available in the market at the time of this writing.

2.3 Wireless Networks

Wireless communications are more susceptible to eavesdropping than wire-line communication, because readily available radio scanners can easily monitor radio signals [6,32]. Because wireless signals are sent over the air using insecure radio channels, eavesdroppers can not only monitor the conversation but also obtain mobile station information such as Mobile Identification numbers. This information can be used to create a clone. Due to mobile station cloning, wireless industry is loosing millions of dollars every year [12,22]. In order to address these security issues, wireless industry started to implement telephone authentication to protect against cloning and eavesdropping.

Authentication and confidentiality for wireless networks are defined in ANSI-41 (IS-41) and Global System for Mobile Communications (GSM) standards. Both IS-41 and GSM security are based on symmetric key cryptographic techniques where a secret key is shared between the mobile station and the authentication center in the network. Details of IS-41 and GSM security appear on [3,5,6,26]. Both IS-41 and GSM security addresses the issue of wireless telephone cloning, but do not offer end-to-end voice privacy or subscriber authentication except where the latter has to be provided under law in Europe.

3 SIGNALING ARCHITECTURE OF THE PSTN

PSTN signaling has been evolving since early days of manual switching. Presently, there are two kinds of signaling techniques are used in the telephone network: channel-associated (in-band) and common-channel (out-of-band) signaling. Channel-associated signaling uses the same channel to carry voice signals and signaling messages. The common-channel signaling uses separate channels or networks to deliver voice and control signals. There are two types of common-channel signaling: built-in separate channel and separate channel signaling. In built-in separate channel signaling, the signaling protocol is carried on a separate channel on the same line or a trunk that carries the voice signals. An example of built-in separate channel signaling is the D channel of the ISDN network [2,3,30]. In separate channel signaling, the signaling protocol is carried on a separate network. An example of separate channel signaling is the signaling system 7 (SS7) [2,3,30].

The signaling protocol enables systems within the telephone network to communicate with each other. There are two types of signaling protocols used in the telephone network: access (local loop) and network (inter-offices) signaling. Access signaling is the basic form of signaling and is used at the interface between the telephone set and the end office. The digital access signaling in the local loop uses the Integrated Services Digital Network (ISDN). The signaling in the PSTN and the wireless network uses the SS7 protocol.

3.1 Signaling System 7

Signaling System 7 (SS7) is an out-of-band signaling standard for the telephone network developed by ITU-TS, and defines the architecture and the protocols of its signaling network. In order to make this paper self contained, the next section provides a brief overview of SS7, but details of SS7 appear in standard references such as [2,3,23,30].

3.1.1 The Signaling Network

The signaling network is a separate network that carries the signaling messages between the SS7 components. There are three main components in PSTN: Service Switching Points (SSP), Signaling Transfer Points (STP), and Service Control Points (SCP). These components are

arranged throughout the SS7 network in such way that the network provides the maximum performance, reliability, and flexibility. SSP is a part of the local switch connecting telephone sets to the telephone network. SSP is the hub of the advance intelligent network (AIN) architecture and is responsible for processing calls that require remote database translations. SSP also provides support to back-office functions such as configuration, billing, performance, error reporting, etc. An SSP in a wireless network is referred as a mobile-switched center (MSC). STP performs routing and is responsible for routing signaling messages from its local SSP to destinations in the SS7 network. SCP is a component with access to the intelligent network (IN) database and is responsible for processing requests from SSPs and other SCPs in the network.

A simplified design of a signaling network is shown in Figure 1. Every SSP is connected to at least two STPs for reliability in a design known as a mated STP pair. A mated STP pair in one service provider is inter-connected with other mated STP pair of other service providers. In addition, mated STP pair in one service provider can support SSPs of other service providers. Each STP is connected to at least one SCP and is responsible for routing the messages in the SS7 network. One SCP may connect to several STPs.

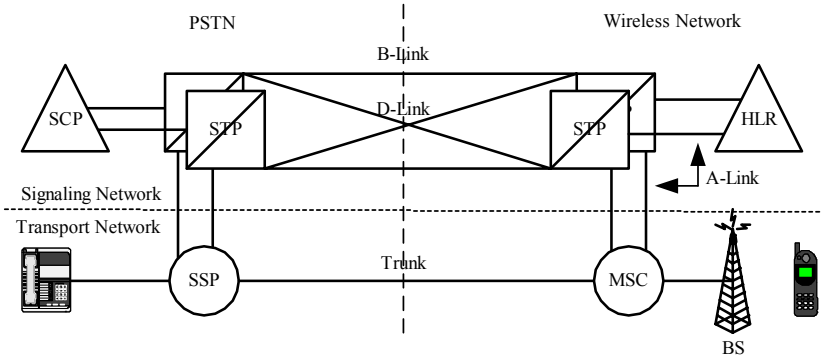

Figure 1. The Transport and Signaling Network in Public Telephone Network

Each database provides service specific applications such as subscriber profiles, mobile station profiles, 800 number translations, security, calling card and other services [2]. Some of the commonly used databases are *line information database* (LIDB), *home location register* (HLR), *authentication*, and c*all management service database* (CMSDB). LIDB contains information such as the subscriber calling features, billing, calling card, etc. HLR contains information such as mobile subscriber features, home location, status, etc. Authentication contains information such as PINs, user identification, authentication schemes, etc. CMSDB contains information such as call routing, call management, call status, etc.

3.1.2 The Signaling Protocol Stack

The SS7 protocol stack has a four-level hierarchy. This arrangement is similar to the four-layer hierarchy of the TCP/IP protocol stack of the Internet [2,3,30] as shown in Figure 2. Following is a short description of the SS7 protocol stacks:

- **The Message Transfer Part (MTP)** provides reliable transport of signaling messages across the SS7 network. MTP consists of the following three levels:
  MTP 1: Signaling Data Link corresponds to the lower half of network layer of the TCP/IP protocol model, and defines physical characteristics of the signaling links.
  MTP 2: Signaling Link corresponds to the upper half of the network layer of the TCP/IP protocol model, and provides reliable transfer of signaling messages between signaling points.

MTP 3: Signaling Network corresponds to the lower half of Internet layer of the TCP/IP protocol model, and provides routing related functionality.

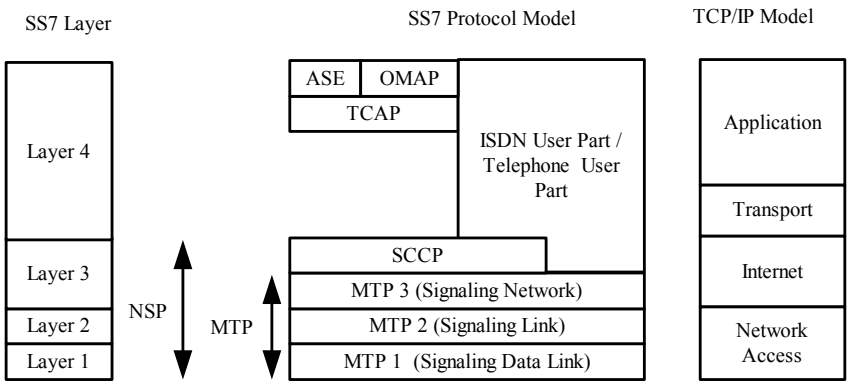

Figure 2. The SS7 Protocol Stack compared with the TCP/IP Protocol Stack

- **Signaling Connection Control Part (SCCP)** corresponds to the upper half of the Internet layer of the TCP/IP protocol model and provides functions to transfer non-trunk related messages such as database accesses. SCCP enhances the MTP 3 services by providing connectionless and connection-oriented classes of services. The combination of MTP and SCCP is referred as the Network Services Part (NSP) of the SS7.
- **Telephone User Part (TUP) / ISDN User Part (ISUP)** is a protocol that provides call control, trunk maintenance, and call setup resources to the telephone network and the ISDN. North America uses ISUP and the rest of the world uses the TUP.
- **Transaction Capabilities Application Part (TCAP)** is a non-circuit related protocol that allows one signaling point to invoke an operation in another signaling point and then use its response. It operates at the application layer of SS7 protocol stack, but is independent from AIN applications or services. TCAP is divided into *component and transaction parts*. The component part (CP) provides a standard interface to the AIN applications and transaction part (TP) provides a connection-oriented transaction for CP. TCAP offers four types of services; request-response, respond only if the request is succeed, respond only if the request fails, and responses-less service. Because the proposed voice privacy service is implemented in TCAP, we provide the TCAP message structure and protocols in section 3.1.3.
- **The Operation, Maintenance, and administration part (OMAP)** is an application of TCAP providing operations to monitor, coordinate, and control the telephone network.
- **The Application Service Element (ASE) or AIN Services** is an application of the TCAP and providing user specific services such as *mobile application part* (MAP).

3.1.3 Overview of the TCAP Message Structure

TCAP messages transports non-circuit related messages within the SS7 network and allows one network entity to invoke an operation in another one and then use its response. It operates at the application layer of the SS7 protocol stack, but is independent of the AIN applications or services. TCAP message consists of *component and transaction parts*. The component part (CP) provides a standard interface to the AIN applications and transaction part (TP) provides a connection-oriented transaction for CP [3,16].

Transport Part Message

The transaction part provides an end-to-end connection for the component part over the connectionless services of SCCP, and can transport one or more component messages at one time.

- BEGIN starts the transaction, and is sent by the sender.
- CONTINUE continues the transaction, and can be sent by either party in need of more information to process he request.
- END ends the transaction, and is sent by the receiver.
- UNIDIRECTIONAL specifies a one-way transaction or the sender does not expect a response from the receiver, and is sent by either the sender or the receiver.
- ABORT says that the transaction has to be terminated due to some problem and is sent by the receiver.

Component Part Message

Component part provides peer-to-peer communication of the proposed architecture, and it contains one request or a response to a request

- INVOKE specifies a request operation, and is sent by the sender.
- RETURN RESULT (RR) specifies a response to an INVOKE indicating a successful operation, and is sent by the receiver.
- RETURN ERROR indicates an unsuccessful operation of an INVOKE, and is sent by the receiver.
- REJECT indicates that the component message can not be processed due to some syntactic problem, and is sent by the receiver.

Typical TCAP Message Flow

When a SSP receives a service request from an entities on the network, it sends a query request to initiate a transaction with the service control point (SCP) or other entities such as an authentication center (AC), etc, to receive instruction on how to handle the call. When a SSP initiates a transaction with an AC, SSP suspends that call processing and moves into a wait state. When AC receives the request , it determines the appropriate action and sends its response to the SSP. There might be several rounds of message exchange between a SSP and an AC before the transaction is completed. Figure 3 shows an example of TCAP message sequences in a transaction that involves a SSP and an AC. The following describes TCAP message sequences:

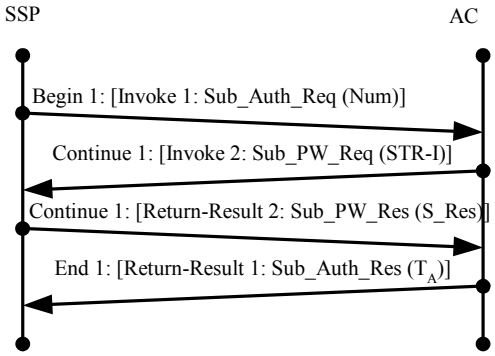

Figure 3. Typical TCAP Message Flow

- SSP initiates transaction 1 by sending a *BEGIN* message with *INVOKE 1* message *BEGIN 1: {INVOKE 1:[Sub_Auth_Req()]}* to AC. *BEGIN* is a Transaction Part of the TCAP message and indicates the start of the transaction. *INVOKE* is a Component Part of the TCAP message, and is a request message. Number 1 in *BEGIN* transaction is a reference number that identifies the transaction. Number 1 in *INVOKE* message is a reference number that identifies the invoke message.
- When AC receives the message, it determines that it needs more information from SSP to process this *INVOKE 1*. Thus it sends to SSP a *CONTINUE* transaction *CONTINUE 1: {INVOKE 2:[Sub_PW_Req ()]}*. *CONTINUE* is a Transaction Part of the TCAP message and indicates the continuation of the transaction 1
- SSP returns *CONTINUE* message *CONTINUE 1:{RR 2[Sub_PW_ INVOKE Res()]}* with to AC as a response to *INVOKE 2* of transaction 1. *RR* 2 is a Component Part of TCAP message, and indicates a response.
- AC uses this new information to respond to *INVOKE 1* from SSP, and send *END 1: {RR 1: [Sub_Auth_Res()]}* message with *RR 1* to SSP as response to *INVOKE 1* of transaction 1. *END* is a Transaction Part of TCAP message and indicates the end of the *BEGIN* transaction.

3.2 Integrated Service Digital Network (ISDN)

ISDN was developed by ITU-T and ANSI to provide a digital interface between the customer equipment and the network for the transport of a wide range of digitized services such as voice, data, images, etc, and their control messages. The ISDN version used in North America was developed by ANSI, and has minor differences from the ITU-T [2,3,30].

3.2.1 ISDN Access Interfaces

ISDN defines two types of digital subscriber line (DSL) access interfaces: Basic Rate Interface and Primary Rate Interface. These access interfaces define the bit rate and the number of available channels.

- **Basic Rate Interface (BRI)** is designed to meet the needs of the residential subscriber or small business with a total bit rate of 192 kbps including 48kbps for overhead functions. It provides two full duplex 64kbps B-Channels and one full duplex 16kbps D-channel over a twisted pair of wires. The B-Channel is used to carry digitized voice or data signals and the D-Channel is used to carry signaling messages to control and manage the associated B-channel. These signaling messages are defined at the network layer of the ISDN protocol stack is discussed shortly.
- **Primary Rate Interface (PRI)** is designed to meet the needs of medium or large business and operates at the bit rates of up to T1 (1.544 Mbps) or E1 (2.048 Mbps). The B-Channel and the D-Channel operate at the same rate of 64Kbps. At the T1 rate, channels 1 to 23 are used as B-Channels and channel 24 is used as a D-Channel (23B + D). At the E1 rate, channels 1 to 15 and channels 17 to 30 are used as B-channels and channel 16 is used as a D-Channel (30B + D).

3.2.2 The ISDN Protocol

The ISDN protocol was developed to support signaling messages between the end user and the network as well as between end users. However, ISDN defines only the network access protocol consisting of three layers. It is equivalent to the lower three layers of the SS7 protocol stack as shown in Figure 4. Because these layers address only the user-to-network interface (UNI) signaling messages and not the user-to-user information, ISDN allows end users to define their own upper layer protocols. ISDN signaling messages are carried in the D-Channel of the DSL. A short description of the ISDN protocol stack is as follows:

- **Physical Layer or Layer 1:** defines the physical characteristic between end user equipment and the local ISDN exchange for both BRI and PRI, and is equivalent to the MTP 1 of the SS7 protocol stack.
- **Link Access Protocol for D-Channel (LAPD) or Layer 2:** defines the logical connection between end user equipment and the local ISDN exchange, and provides reliable transfer of frames over the DSL. LAPD is equivalent to MPT 2 and 3 layers of the SS7 protocol stack.
- **Network Layer or Layer 3:** defines the signaling messages that are exchanged over the D-Channel to setup B-Channel connection, and these message are generally referred as Q.931 messages. Q.931 provide a reliable transport circuit and non-circuit messages over the D-Channel of the DSL. USER INFOrmation (USER INFO) message is used to transport non-circuit messages. The network layer is equivalent to the ISUP layer of the SS7.

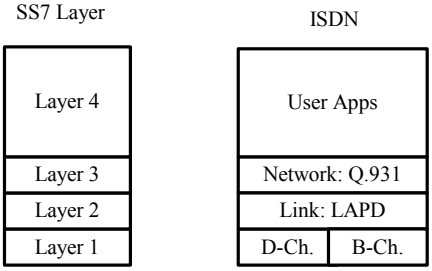

Figure 4. The ISDN Protocol Stack compare with the SS7 Protocol Stack

3.3 Typical Call Processing in PSTN

This section describes the message exchange between two ISDN telephones in the PSTN as shown in Figure 5. S_1 uses T_1 to initiate call request and T_1 is connected to SSP_1 via local loop (DSL). SSP_1 is connected to SSP_2 via a trunk and SSP_2 is connected to T_2 via a local loop. S_2 uses T_2 to accept the call. ISDN protocol is used between T and SSP, and SS7 protocol is used between SSPs. The following steps describe the end-to-end connection setup between T_1 and T_2.

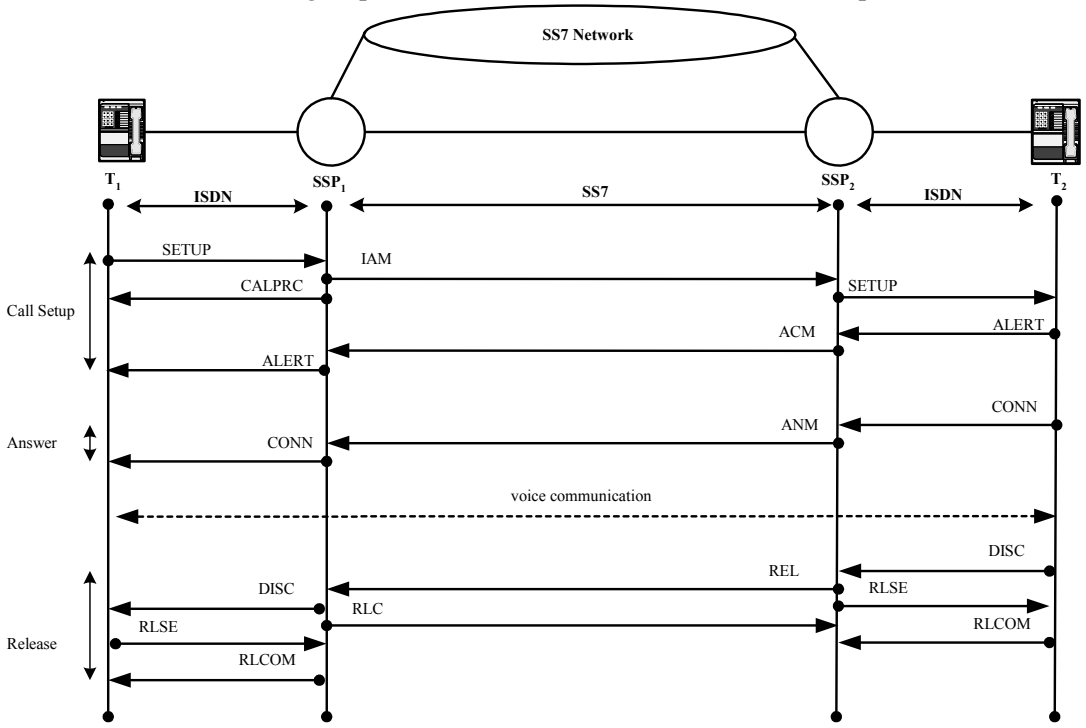

Figure 5. Typical Call setup processing in PSTN

1. S_1 enter T_2's number. T_1 allocates the D-channel on the DSL between T_1 and SSP_1 and sends SETUP message to SSP_1 to inform the connection request over the D-channel on the DSL between T_1 and SSP_1.
2. In response, SSP_1 allocates the B-channel on the DSL and sends a CALPRC message via the D-Channel to T_1 in order to inform that the B-Channel allocation and connection setup has started. In addition, SSP_1 allocates a channel on the trunk between SSP_1 and SSP_2, and informs SSP_2 of the trunk allocation request by sending an Initial Address Message (IAM) to SSP_2 via the SS7 signaling network.
3. T_1 start to listen to the B-Channel.
4. Once SSP_2 receives the IAM message from SSP_1, it allocates D and B Channels on the DSL between SSP_2 and T_2, and inform T_2 of these allocations by sending a SETUP message to T_2 over the D-Channel.
5. In response, T_2 alerts S_2 and sends an ALERT message to SSP_2 over the D-Channel.
6. When SSP_2 receives the ALERT message from T_2, it sends an Address Complete Message (ACM) to SSP_1 via SS7 network. ACM informs SSP_1 that the request trunk is reserved and S_2 has been alerted.
7. In response, SSP_1 sends an ALERT message to T_1 over the D-channel.
8. When T_1 receives the ALERT message, it connects the ringing tone source to the B-channel and S_1 hears the ringing tone.
9. When S_2 answers T_2, T_2 sends a Connection (CONN) message to SSP_2 over the D-Channel. CONN informs SSP_2 that S_2 answered the call.
10. In response, SSP_2 sends an Answer Message (ANM) to SSP_1 via the SS7 network.
11. When SSP_1 receives ANM message from SSP_2, it sends CONN message to T_1 over the D-Channel.
12. In response, T_1 removes the ringing tone source from the B-Channel and allows the conversation to start between S_1 and S_2 over the B-channel.
13. Assume S_2 hangs up T_2 first. Once S_2 hangs up, T_2 sends a Disconnect (DISC) message to SSP_2 over the D-channel.
14. In response, SSP_2 send a Release (REL) message to SSP_1 via the SS7 network and sends a Release (RLSE) to T_2 over the D-Channel to inform the release of the D and B Channels at SSP_2
15. When T_2 receives the RLSE message, it sends a Release Complete (RLCOM) to SSP_2 over the D-Channel to confirm the release of the D and B Channel at T_2.
16. When SSP_1 receives a REL message, it sends a Release Complete (RLC) to SSP_2 via the SS7 network to confirm the release of trunk between SSP_1 and SSP_2, and sends a DISC message to T_1 over the D-Channel.
17. When T_1 receives the DISC message, it sends a RLSE message to SSP_1 over the D-Channel to inform the release of the D and B Channels at T_1.
18. In response, SSP_1 sends a RLCOM message to T_1 over the D-Channel to confirm the release of the D and B Channels at SSP_1.

4 PROPOSED VOICE PRIVACY ARCHITECTURE

We now discuss the proposed voice privacy architecture. First, we describe the components of the architecture. Next, we describe the device and subscriber authentication protocols. Finally, we describe voice encryption key generation and distribution protocols.

The proposed voice privacy architecture consists of *certificate authorities (CA)*, *authentication centers (AC)* and telephone sets with cryptographic capabilities on top of the existing public telephone network infrastructure as shown in Figure 6. Telephones are connected to SSPs through DSL, and SSPs are connected to an AC, through mated STP pairs. In addition,

CAs are connected to mated STP pairs. Signaling between the telephone and SSP uses the ISDN protocol, and signaling within the network uses the SS7 signaling protocol.

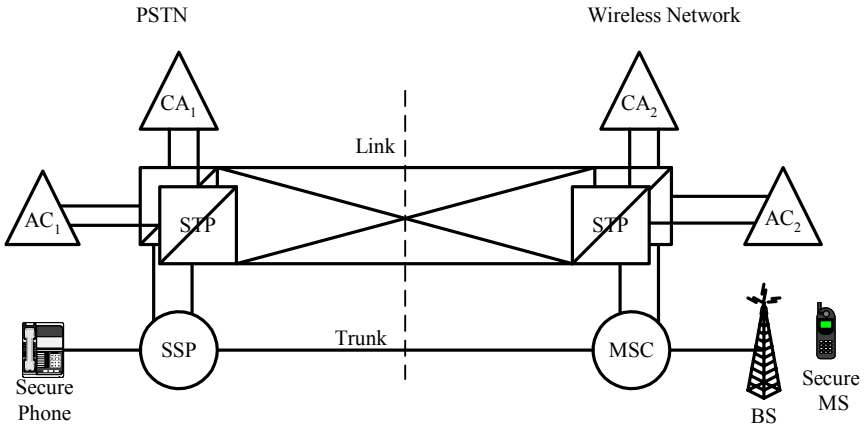

Figure 6. Proposed Voice Privacy Architecture

The CA and AC are to be implemented at the *application service element (ASE)* of the SS7 protocol model. CA is responsible for generating public/private keys, creating digital certificate of public keys, and storing the digital certificates in the publicly available database as well as interfacing with other CA's in the public telephone network. In addition, it is responsible for maintaining the *certificate revocation list (CRL)* containing the list of compromised or expired keys. A digital certificate is a record that binds the telephone's public key to the telephone number and is signed by the CA of the privacy service provider. Proposed voice privacy service assumes that every privacy service provider establishes a CA and each CA establishes trust relationship with other CA's in other privacy service providers. This trust relationship is referred as *cross-certification.*

An AC is responsible for generating and distributing keys, and authenticating telephones and subscribers. It is also responsible for maintaining the authentication database that contains subscriber and telephone profiles. Subscriber's profile contains the subscriber identification with corresponding encrypted password as well as other information related to the subscriber. Telephone's profile contains the line number, telephone's digital certificate, device number, and other information related to the telephone. AC interfaces with other AC's in the public telephone network to service roaming subscribers who request privacy service outside of their home location.

The privacy telephone is an intelligent device that is capable of voice encryption and key management and is an ISDN compatible device that is capable of communicating with other devices in the network with or without subscriber intervention. We describe authentication protocols in detail next.

4.1 Authentication Protocols

The proposed voice privacy protocol uses public key cryptography to authenticate telephones and subscribers in the voice privacy service. The CA of the service provider generates the public/private key pair and the digital certificate of the AC. In addition, the CA stores the digital certificate of the AC in the CA's database that is publicly available and stores AC's public/private key pair in a secure file.

Whenever a subscriber signs up for the privacy services, the CA generates the public/private key pair and a digital certificate of the telephone. It stores the digital certificate of the telephone

in the telephone's profile as well as in the CA's database. Then the CA installs the telephone's public/private key pair and AC's public key in the telephone set. Storing the digital certificate of the telephone in the CA's database will allow the CA to revoke the certificate in the case of service suspension and/or security problems such as the disclosure of telephone's private key. This feature makes our protocol more secure than other security architectures described in section 2, because they do not provide this capability.

The telephone's profile contains other information in addition to public/private key pairs such as the telephone location, subscriber, billing information etc. It can be stored in the *line information database (LIDB)* for PSTN telephones and in the *home location register (HLR)* for the wireless phones. The telephone's public/private key pair is *unknown* to the subscriber. When a private key of the telephone is compromised, the CA will revoke the digital certificate of that telephone and store it in the CRL as well as in the telephone's profile. Then it generates a new key pair, as described shortly.

When the subscriber signs up for the privacy service, s/he selects an ID and password pair. They are stored in the subscriber's profile in the authentication database located at the AC. *Any* subscriber of the privacy service can use *any* privacy telephone to get the proposed voice privacy service. There are two types of authentication taking place in the proposed service: *device authentication and subscriber authentication*. Device authentication is used to authenticate the telephone set that is signed up for the privacy service, and the subscriber authentication is used to authenticate the subscriber who is requesting or accepting an encrypted telephone messages.

4.1.1 Protocol 1: Device Authentication

Any AC can initiate the device authentication and is transparent to the subscriber. As shown in Figure 7, following steps describe protocol 1. The □ symbols indicates the end of protocols and proofs.

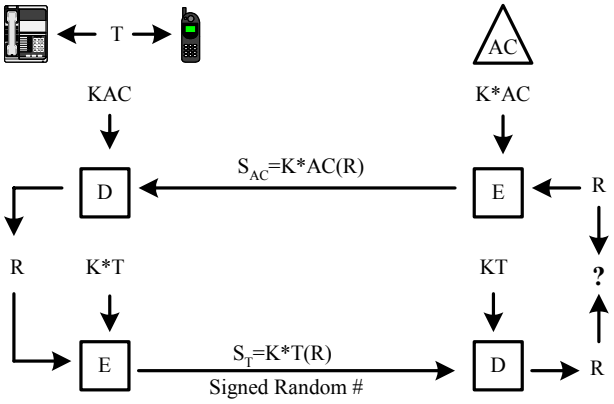

Figure 7. The Device Authentication Protocol

1. The AC generates a random number R, and it encrypts R with AC's private key (K^*AC) to obtain signed R (S_{AC}), which is used as the digital signature $S_{AC}=K^*AC(R)$ of the AC, and sends S_{AC} to the telephone over the D-channel of the DSL.
2. When the telephone set receives S_{AC}, it decrypts S_{AC} with AC's public key KAC to recover $R=KAC\ (S_{AC})=KAC\ (K^*AC\ (R))$, encrypts the received R with its private key (K*T). $S_T=K^*T(R)$ and sends S_T to the AC over the D-channel of the DSL.
3. When AC receives S_T it decrypt S_T with the telephone's public key KT and compares the received R with the original $R = KT\ (S_T)= KT\ (K^*T\ (R))$ that AC sent to T. If the two R's are the same, AC allows the telephone to receive the voice privacy service and else denies.□

4.1.2 Protocol 2: Subscriber Authentication

When the caller requests a privacy service, the AC initiates the protocol 2 as shown in Figure 8, consisting of the following steps:

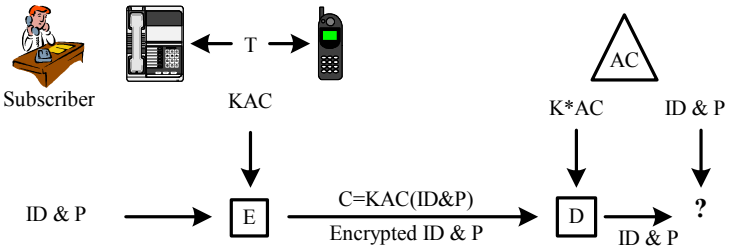

Figure 8. The Subscriber Authentication Protocol

1. In response to a subscriber request for a secure connection the *interactive voice response (IVR)* at the end office requests and the subscriber to enter the subscriber's ID and password (ID&P) pair, encrypts ID&P with AC's public key (say KAC) to obtain *C=KAC(ID&P)* and transmits C over the B-channel of the DSL.
2. When AC receives C, it decrypts C with AC's private key K*AC to recover *ID&P =K*AC (C)=K*AC[KAC(ID&P)],* and verifies the ID&P received with the ID&P in the authentication database. If verified to be correct, the calling subscriber is allowed to receive the privacy service, and is denied otherwise. Once the calling subscriber is authenticated, the AC authenticates the called subscriber using the same process. □

4.2 Protocol 3: Key Management and Voice Encryption

Voice signals are encrypted between the two end telephones using a symmetric key when both telephone sets and subscribers are authenticated as shown in Figure 9 and described in protocol 3.

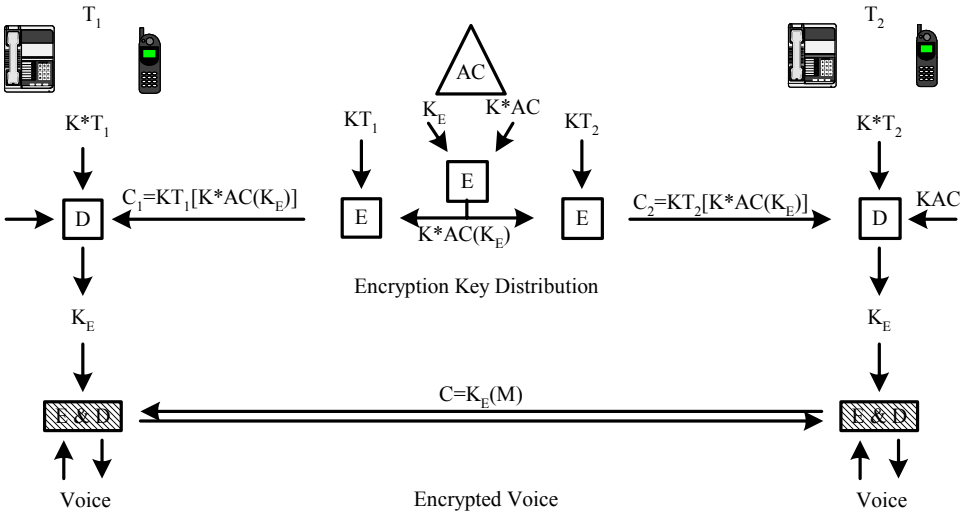

Figure 9. Key Management and Voice Encryption Protocol

1. The AC generates the encryption key K_E, encrypts K_E with its private key (K*AC) to obtain *K*AC(K_E)*. Then it encrypts *K*AC(K_E)* with T_i's public key KT_i to obtain *KT_i[K*AC (K_E)]* say C_i, and sends C_i to T_i over the D-channel of the DSL for i=1,2.

2. When T_i receives the encrypted C_i it first decrypts C_i with its private key and then with the public key of AC to recover K_E.
3. Both telephones use K_E to encrypt/decrypt the voice signals. □

This key K_E is only valid during the call in progress, and is destroyed once the call is terminated. If K_E is compromised during the call and the call is on digital subscriber line, the AC will generate a new K_E and sends to the telephone over the control channel. Once the telephone receives the new key $K_{E,}$ it destroys the compromised key and uses the new key. We describe this process in section 5.3.4.

5 INTEGRATING VOICE PRIVACY INTO THE BASIC CALL MODEL

The *basic call model* (BCM) defines the process of establishing and terminating calls in the PSTN [1,2,3]. This section provides an overview of the BCM and how the voice privacy protocols can be integrated into the BCM. Next, we list new messages designed to support the proposed protocol. Then we show how these messages can be used to implement the protocols described in section 4.

5.1 The Basic Call Model

The *advance intelligent network* (AIN) component of the BCM is traditionally designed as a collection of communicating state machines implemented at the originating and terminating *service switching points* (SSPs), referred to as *Originating BCM* (OBCM) and *Terminating BCM* (TBCM) respectively. Each state in a BCM is referred as a *point in call* (PIC), and has a set of input and output parameter known as *detection points* (DP) and *triggers* [2] respectively. The detection points of a BCM are the conditions under which a state transition takes place, and associated triggers are invoked when the corresponding DP conditions are met.

A simplified BCM with the proposed voice privacy states is shown in Figure 10. States found in a BCM without our protocol are drawn using unbroken edges and additional states necessary to implement our protocols are shown in broken edges. As shown in Figure 8, proposed protocols add three new states and modify an existing state in both OBCM and TBCM. We name the three new states as *Subscriber Authentication, Privacy Processing, and Key Distribution* reflecting their utility in the overall protocol. The modified states are *Authorization Origination* in OBCM and *Authorization Termination* in TBCM. New and modified states are as fallows:

- **Subscriber authentication state:** SSP verifies if the subscriber is signed for the privacy services and is to whom s/he claims to be.
- **Privacy processing state:** The originating SSP informs the terminating SSP that the calling subscriber is approved to have a private conversation with one of its subscriber.
- **Key distribution state:** The authentication Center (AC) generates and distributes the encryption key to originating and terminating SSPs after it authenticates the called subscriber.
- **Authorization origination state:** The originating SSP verifies that the calling telephone is authorized to participate in the privacy services.
- **Authorization termination state:** The terminating SSP verifies that the calling telephone is authorized to participate in the privacy services.

When a SSP detects a service request from the subscriber or other entities in the network it sends a query request to (initiate a transaction with) the service control point (SCP) or other entities such as an authentication center (AC) etc, to receive instruction on how to handle the call. Then, the SSP suspends that call processing and moves into a wait state until it receives a response from the AC. When the AC receives the request , it determines the appropriate action and sends its response to the SSP. There may be several rounds of message exchange between a SSP and an AC before the transaction is completed. BCM uses TCAP message transfer protocol to transport non-circuit related message between entities in the SS7 network. In addition, BCM uses USER

INFO component of Q.931 protocol of the ISDN to transport user information between the ISDN telephone and the SSP over the D-Channel of the DSL.

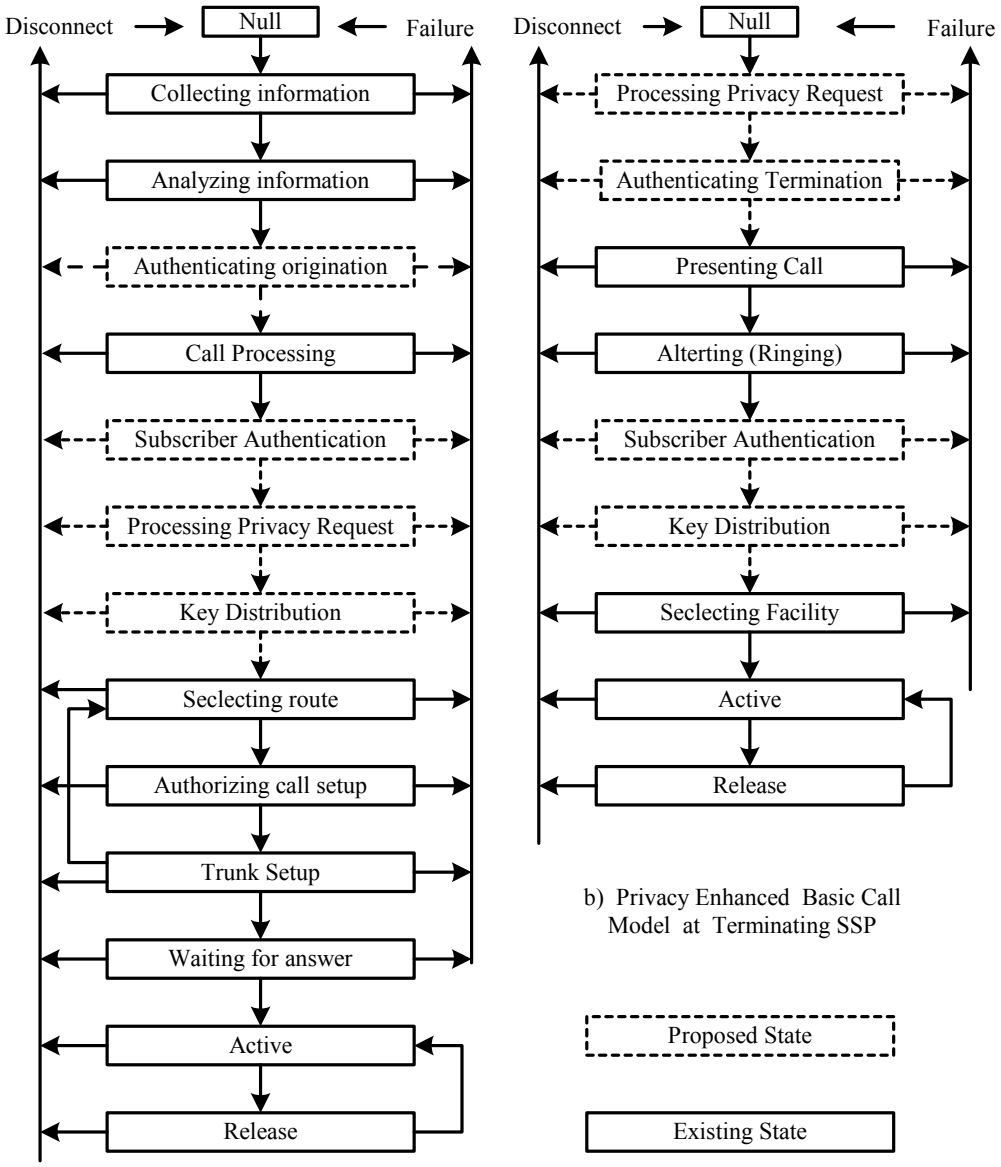

Figure 10. Basic Call Model with Proposed Voice Privacy Protocol

5.2 Voice Privacy Messages and Parameters

This section describes messages and parameters designed to implement proposed protocols. Table 1 and Table 2 provide a brief description of the voice privacy parameters and messages, respectively. Table 3 lists all of the voice privacy parameters and messages with their transport protocols. Details of the voice privacy messages and parameters are described in appendix A and B respectively.

Parameter	Parameter Name	Description
*** (Asterisk)**	Privacy Code	Telephone uses to request privacy services from the SSP.
S	Subscriber	Calling subscriber (S_1), and called subscriber (S_2).
T	Telephone	Calling telephone (T_1), and called telephone (T_2).
AC	Authentication Center	Authenticates T's and S's. Generates and distributes encryption keys.
CA	Certificate Authority	Generates public/private keys. Also, generates and stores the digital certificates in public database.
K, K*	Public (K) and Private (K*) Key pair	These keys are used to encrypt/decrypt messages between T and AC
S_ID	Subscriber Identification	S_1_ID and S_2_ID identify $S_1$and S_2, respectively.
S_PW	Subscriber Password	S_1_PW and S_2_PW authenticate S_1 and S_2, respectively
T_LN	Telephone Line Number	T_LN is 10-digit line number and identify the line directly connected between SSP and T.
T_DN	Telephone Device Number	T_DN is a unique serial number, and is permanently stored in the T. T_DN and T_LN uniquely identify T
S_1_RES	T_1 Encrypted Response	KAC[S_1_ID, S_1_PW, S_2_ID, T_1_LN, T_2_LN, T_1_DN]
S_2_RES	T_2 Encrypted Response	KAC [S_2_ID, S_2_PW, T_2_LN, T_2_DN]
t_A	Approval Timestamp	t_A determines the period which the approval ticket (T_A) is valid.
t_C	Confirmation Timestamp	t_c determines the period which the confirmation ticket (T_C) is valid.
t_E	Encryption Key timestamp	t_E determines the period which the encryption key (K_E) is valid
T_A	Approval Ticket	T_A is a signed ticket and indicates that the S_1 is approved to have private communication with S_2. T_A = K*AC [S_1_ID, S_2_ID, T_1_LN, T_2_LN, t_A],
T_C	Confirmation Ticket	T_C is a signed ticket and indicates that the S_1 and S_2 are approved to have private communication. T_C = K*AC [S_1_ID, S_2_ID T_1_LN, T_2_LN, t_C]
K_E	Voice Channel Encryption Key	K_E is used to encrypt the voice signals between T_1 and T_2.
K_{ET}	Key Distribution Ticket	K_{ET} is a signed value to assure T, that AC issued K_E. K_{ET} contains K_E and t_E. K_{ET} = K*AC[KT (K_E, t_E)]
R	Random Number	AC generates and uses R to authenticate the T.
CONT	Continue	SSP continues the privacy call set up
DISC	Disconnect	SSP plays disconnect announcement, and sends disconnect message to T.
STR-I	Send to Resource, Caller Interaction	SSP connects to Intelligent Network Services Circuit (INSC) to play Password request announcement, and to collect the digits sent by T. In addition, SSP sends the collected digits to AC

Table 1. Voice Privacy Parameters

Message	Name	Description

Sub_Auth_Req	Subscriber Authentication Request	SSP sends this message to AC to request S authentication**.**
Sub_Auth_Res	Subscriber Authentication Response	AC sends this message to SSP to respond to Sub_Auth_Req.
Sub_PW_Req	Subscriber Password Request	In response to Sub_Auth_Req, the AC sends this message to SSP to request information from S.
Sub_PW_Res	Subscriber Password Response	SSP sends this message to AC in response to Sub_PW_Req.
Pri_Call_Req	Privacy Call Request	The originating SSP sends to the terminating SSP to request privacy call setup.
Pri_Call_Res	Privacy Call Response	This message is sent by the terminating SSP to respond to Pri_Call_Req message.
Pri_Call_Approved	Privacy Call Approved	SSP sends this message to the T_1 to inform the privacy call setup is approved.
Pri_Call_Confirm	Privacy Call Confirm	SSP sends this message to Both T to inform the privacy call setup is confirmed.
Key_Dist_Req	Key Distribution	This message is sent by the AC to distributed the encryption key
Enc_Key_Req	New Encryption Key Request	T sends this message to request a new encryption key due to time out or compromise.
Enc_Key_Res	New Encryption Key Response	This message is sent by the AC to respond to Enc_Key_Res message.
New_Conf_Ticket	New Confirmation Ticket Distribution	When SSP receive an updated T_C from AC, it sends this message to the other SSP to forward the updated T_C.
Tel_Val_Req	Telephone Validation Request	SSP sends this message to the AC to request telephone authentication.
Tel_Val_Res	Telephone Validation Response	AC sends this message to SSP to respond Tel_Val_Req (TLN) message
Tel_Auth_Req	Telephone Authentication Request	AC sends this message to the telephone for authentication request.
Tel_Auth_Res	Telephone Authentication Response	telephone sends this message to the AC to respond the Tel_Auth_Req message.

Table 2. Voice Privacy Messages

Privacy Service Protocol		**Transport Protocol**		
Message	**Parameters**	**ISDN**	**SS7**	
		Q.931	**TCAP**	
			Component Part	**Transaction Part**
Sub_Auth_Req	S or T_A		INVOKE	BEGIN
Sub_Auth_Res	T_A , T_C or DISC		RR	END
Sub_PW_Req	STR-I or DISC		INVOKE	CONTINUE
Sub_PW_Res	S_1_RES or S_2_RES		RR	CONTINUE
Pri_Call_Req	T_A, T_1_LN, T_2_LN		INVOKE	BEGIN
Pri_Call_Res	T_C or DISC		RR	END
Pri_Call_Approved	T_A or DISC	USER INFO		

Pri_Call_Confirm	T_C or DISC	USER INFO		
Key_Dist_Req	K_{ET1}, K_{ET2}	USER INFO	INVOKE	BEGIN
Enc_Key_Req	T_C, KAC(T_LN, T_DN)	USER INFO	INVOKE	BEGIN
Enc_Key_Res	T_C, or DISC	USER INFO	RR	END
New_Conf_Ticket	T_C	USER INFO	INVOKE	UNIDIRECTIONAL
Tel_Val_Req	TLN		INVOKE	BEGIN
Tel_Val_Res	OK or DISC		RR	END
Tel_Auth_Req	K*AC(R)	USER INFO	INVOKE	CONTINUE
Tel_Auth_Res	K*T[KAC(T_i_LN, T_i_DN),R]; *where i = 1 or 2*	USER INFO	RR	CONTINUE

Table 3. Voice Privacy Messages and Parameters with their Transport Protocols

5.3 Implementing Proposed Protocols

In order to describe the voice privacy protocol implementation in BCM, first, we describe the environment setup and then assumptions. S_1 uses T_1 to initiate the privacy call and T_1 is connected to SSP_1 via local loop. SSP_1 is supported by AC and is connect to SSP_2 via a trunk. SSP_2 is also supported by AC and is connected to T_2 via a local loop. TCAP is used to transport the voice privacy message within the SS7 network, and the USER INFO function of the Q.931 protocol is used to transport the privacy message between the telephones (Ts) and SSPs.

5.3.1 Implementing Protocol 1

After SSP_1 receives a SETUP message from T_1, it initiates the authentication process for T_1 as shown in Figure 11. If the authentication successful, then B-channel of the DSL is setup. Otherwise, the connection is terminated. Similarly, SSP_2 initiates the authentication process for T_2 when it receives Pri_Call_Req message from SSP_1. Following steps implements protocol 1.

Let *i* be 1 or 2.

1. SSP_i initiates transaction 1 by invoking the telephone line authorization request in AC by sending *BEGIN 1: [INVOKE 1:[Tel_Val_Req(T_i_LN)]]*.
2. In response, AC generates a random number R, encrypts R with K*AC and sends the encrypted value K*ACC[R] to SSP_i with a request to invoke telephone authentication request as a continuation of transaction 1 by sending *CONTINUE 1: [INVOKE 2: [Tel_Auth_Req (K*AC[R])]]* to SSP_i.
3. In response, SSP_i reformats the request {*Tel_Auth_Req (K*AC[R])}* and sends it to T_I as *USER INFO [Tel_Auth_Req (K*AC[R])]*.
4. T_i decrypts the encrypted value by applying KAC to obtain R. (*KAC(K*AC[R]) = R*). Then, T_i encrypts (T_i_DN, Ti_LN) with KAC, encrypts the encrypted value *(KAC[TND_i,TLN_i])* and the received R with K*T_i, and sends the signed value to SSP_I by sending *USER INFO [Tel_Auth_Res{K*T_i(KAC[TND_i,TLN_i], R)}]*.
5. When SSP_i receives *[Tel_Auth_Res{K*T_i(KAC[TND_i,TLN_i], R)}]* from T_i, it reformats and forwards to AC as a response to the invoke 2 of the transaction 1 by sending
   CONTINUE 1: [RR 2:[Tel_Auth_Res{K*T_i(KAC[TND_i,TLN_i],R)}]]
6. When AC receives the response from SSP_i, it decrypts to obtain *(T_i_ND,T_i_LN)* and *R*.
   a. If the received R is the same as the original R, and the received *[T_i_ND,T_i_LN]* pair is the same as the pair in the privacy database. AC sends *END 1: [RR 1: [Tel_Val_Res (CONT)]]* to SSP_i transaction 1 and SSP_i continues to process the privacy call setup.
   b. Else, AC sends *END 1: [RR 1: [Tel_Val_Res (DISC)]]* to SSP_i as response to invoke 1 of transaction 1 terminating the call processing.□

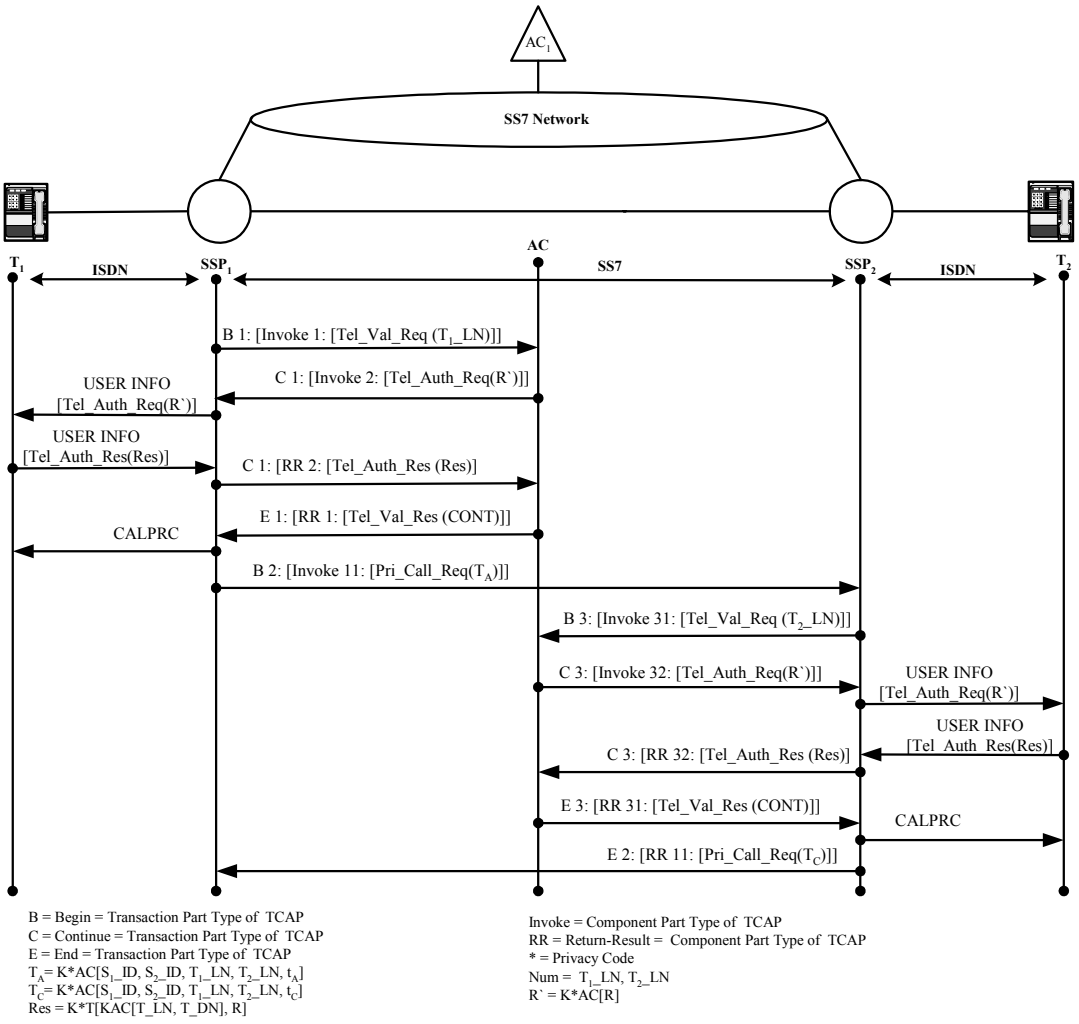

Figure 11. Implementating Protocol 1

5.3.2 Implementing Protocol 2

When T_1 is authenticated, SSP_1 initiates S_1 authentication process as shown in Figure 12. If successful, SSP_1 sends Pri_Call_Req message to SSP_2, and else terminate connection. Similarly, SSP_2 initiates S_2 authentication process when T_2 is authenticated. The following steps implement protocol 2.

Let *i* be 1,2.

1. Once T_i is authenticated, SSP_i initiates transaction 1 by sending *BEGIN 1: [INVOKE 1: [Sub_Auth_Req (S_i)]]* to AC.
2. In response, AC sends *CONTINUE 1: [INVOKE 2:[Sub_PW_Req(STR-I)]]* to SSP_i as a continuation of transaction 1 requesting the subscriber ID and password.
3. In response SSP_i sends the *CONN()* message to T_i, via the D-Channel and connects the voice announcement source (IVR) to the B-channel requesting subscribers ID and password.
4. In response subscribers respond as follows.
   a. S_1 enters S_1_ID, S_1_PW, and S_2_ID on T_1. T_1 collects S_1's entry and adds T_1_LN, T_1_LN, and T_1_DN_1, encrypts these values with KAC, and sends this encrypted value *S_1_RES = KAC [S_1_ID, S_1_PW, S_2_ID, T_1_TLN, T_2_LN, T_1_DN]* to SSP_1 via the B-channel.

b. S_2 to enter S_2_ID, and S_2_PW. T_2 collects S_2's entries and adds T_2_LN, T_2_DN, encrypts these values with KAC_2 and sends this encrypted value *S_2_RES = KAC [S_2_ID, S_2_PW, TLN_2, TDN_2]* to SSP_2.

5. When SSP_i receives S_i_RES, it sends *CONTINUE 1:[RR2:[Sub_PW_Res(S_i_RES)]]* to AC.
6. On receipt of S_i_RES, AC does the following:
   a. AC decrypts S_i_RES by applying K*AC.
   b. For S_2 authentication only, compares the received S_2_ID with the one in T_A.
      i. If the two S_2_ID*s* are not the same, sends *END1:[RR1:[Sub_Auth_Res (DISC)]]* to SSP_2 as response to invoke 1 of transaction 1, thereby ending call processing.
      ii. Else, AC goes to step c.
   c. AC authenticates S_i by comparing received *(S_i_ID, S_i_PW)* pair with those in the authentication database, and verifies that *(T_i_DN, T_i_LN)* pair is in the privacy database.
      i. If S_i is authenticated and *(T_i_DN,T_i_LN)* pair is in the privacy database, then
         1. If $S_i = S_1$, AC creates and sends *END1:[RR1:[Sub_Auth_Res (T_A)]]* to SSP_1 as response to invoke 1 of transaction 1.
         2. Else, AC creates and sends *END1:[RR1:[Sub_Auth_Res (T_C)]]* to SSP_2 as a response to invoke 1 of transaction 1.
      ii. Else, AC sends *END1:[RR1:[Sub_Auth_Res (DISC)]]* to SSP_i as response to invoke 1 of transaction 1 thereby ending call processing.□

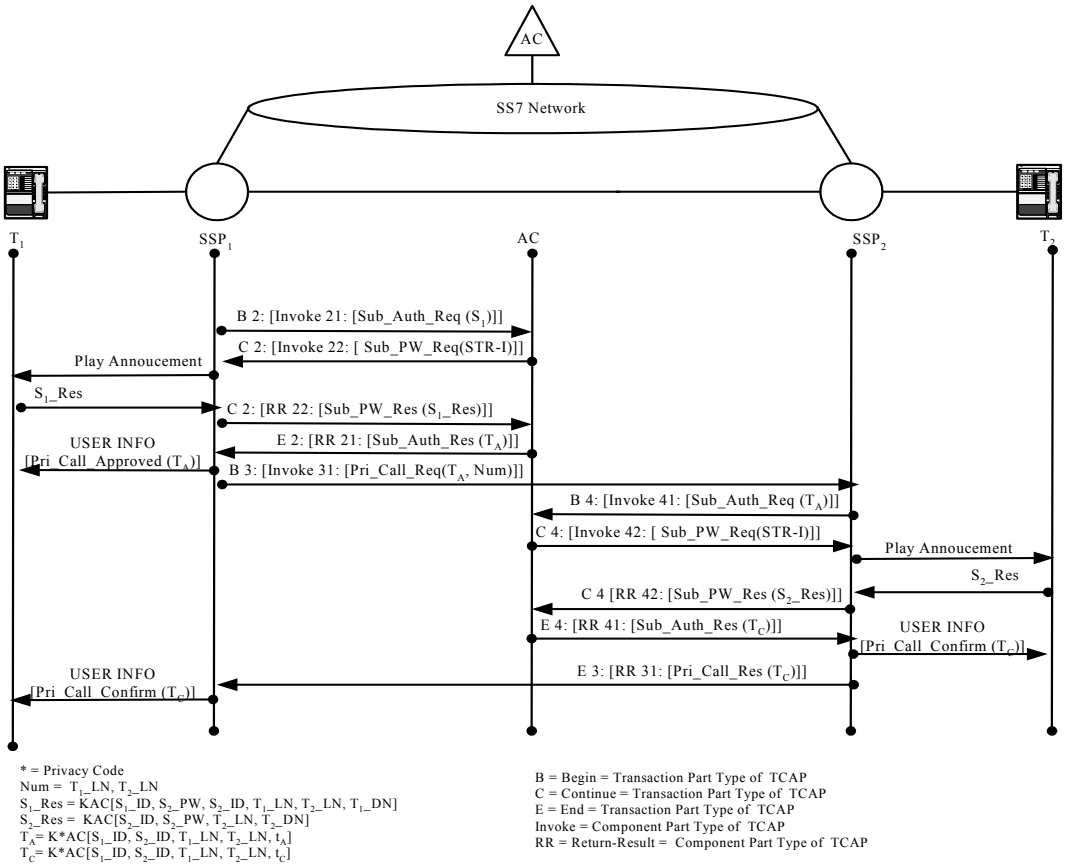

Figure 12. Implementing Protocol 2

5.3.3 Implementing Protocol 3

Once S_1 and S_2 are authenticated, AC generates and distributes K_E as shown in Figure 13 and described as follows.

Let *i* be 1,2

1. AC generates a voice encryption key K_E and a ticket t_E, encrypts (K_E, t_E) with K*AC and KT_i, respectively, to obtain $KT_i\{K*AC[K_E, t_E,]\}$ and then initiates a unidirectional transaction 11 by sending *UNIDIRECTIONAL 11: [INVOKE 22:[Key_Dist_Req (K_{ETi})]]* to SSP_i.
2. SSP_i forwards Key_Dist_Req (K_{ETi}) to T_i by sending *USER INFO [Key_Dist_Req(K_{ETi})]*.
3. When T_i receives K_{ETi} it decrypts K_{ETi} by applying K*T_i and KAC, respectively, to obtain $(K_E, t_E,)$. □

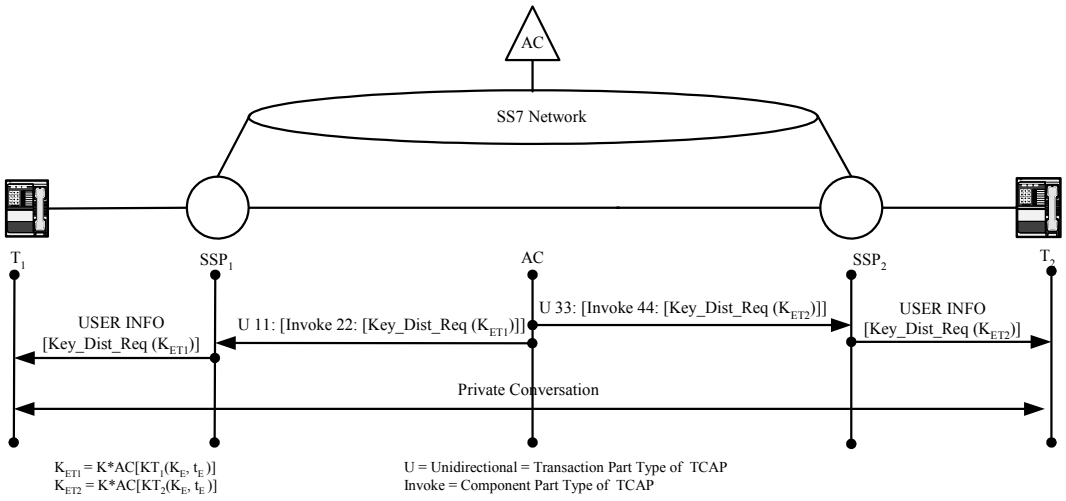

Figure 13. Implementation of Protocol 3

5.3.4 Implementation of requesting a new voice encryption key

When K_E is about to expire, is compromised, or is corrupted, one of the two Ts can request a new K_E from the AC while the connection is alive. Say S_1 initiates the request. The following protocol requests another K_E as shown in Figure 14.

1. S_1 enters the privacy code (*) and presses "ENTER" on T_1 while the connection is active.
2. T_1 interprets * to Encryption Key Request (Enc_Key_Req) and sends *USER INFO [Enc_Key_Req (T_C, KAC[T_1_LN, T_1_DN])]* to SSP_1.
3. SSP_1 initiates transaction 11 by sending *BEGIN11:[INVOKE22:[Enc_Key_Req(T_C, KAC[T_1_LN,T_1_DN])]]* to AC.
4. In response, AC decrypts the message, T_C to obtain t_C, T_1_LN and T_1_DN.
   a. If t_C has not expired and the received $(T_1$_LN, T_1_DN) pair is the same as the pair in the privacy database, AC generate and sends a new ticket T'_C with an updated of t'_C to SSP_i by sending *END 11: [RR 22: [Enc_Key_Res (T_C)]]* and goes to step 8.
   b. Else, AC sends *END 11: [RR 22: [Enc_Key_Res (DISC)]]* to SSP_1 as a refusal.
5. When SSP_1 receives *Enc_Key_Res (T_C)* from AC, it forwards T_C to T_I by sending *USER INFO [Enc_Key_Res (T_C)]* and forwards T_C to SSP_2 by sending *UNIDIRECTIONAL 33: [INVOKE 44: [New_Conf_Ticket (T_C)]*.
6. When SSP_2 receives T_C from SSP_1, it forwards *USER INFO [New_Conf_Ticket (T_C)]* to T_2
7. When T_1 and T_2 receive the new T_C, they replace the old T_C with the new one.
8. AC starts to generate and to distribute a new K_E and t_E, as described in section 5.3.3. □

Above implementations do not mention any encryption algorithm, because voice privacy protocols support most existing encryption algorithms. We are in process of determining which ones produce optimal results.

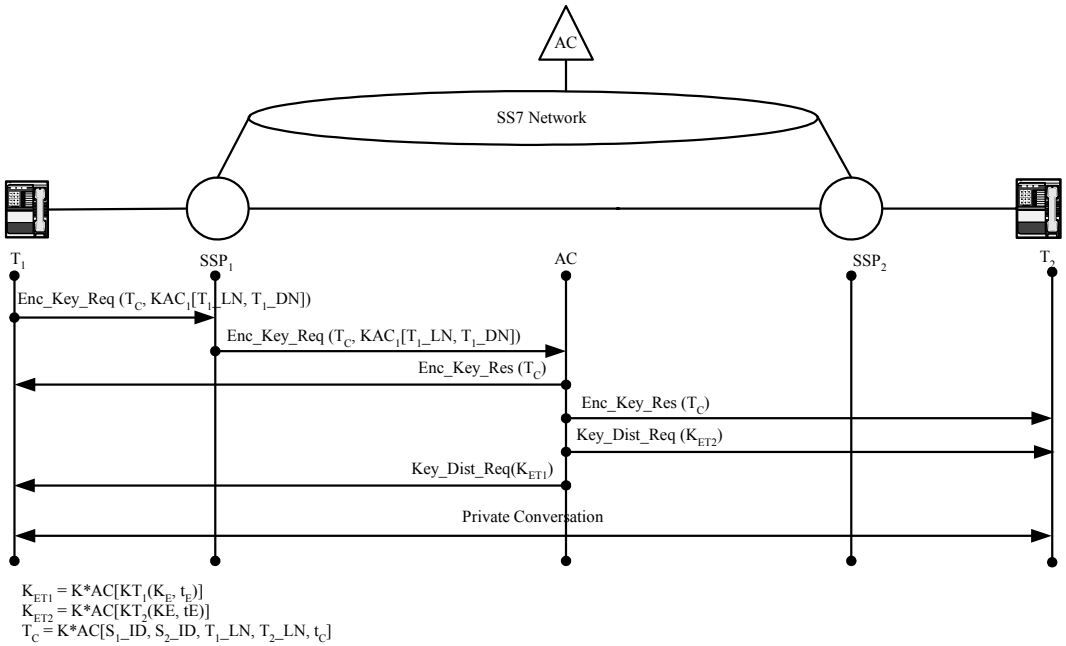

Figure 14. Implementation of Requesting a new Voice Encryption Key (K_E) Protocol

6 THE END-TO-END SETUP

This section describes the privacy call set-up between two telephones in PSTN as shown in Figure 15 with the assumptions stated in previous sections. Here we show that the service requires seven TCAP transactions ensuring the end-to-end service setup. Figure 15 shows privacy messages in broken lines.

Initiating voice privacy

1. S_1 enters TLN_2 and privacy code (*) on T_1, and in response T_1 sends the SETUP message *SETP(*,T_1_LN, T_2_LN)* to SSP_1.

Transaction 1 (T_1 Authentication)

2. In response, SSP_1 initiates transaction 1 by invoking the telephone line validation request to AC with a *BEGIN 1: {INVOKE 1: [Tel_Val_Req(T_1_LN)]}* message and ends when it receives the message *END 1: {RR 1: [Tel_Val_Res (response)];* where response is CONT or DISC. Details appear in section 5.3.1.

B-Channel setup at T_1

3. If the response is *END 1: {RR 1: [Tel_Val_Res (DISC)]}*,
   a. SSP_1 allocates the B-channel on T_1's line and sends CALPRC message to T_1 to inform the allocation of the B-channel and the connection setup is in progress, *CALPRC()*, play disconnection announcement and sends *DISC* message *DISC()* to T_1 via the D-channel. When T_1 receives *DISC* message from SSP_1, it releases D and B channels, and sends *RLSE* message to SSP_1 to inform the release of the channels.
   b. Else, SSP_1 allocates B-channel on T_1's line and sends CALPRC message to T_1 to inform the allocation of the B-channel, *CALPRC()*. When T_1 receives "*CALPRC"* from SSP_1, it starts to dial-tone ringing and waits *"CONN"* message from SSP_1

Transaction 2 (S_1 Authentication)

4. SSP_1 initiates transaction 2 by sending *BEGIN 2:{INVOKE 21: [Sub_Auth_Req (S_1)]}* to AC. AC responds to SSP_1 with *END 2: {RR 21: [Sub_Auth_Res (response)]};* where response is DISC or T_A. Details appear in section 5.3.2.

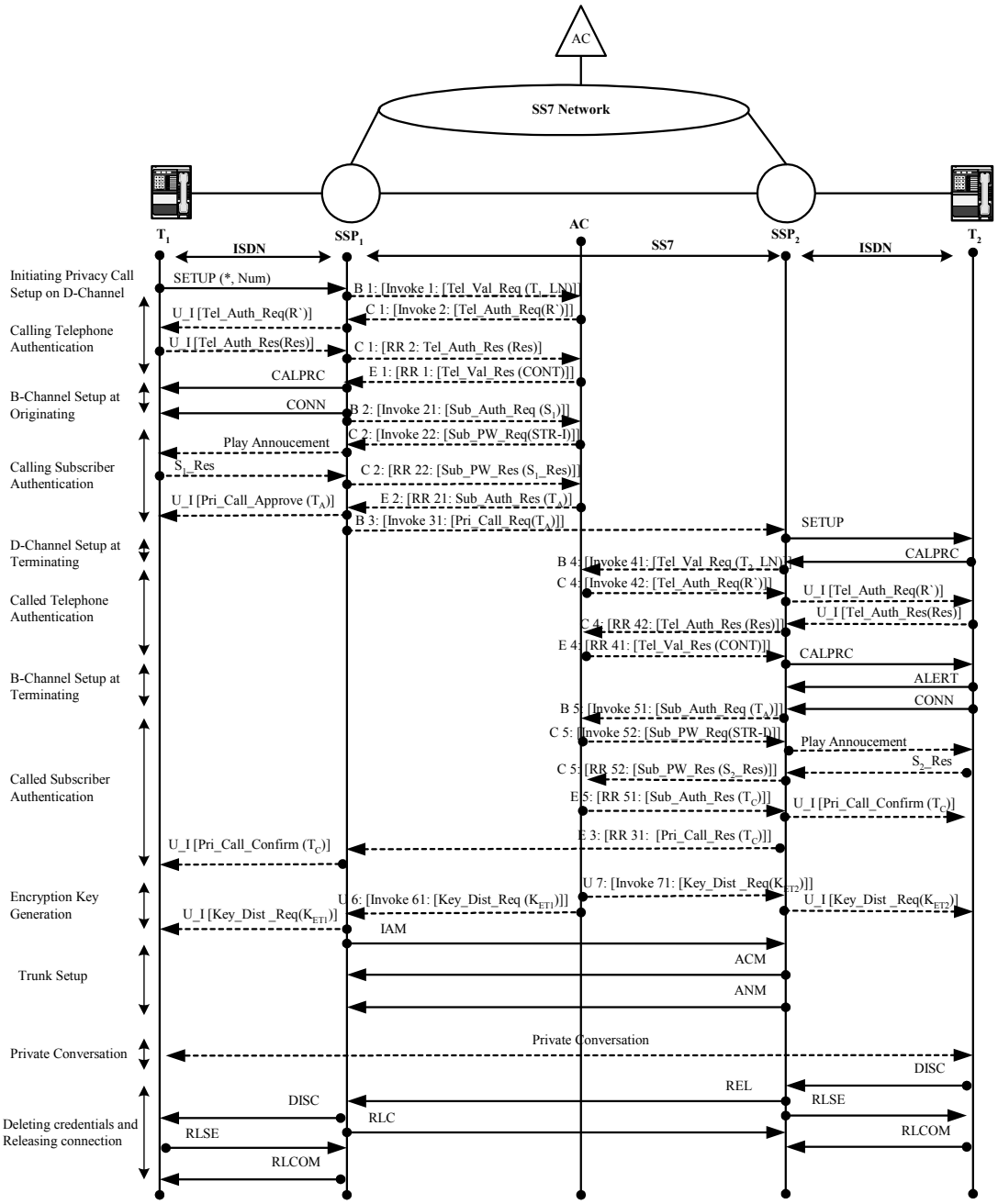

Figure 15. End-to-End Voice Privacy Call Processing

SSP_1 informs T_1 of the approval

5. If the response is "*END 2:{RR 21:[Sub_Auth_Res (DISC)]}",
   a. SSP_1 disconnects as in 3 a.

b. Else, SSP_1 plays the approved announcement to S_1, and sends *USER INFO [Pri_Call_Approve(T_A)]* to T_A to T_1 via D-Channel. T_1 saves T_A, it until T_C arrives.

Beginning of transaction 3: SSP_1 requests SSP_2 for a privacy connection

6. SSP_1 initiates transaction 3 by sending *BEGIN 3: { INVOKE 31: [Pri_Call_Req(T_A, T_1_LN, T_2_LN)]}* requesting SSP_2 for a privacy call to S_2.

D-Channel setup at T_2

7. In response, SSP_2 allocates the D-channel on T_2's line, and sends *SETUP* message to T_2 to allocate the D-channel. In response T_2 confirm s with *CALPRC()* message.

Transaction 4 (T_2 Authentication)

8. In response SSP_2 authenticates T_2 as described in step 4.

B-Channel setup at T_2

9. If authentication fails in step 8,
   a. SSP_2 disconnects T_2 as described in step 3a.
   b. Else SSP_2 allocates B-channel on T_2's line as in step 3b.
10. In response, T_2 starts to alert S_2, and when S_2 answers T_2, T_2 sends *CONN* message to SSP_2.

Transaction 5 (S_2 Authentication)

11. SSP_2 authenticates S_2 as described in step 4.

SSP_2 informs T_2 of the confirmation

12. If AC refuses to authenticate S_2,
    a. SSP_2 disconnects T_2 as described step 3a.
    b. Else, SSP_2 connects plays the privacy call confirmation announcement to S_2, and forwards *USER INFO [Pri_Call_Confirm(T_C)]* T_C to T_2 via the D-Channel. In response, T_2 saves T_C, and waits for K_{ET2}.

Ending of transaction 3: SSP2 sends the response of invoke 31 of transaction 3 to SSP_1

13. SSP_2 sends privacy call response *END3:{RR 31: [Pri_Call_Res(response)]}* where response is DISC or T_C. to SSP_1 to inform of (S_2 ,T_2) authentications, ending transaction 3.

SSP_1 informs T_1 of the confirmation

14. In response, if SSP_1 receives *END 3: {RR 31: [Pri_Call_Res (DISC)]",*
    a. SSP_1 disconnects T_1 as described instep 3a
    b. Else, SSP_1 sends *USER INFO [Pri_Call_Confirm(T_C)]* to T_1 confirming call and plays the announcement via the D-Channel.
15. When T_1 receives T_C from SSP_1, it save T_C, deletes T_A, and waits for K_{ET1}.

K_E generation and distribution

16. If AC authenticates S2 and sends "*END 5: { RR 51: [Sub_Auth_Res (T_C)]}",* it generates a new K_E and t_E, encrypts both K_E, and t_E, with KT_i and K*AC $KT_i[K_E\ t_E,]$ to produce *K*AC{KT_i[K_E, t_E,]}* for i=1,2

Transaction 6 (AC invokes the key distribution request in SSP_1)

17. AC initiate a unidirectional transaction 6 by sending *UNIDIRECTIONAL 6: { INVOKE 61: [Key_Dist_Req (K_{ET1})]}* SSP_1.

Transaction 7 (AC invokes the key distribution request in SSP_2)

18. AC initiate a unidirectional transaction 7 by sending *UNIDIRECTIONAL 7: { INVOKE 71: [Key_Dist_Req (K_{ET2})]}* to SSP_2.
19. When $SSP_{1(2)}$ receives "*UNIDIRECTIONAL 6(7): {INVOKE 6(7)1: [Key_Dist_Req ($K_{ET1(2)}$)]}* it forwards *USER INFO [Key_Dist_Req ($K_{ET1(2)}$)]* to $T_{1(2)}$.
20. In response, $T_{1(2)}$ decrypts K_E, and t_E.

Trunk Setup

21. After SSP_1 forwards K_{ET1} to T_1, it identifies an available trunk, and sends an *initial address message IAM()* to SSP_2
22. In response SSP_2 sends an address complete message (ACM) and Answer Message (ANM) respectively to SSP_1. When both are received, SSP_1 sets up the forward path of the trunk between SSP_1 and SSP_2.

Private communication

23. The private conversation between S_1 and S_2 starts.

Releasing the privacy call and deleting credentials

24. When S_2 hangs up, T_2 sends a *DISC* message to SSP_2. In response, SSP_2 sends a *RLSE* message to T_2, and a *REL* message to SSP_1.
25. When T_2 receives *RLSE* message from SSP_2, it sends *RLCOM* message to SSP_2, and deletes K_E and T_C.
26. In response, SSP_1 sends *DISC* message to T_1.
27. When T_1 receives *DISC* message from SSP_1, it sends *RLSE* message to SSP_1. In response, SSP_1 sends *RLCOM* message to T_1, and T_1 deletes K_E and T_C. □

7 CORRECTNESS OF THE PROPOSED PROTOCOLS

This section proves the correctness of the protocols described in section 5.3.

Theorem 1: T must present a valid T_LN and T_DN pair to AC to receive the privacy service

Proof: Suppose that T present a non-valid T_LN-T_DN pair to the AC. Because T_LN-T_DN pair of T is not in the authentication database, AC will allow T to receive privacy service according to step 5c in section 5.3.1**.** □

Theorem 2: S_ID and S_PW are necessary to request/receive privacy call.

Proof: Suppose that S want to request privacy call set up, and uses a participating T, but presents a non valid S_ID to AC. Since S_ID is not valid, AC will reject the request according to step 6c in section 5.3.2.

Suppose that S want to accept privacy call, but presents a non valid S_PW to AC. Because S_PW is not valid, AC will reject the request according to step 6c in section 5.3.2. □

Theorem 3: Any T can receive voice signals but only the T with right credentials can decrypt the signals.

Proof: Suppose there are two telephones (T_1 and $T`_1$) that share the same telephone line number ($T_1$_LN), and $T_1$ is privately communicating with $T_2$ on $T_2$_LN. Because $T_1$ and $T`_1$ share the same T_1_LN, $T`_1$ receives $T_1$'s signals. However, $T`_1$ cannot decrypt signals, because $T`_1$ does not know either the private key of $T_1$ ($K*T_1$) or $K_E$. In addition, $T`_1$ can not impersonate T_1 because $T`_1$ does not know the telephone device number of T_1 (T_1_DN) see next the theorem. □

Theorem 4: If a T presents T_C, which was issued to another T, to AC to request a new K_E, then AC rejects the request.

Proof: Suppose that $T`_1$ recovers a valid $T_C$ from $T_1$_LN and present the $T_C$ to AC in order to request a new $K_E$. Because ($T_1$_LN, $T`_1$_DN) is different from the (T_1_LN, T_1_DN) in the authentication database, the $T`_1$ request will be rejected by AC according to step 4d in section 5.3.6. □

Theorem 5: Any T can modify T_C but it can not present the modified T_C to AC.

Proof: Suppose that $T`_1$ attempts to modify $T_C$ of $T_1$ in order to pretend that $T_C$ was issued to $T`_1$. Once a parameter in T_C is changed, T_C needs to be encrypted with K*AC, and since $T`_1$ does not have K*AC, it can not sign the modified $T_C$. Hence, $T`_1$ can not present a modified T_C to AC. □

Theorem 6: Any T can request a new K_E using T_DN of another T, But it will not be able to recover K_E from K_{ET}

Proof: Suppose that $T`_1$ somehow obtained the TDN_1 and used it to request a new K_E. Once the steps as described in section 5.3.6 are performed, AC issues a new K_E and encrypts it with KT_1

then sends the encrypted value to T_1. $T`_1$ can retrieve the encrypted value from the $T_1$_LN but it can not recover the new $K_E$, because $T`_1$ needs $K*T_1$ as shown in step 9 of section 5.3.6. □

Remark: Once the private key of the telephone (K*T) is compromised CA will issue a new public/private key pair to the telephone, to delete the compromised one, and publish in the certificate revocation list (CRL).

8 PERFORMANCE OF THE PROPOSED PROTOCOLS

This section describes the expected delay for voice privacy call. We first describe end-to-end delay for a typical call that does not need database assistance. Next, we describe end-to-end delay for a calling card call that requires database assistance. Then, we describe end-to-end voice privacy call. Finally, we compare all three delays.

8.1 Typical Calls

A typical call, described in section 3.3 does not require database lookups or supplementary services. The calculation of end-to-end delay for typical call involves three parameters and they are as follows:

- **Call setup delay** is the time interval from T_1 sends SETUP message to SSP_1 until T_1 receives ALERT from SSP_1.
- **Call answer delay** is the time interval between T_2 sending CONN message to SSP_2 until T_1 receiving CONN message from SSP_1.
- **Call release delay** is the time interval between T_2 sending DISC message to SSP_2 until T_2 receiving RLSE message from SSP_2.

Parameters	**Mean Time Delay under Normal Load**[13]
Setup on local connection	3.0 sec
Answer on local connection	0.75 sec
Release on local connection	0.4 sec

Table 4. Mean Time Delay For Typical Call

The mean time of end-to-end connection establishing is the sum of the mean time for setting up the connection and answering it, and is equal to 3.75 seconds as shown in Table 4 an in Figure 5. This time does not include the time for SSP_2 to wait S_2 to anwer the telephone. In other word, S_1 waits at least 3.75 seconds to hear the voice of S_2.

8.2 Calling Card Calls

In calling card call, the connection needs to be authorized by the *Service Control Point* (SCP) before SSP_1 can establish the connection as shown in Figure 16. The calculation of end-to-end delay for calling card call involves the parameters and they are as follows:

- **Call Authorization delay** is the time interval from T_1 sends SETUP message with access telephone number to SSP_1 until T_1 receives enter called number announcement.
- **Call setup delay** is the time interval from T_1 sends called telephone number to SSP_1 until T_1 receives ALERT from SSP_1.
- **Call answer delay** is the time interval from T_2 sends CONN message to SSP_2 until T_1 receives CONN message from SSP_1.
- **Call release delay** is the time interval from T_2 sends DISC message to SSP_2 until T_2 receives RLSE message from SSP_2.

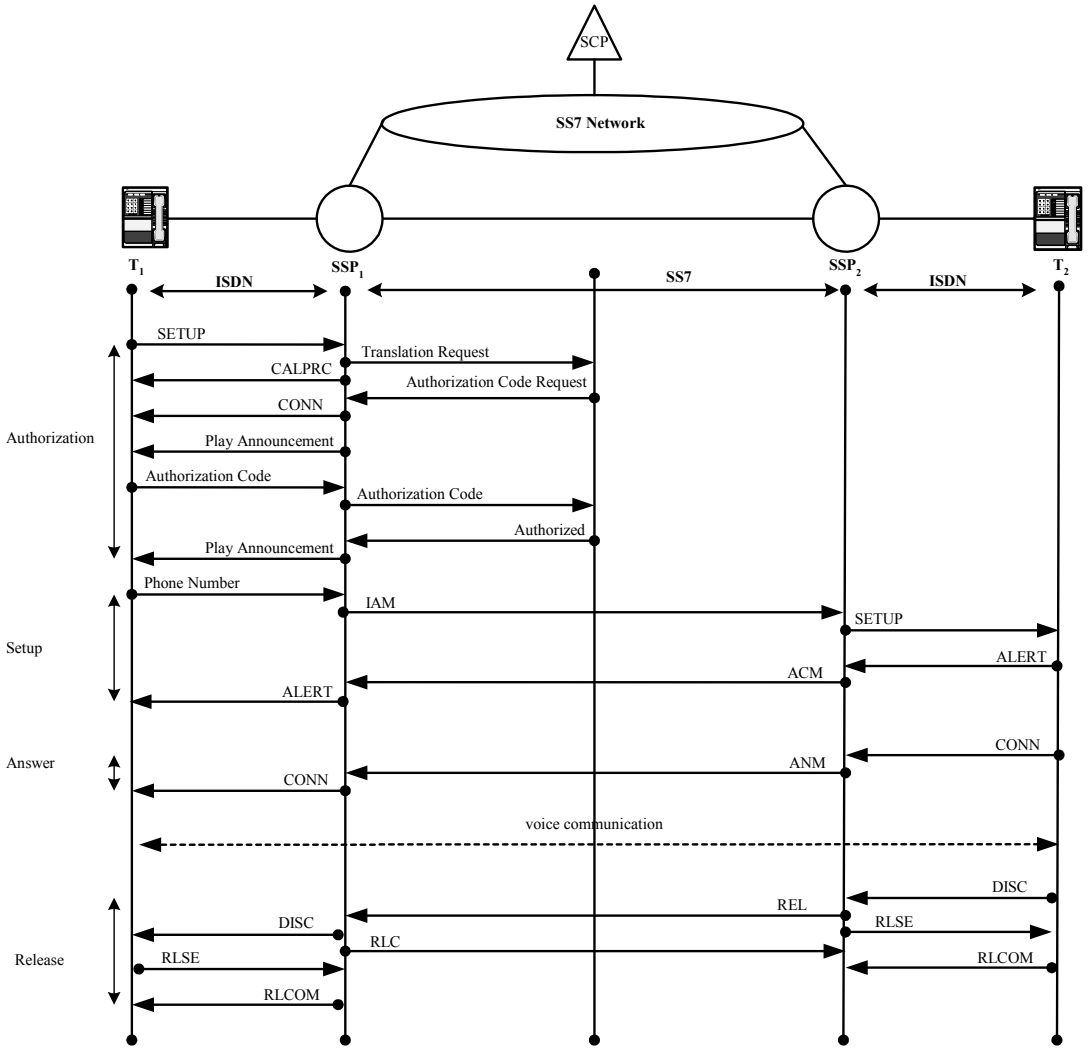

Figure 16. Calling Card Call Processing

Parameters	Etimated Mean Time Delay under Normal Load	Description of Estimated values
Authorization	4.6 sec	This value consists of two round trips between SSP_1 and AC, and five one-way delay between SSP_1 and T_1. It includes 0.8 sec delay, the ITU-T E.723 [14] estimation for mean round trip delay for database access not including application process, and 0.2 sec delay for one-way delay between SSP and T [13]. We assume that the delay for application process is 0.1 sec at each signaling point (SP) and is 0.2 sec at the telephone (T). Also, we assume that the delay for

		subscriber to enter the authorization code is 2.0 sec.
Setup on local connection	3.0 sec	ITU-T E.721
Answer on local connection	0.75 sec	ITU-T E.721
Release on local connection	0.4 sec	ITU-T E.721

Table 5. Mean Time Delay For Calling Card Call

The mean time of end-to-end connection establishing for calling card call is the sum of the mean time for authorizing, setting up and answering the connection, and is equal to 8.35 seconds as shown in Table 5 and Figure 16. Once S_1 enters the access number and T_1 send SETUP message to SSP_1, S_1 will be asked to enter the authorization code 1.4 second later. Then, S_1 will be asked to enter T_2's number 1.4 second later. After 3.0 second delays, S_1 will hear the ringing tone. Once S_1 answers the call and 0.75 second later, the conversation between S_1 and S_2 starts. This time does not include the time for SSP_2 to wait S_2 to anwer the telephone. In other word, S_1 waits at least 8.35 seconds to hear the voice of S_2.

8.3 Voice Privacy Call

Voice privacy call processing is described in section 8 require database assistances as well as subscriber and telephone inputs as shown in Figure 15. The calculation of end-to-end delay for voice privacy call involves several parameters we summarized these parameters below. The estimated values of these parameters are shown in Table 6.

- **Initiating voice privacy delay** is the time interval between T_1 sending the SETUP message to SSP_1 and SSP_1 receiving it.
- **T_1 Authentication delay** is the time interval between SSP_1 sending the Tel_Val_ Req message to AC and SSP_1 receiving it.
- **B-Channel Setup at T_1 delay** is the time interval between SSP_1 sending CALPRC to T_1 and T_1 receives it.
- **S_1 Authentication delay** is the time interval between T_1 receives CONN from SSP_1 and T_1 receiving the Pri_Call_Approve message from SSP_1.
- **Informing the privacy call request to SSP_2 delay** is the time interval between SSP_1 sending Pri_Call_Req message and SSP2 receiving it.
- **D-Channel setup at T_2** is the time interval between SSP_2 sending the SETUP message to T_2 and SSP_2 receiving CALPRC from T_2.
- **T_2 Authentication delay** is the time interval between SSP_2 sending Tel_Val_ Req message to AC and T_2 receives CALPRC from SSP_2.
- **B-Channel setup at T_2 delay** is the time interval between SSP_2 sending CALPRC and T_2 receives ALERT from T_2.
- **S_2 Authentication delay** is the time interval from T_2 sending CONN to SSP_2 until T_1 receiving Pri_Call_Confirm message from SSP_1.
- **Informing the response of the privacy call request to SSP_1 delay** is the time interval between SSP2 sending Pri_Call_Res until T_1 receiving Pri_Call_Confirm or DISC message from SSP_1.
- **K_E Generation and Distribution delay** is the time interval between AC starting to generate KE until T_1 and T_2 starting the conversation.

- **Trunk Setup delay** is the time interval between SSP_1 sending IAM message to SSP_2 and SSP_1 receiving ANM from SSP_2.
- **Releasing Privacy Call delay** is the time interval between T_2 sending DISC message to SSP_2 until T_2 receiving RLSE message from SSP_2.

Parameters	Estimated Mean Delay on Normal Load	Description of Estimated value
Initiating voice privacy	0.2 sec	Mean one-way delay between SSP and T [13]
T_1 Authentication	2.6 sec	This value consists of two round trips between SSP_1 and AC, and one round trip delay between SSP_1 and T_1. It includes 0.8 sec delay, the ITU-T E.723 [14] estimation for mean round trip delay for database access not including application process, and 0.2 sec delay for one-way delay between SSP and T [13]. We assume that the delay for application process is 0.1 sec at each signaling point (SP) and is 0.2 sec at the telephone (T).
B-Channel Setup at T_1	Overlap with S_1 authentication delay	Mean one-way delay between SSP and T [13]
S_1 Authentication	4.4 sec	This value consists of two round trips between SSP_1 and AC, and one round trip between SSP_1 and T_1. It includes our assumption of 2.0 sec delay for subscriber to enter his ID and password.
Informing privacy call request to SSP_2	0.5 sec	This value is based on information from ITU-T E.721 and Q.709 [13,16]
D-Channel setup at T_2	0.4 sec	Mean one-way delay between SSP and T [13]
T_2 Authentication	2.6 sec	See description of T_1 authentication
B-Channel setup at T_2	0.2 sec	Mean one-way delay between SSP and T [13]
S_2 Authentication	4.4 sec	See description of S_1 authentication
Informing the result of privacy call request to SSP_1	0.5 sec	This value is based on information from ITU-T E.721 and Q.709 [13,16]
K_E Generation & Distribution	1.5 sec	This is based on information from ITU-T E.721 and our assumption of application process at SP and T.
Trunk Setup	2.4 sec	This value is based on information from ITU-T E.721 and E.723 [13,14]
Releasing Privacy Call	0.4 sec	ITU-T E.721 [13]

Table 6. Estimated Mean Time Delay For Privacy Call

The estimated value of end-to-end voice privacy connection establishing is the sum of all parameter times in Table 5 (except releasing time) and it is equal to 19.7 seconds including 4

seconds delay for S_1 and S_2 to enter their user IDs and passwords. Moreover, S_1 will hear a ringing dial tone or disconnection announcement in 3 seconds and then S_1 will hear the ID and password request announcement in 1.2 seconds later. After S_1 enters the requested ID and password, S_1 will hear the connection approval or disconnection announcement 1.2 seconds later. After 8.6 seconds of T_2 and S_2 authentications, S_1 will hear the connection confirmation or disconnection announcement. After 3.9 second of key generation & distribution and trunk setup S_1 will hear S_2s voice over the encrypted B-Channel.

In conclusion, the privacy call setup delay under normal traffic loads is 19.7 seconds, calling card call setup delay under normal traffic loads is 8.35 second, and typical call setup delay under normal traffic loads is 3.75 seconds. Comparing the voice privacy delay with typical and calling card delay, the voice privacy delay is higher than the other two. Because the privacy call setup requires at least seven database transactions and four subscriber inputs, it takes twice longer than to setup a calling card calls requiring only one database transaction. Furthermore, the privacy call setup takes five limes longer than normal call setup that does not require any database transaction. We are in processing of developing simulation code to confirm and/or improve the end-to-end delay for proposed protocols.

9 SOFTWARE DESIGN FOR THE COMPONENTS OF VOICE PRIVACY PROTOCOLS

This section describes the software design for the components of the voice privacy protocol. They are calling telephone, originating switch, authentication center, terminating switch, and called telephone. Moreover, authentication center consists of two parts. One part provides support to the originating switch and the other part provides support to terminating switch. For each of the component, we first show the state diagram and we next describe the input and output messages. Finally, we describe the pseudo code.

9.1 Originating Switch (SSP_1)

9.1.1 State Diagram

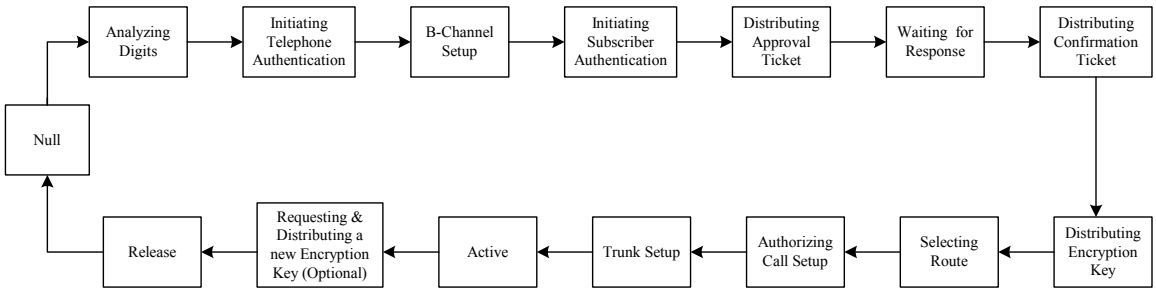

Figure 17. Originating Switch (SSP_1) States

9.1.2 Input and Output Messages

1. When SSP_1 receives *SETUP* message from T_1, it examines
   a. If *SETUP* does not contains privacy code (Pri_Code), perform typical connection setup
   b. Else, go to step 2.
2. SSP_1 initiate transaction 1 by invoking the telephone line validation request *BEGIN 1: {INVOKE 1: [Tel_Val_Req(T_1_LN)]}* in AC.

3. When SSP_1 receives *CONTINUE 1: {INVOKE 2: [Tel_Auth_Req (K*AC[R])]}* message from AC, it reformats and forwards the request *USER INFO [Tel_Auth_Req (K*AC[R])]* to T_1.
4. When SSP_1 receives *USER INFO [Tel_Auth_Res{K*T_1(KAC[T_1_ND,T_1_LN], R)}]* message from T_1, it reformats and forwards the request *CONTINUE 1: {RR 2: [Tel_Auth_Res{K*T_1(KAC[T_1_ND,T_1_LN], R)}]}*to AC as response to invoke 2 of transaction 1.
5. If the response from AC is *END 1: {RR 1: [Tel_Val_Res (DISC)]}*,
   a. SSP_1 allocates B-channel on T_1's line and sends *CALPRC* message to T_1 to inform the allocation of B-channel and the connection setup is in progress.
   b. SSP_1 connect announcement source to B-Channel to play disconnection announcement and sends *DISC* message to T_1 via D-channel.
   c. Else, go to step 6
6. SSP_1 allocates B-channel on T_1's line and sends *CALPRC* message to T_1 to inform the allocation of B-channel.
7. SSP_1 initiate transaction 2 by invoking the subscriber authentication request *BEGIN 2:{INVOKE 21: [Sub_Auth_Req (S_1)]}* in AC.
8. When SSP_1 receives *CONTINUE 2: {INVOKE 22: [Sub_PW_Req(STR-I)]}* request from AC,
   a. SSP_1 sends *CONN* message to T_1 via D-channel.
   b. SSP_1 connects announcement source to B-Channel to play calling subscriber password request announcement.
9. When SSP_1 receives *S_1_RES* from T_1, it forwards it to AC as response to invoke 22 of transaction 2 by sending a *CONTINUE 2: {RR 22: [Sub_PW_Res(S_1_RES)]}* to AC
10. If the response from AC is *END 2: {RR 21: [Sub_Auth_Res (DISC)]}*,
    a. SSP_1 connect announcement source to B-Channel to play disconnection announcement and sends *DISC* message to T_1 via D-Channel.
    b. Else, go to step 11.
11. SSP_1 connects announcement source to B-Channel to plays the privacy call approved announcement to S_1, and forwards T_A to T_1 via D-Channel by sending a *USER INFO [Pri_Call_Approve(T_A)]* to T_1.
12. SSP_1 initiate transaction 3 by invoking the privacy call request *BEGIN 3: { INVOKE 31: [Pri_Call_Req(T_A, T_1_LN, T_2_LN)]}* to SSP_2.
13. If the response from SSP_2 is *END 3: {RR 31: [Pri_Call_Res(DISC)]},*
    a. SSP_1 connects announcement source to B-Channel to play disconnection announcement, and sends *DISC* message to T_1 via D-Channel.
    b. Else, SSP_1 connects announcement source to B-Channel to plays the privacy call confirmation announcement to S_1, and forwards T_C to T_1 via D-Channel by sending a *USER INFO [Pri_Call_Confirm(T_C)]* to T_1.
14. When SSP_1 receives "*UNIDIRECTIONAL 6: {INVOKE 61: [Key_Dist_Req_Req (K_{ET1})]}"* message from AC_2, it forwards the message to T_1 by sending a *USER INFO [Key_Dist_Req (K_{ET1})]* to T_1.
15. After SSP_1 forwards K_{ET1} to T_1, it identifies an available trunk, and sends *IAM* to SSP_2.
16. Once SSP_1 receives *ACM* message from SSP_2, it goes to wait state until it receives *ANM* message from SSP_2. Once *ANM* message arrives, SSP_1 sets up the forward path of the trunk between SSP_1 and SSP_2.
17. When SSP_1 receives the *REL* message, it sends a *RLC* message to SSP_2, and sends *DISC* message to T_1.
18. When SSP_1 receives *RLSE* message, it sends *RLCOM* to T_1.

9.2 Terminating Switch (SSP_2)

9.2.1 State Diagram

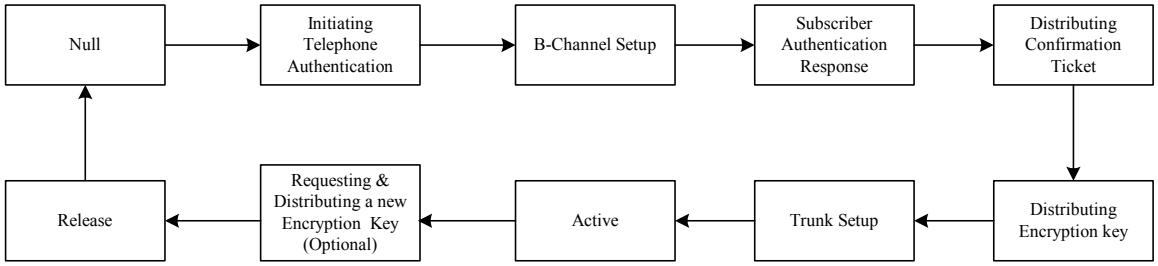

Figure 18. Terminating Switch (SSP_2) States

9.2.2 Input and Output Messages

1. When SSP_2 receives *BEGIN 3: {INVOKE 31:[Pri_Call_Req(T_A, T_1_LN, T_2_LN)]}* message from SSP_1, it allocates D-channel on T_2's line, and sends *SETUP* message to T_2 to inform the connection setup request and the allocation of D-channel.
2. When SSP_2 receives *CALPRC* from T_2, it initiate transaction 4 by invoking the telephone line validation request *BEGIN 4: { INVOKE 41: [Tel_Val_Req(T_2_LN)]}* in AC.
3. Once SSP_2 receives *CONTINUE 4: {INVOKE 42: [Tel_Auth_Req (K*AC[R])]}* request, it reformat and forwards the request to T_2 by sending a *USER INFO [Tel_Auth_Req (K*AC[R])]* to T_2.
4. When SSP_2 receives *Tel_Auth_Res {K*T_2(KAC[T_2_DN, T_2_LN], R)}* message from T_2, it forwards the message to AC as response to invoke 42 of transaction 4 by sending a *CONTINUE 4: [RR 42: (Tel_Auth_Res {K*T_2(KAC[T_2_DN, T_2_LN], R)})]* to AC.
5. If the response from AC is *END 4: {RR 41: [Tel_Val_Res (DISC)]}*,
   a. SSP_2 sends *DISC* message to T_2 via D-channel.
   b. SSP_2 informs failed T_2 authentication to SSP_1 as response to invoke 31 of transaction 3 by sending a *END 3: {RR 31: [Pri_Call_Res (DISC)]}*.
   c. Else, go to step 6.
6. SSP_2 allocates B-channel on T_2's line, and sends *CALPRC* message to T_2 to inform the allocation of B-channel.
7. When SSP_2 receives an *ALERT* message from T_2, it goes to wait state until *CONN* message arrives from T_2.
8. When SSP_2 receives *CONN* message from T_2, it initiate transaction 5 by invoking the called subscriber authentication request in AC by sending a *BEGIN 5: {INVOKE 51: [Sub_Auth_Req(T_A)]}*
9. When SSP_2 receives *CONTINUE 5: {INVOKE 52: [Sub_PW_Req (STR-I)]}* from AC, it connect announcement source to B-Channel to play password request announcement.
10. When SSP_2 receives *S_2_RES* response from T_2, it forwards the response to AC by sending a *CONTINUE 5: { RR 52: [Sub_PW_Res (S_2_RES)]}*
11. If the response from AC is "*END 5: {RR 51: [Sub_Auth_Res (DISC)]}"*
    a. SSP_2 connect announcement source to B-Channel to play disconnection announcement, and sends *DISC* message to T_2 via D-channel.
    b. SSP_2 informs failed S_2 authentication to SSP_1 as response to invoke 31 of transaction 3 by sending a *END 3: {RR 31: [Pri_Call_Res (DISC)]}*.
    c. Else, go to step 12.

12. SSP_2 connects announcement source to B-Channel to plays the privacy call Confirmation announcement to S_2, and forwards T_C to T_2 via D-Channel by sending a *USER INFO [Pri_Call_Confirm(T_C)]* to T_2.
13. SSP_2 sends privacy call response to SSP_1 to inform S_2 and T_2 authentication as response to invoke 31 of transaction 3 by sending a *END 3: {RR 31: [Pri_Call_Res(T_C)]}* to SSP_1.
14. When SSP_2 receives *UNIDIRECTIONAL 7: {INVOKE 71: [Key_Dist_Req_Req (K_{ET2})]}* message from AC, it forwards the message to T_2 by sending a *USER INFO [Key_Dist_Req (K_{ET2})]* to T_2
15. In response to *IAM* message from SSP_1, SSP_2 sends an *ACM* and an *ANM* respectively to SSP_1.
16. When SSP_2 receives *DISC* message, it sends *RLSE* message to T_2, and sends a *REL* message to SSP_1.
17. Once SSP2 receives *RLCOM* and *RLC* message from T_2 and SSP_1, respectively, SSP_2 ends the privacy call.

9.3 AC Software

AC software consists of two processes: one process provides support to the SSP_1 and the other provides support to SSP_2.

9.3.1 AC process provides support to SSP_1

9.3.1.1 State Diagram

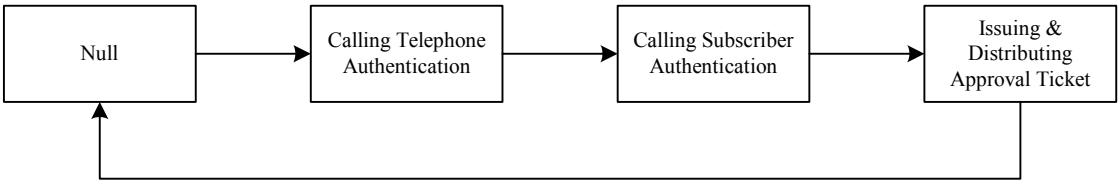

Figure 19. Authentication Center States that provide support to SSP_1

9.3.1.2 Input and Output Messages

1. Once AC receives *BEGIN 1: {INVOKE 1:[Tel_Val_Req(T_1_LN)]}* request, it generates a random number (R), encrypts R with K*AC, and sends the encrypted value to T_1 through SSP_1 with request to invoke the telephone authentication request. This is done a continuation of transaction 1 by sending invoke 2 message *CONTINUE 1: {INVOKE 2:[Tel_Auth_Req (K*AC[R])]}* to SSP_1.
2. When AC receives *CONTINUE 1: {RR 2:[Tel_Auth_Res{K*T_1(KAC[T_1_ND,T_1_LN], R)}]}* response,
   a. AC decrypts the response by applying the public key of the telephone (KT_1) to obtain R and *KAC[T_1_ND,T_1_LN]*.
   b. AC decrypts *KAC[T_1_DN,T_1_LN]* to determine whether T_1 signed up for the privacy service by comparing the received *T_1_ND,T_1_LN pair* and the one in the privacy database.
   c. If the received R is the same as the original R, and the received *[T_1_ND,T_1_LN]* pair is the same as the one in the privacy database, AC sends *END 1: {RR 1: [Tel_Val_Res (CONT)]* to SSP_1 as response to invoke 1 of transaction 1 in order SSP_2 to continue to process the privacy call setup.
   d. Else, AC sends *END 1: {RR 1: [Tel_Val_Res (DISC)]* message to SSP_1 as response to invoke 1 of transaction 1.

3. Once AC receives *BEGIN 2:{INVOKE 21:[Sub_Auth_Req (S_1)]}* message from SSP_1, it requests SSP_1 to play the subscriber password request announcement by sending a *CONTINUE 2: {INVOKE 22:[Sub_PW_Req(STR-I)]}*. This invoke 22 is a continuation of transaction 2.
4. When AC receives *CONTINUE 2: {RR 22:[Sub_PW_Res(S_1_RES)]}* response, AC decrypts *S_1_RES* by applying K*AC to obtain *[S_1_ID, S_1_PW, S_2_ID, T_1_LN, T_2_LN, T_1_DN]*
   a. AC compares the received S_1_ID and S_1_PW with the ones in the authentication database, and then AC verifies whether *[T_1_DN, T_1_LN]* pair is in the privacy database or not.
   b. If S_1 is authenticated, and the received *[T_1_DN, T_1_LN]* pair is the same as the one in the privacy database, AC creates and sends *END 2: {RR 21: [Sub_Auth_Res (T_A)]}* to SSP_1 as a response to invoke 21 of transaction 2.
   c. Else, AC sends *END 2: {RR 21: [Sub_Auth_Res (DISC)]}* to SSP_1 as a response to invoke 21 of transaction 2.

9.3.2 AC process provides support to SSP_2

9.3.2.1 State Diagram

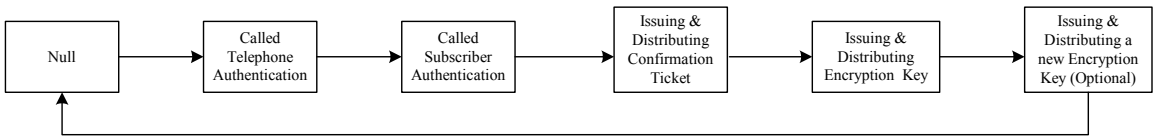

Figure 20. Authentication Center States that provide support to SSP_2

9.3.2.2 Input and Output Messages

1. Once AC receives *BEGIN 4: {INVOKE 41:[Tel_Val_Req(T_2_LN)]}* request, it generates a random number (R), encrypts R with K*AC, and sends the encrypted value to T_2 through SSP_2 with request to invoke the telephone authentication request by sending a *CONTINUE 4: {INVOKE 42:[Tel_Auth_Req (K*AC[R])]}*.
2. When AC receives *CONTINUE 4: [RR 42:(Tel_Auth_Res{K*T_2(KAC[T_2_DN, T_2_LN], R)})]*,
   a. AC decrypts the response by applying the public key of the telephone (KT_2) to obtain R and *KAC[T_2_DN, T_2_LN]*.
   b. AC decrypts *KAC[T_2_DN, T_2_LN]* to determine whether T_2 signed up for the privacy service by comparing the received *[T_2_DN, T_2_LN]* pair and the one in the privacy database.
   c. If the received R is the same as the original R, and the received *[T_2_DN, T_2_LN]* pair is the same as the one in the privacy database, AC sends *END 4: {RR 41: [Tel_Val_Res (CONT)]* to SSP_2 as response to invoke 41 of transaction 4 in order SSP_2 to continue to process the privacy call setup
   d. Else, AC sends *END 4: {RR 41: [Tel_Val_Res (DISC)]* message to SSP_2 as response to invoke 41 of transaction 4
3. When AC receives "*BEGIN 5: {INVOKE 51: [Sub_Auth_Req(T_A)]}"* message from SSP_2, it retrieves parameters in T_A; *KAC (T_A)=[K*AC [S_1_ID, S_2_ID, T_1_LN, T_2_LN, t_A]*.
   a. If AC can not verify T_A, it sends *END 5: [RR 51: {Sub_Auth_Res(DISC)}]* message to SSP_2.

b. Else, AC requests SSP_2 to play the subscriber password request announcement by sending a *CONTINUE 5: {INVOKE 52: [Sub_PW_Req (STR-I)]}*. This invoke 52 is a continuation of transaction 5.

4. When AC receives "*CONTINUE 5:{RR 52:[Sub_PW_Res(S_2_RES)]}"* response,
   a. AC decrypts S_2_RES by applying K*AC, and then AC compares the received S_2_ID with the one in T_A.
      i. If the two S_2_ID*s* are not the same, AC sends *END 5: [RR 51: {Sub_Auth_Res(DISC)}]* to SSP_2 as response to invoke 51 of transaction 5.
      ii. Else, go to step b.
   b. AC compares the received S_2_ID and S_2_PW with the ones in the authentication database, and then verifies whether *[T_2_DN, T_2_LN]* pair is in the privacy database
   c. If S_2 is authenticated, and the received *[T_2_DN, T_2_LN]* pair is the same as the one in the privacy database, AC creates and sends a confirmation ticket *END 5: { RR 51: [Sub_Auth_Res (T_C)]}* to SSP_2 as response to invoke 51 of transaction 5. Then AC goes to step 5.
   d. Else, AC sends *END 5: [RR 51: {Sub_Auth_Res(DISC)}]* to SSP_2 as response to invoke 51 of transaction 5.
5. AC generates a new K_E and t_E, and encrypts *[K_E, t_E,]* with K*AC to obtain *K*AC[K_E, t_E,]*
   a. AC encrypts *K*AC[K_E, t_E,]* with KT_1 to obtain *K*AC{KT_1[K_E, t_E,]}* and initiates a unidirectional transaction 6 by sending a *UNIDIRECTIONAL 6:{ INVOKE 33: [Key_Dist_Req (K_{ET1})]}* to SSP_1.
   b. Again, AC encrypts *K*AC[K_E, t_E,]* with KT_2 to obtain *K*AC{KT_2[K_E, t_E,]}*, and initiates a unidirectional transaction 7 by sending *UNIDIRECTIONAL 7: { INVOKE 71: [Key_Dist_Req (K_{ET2})]}* to SSP_2.
   ;

9.4 Calling Telephone (T_1)

9.4.1 State Diagram

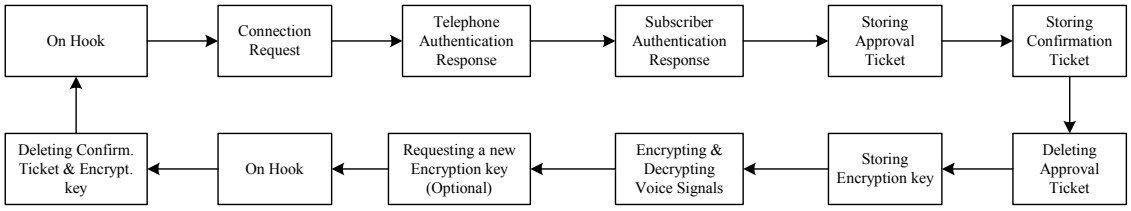

Figure 21. Calling Telephone (T_1) States

9.4.2 Input and Output Messages

1. S_1 enters TLN_2 and privacy code (*) on T_1 and pushes "Enter" button on T_1.
2. T_1 sends a *SETUP(*,T_1_LN, T_2_LN)* to SSP_1
3. If the message from SSP_1 is *DISC*, T_1 releases D-Channel and sends *RLSE* message to SSP_1 to confirm the release D-Channel..
4. Else, T_1 decrypts *KAC(K*AC[R])* by applying KAC to obtain *R*
5. T_1 encrypts *T_1_DN and T_1_LN* with *KAC*, and then encrypts the received R and *KAC[T_1_DN,T_1_LN]* with K*T_1 to obtain *K*T_1(KAC[T_1_DN,T_1_LN], R)*.
6. T_1 sends *USER INFO [Tel_Auth_Res{K*T_1(KAC[T_1_DN,T_1_LN], R)}]* to SSP_1.

7. If the message from SSP_1 is *DISC*, T_1 releases D-Channel and sends *RLSE* message to SSP_1 to confirm the release D-Channel.
8. Else, T_1 starts to listen the assign B-Channel and waits for *ALERT* or/and *CONN* message from SSP_1.
9. Once T_1 receives *ALERT* message from SSP_1, it starts to dial-tone ringing.
10. Once T_1 receives *CONN* message from SSP_1, it stops dial-tone ringing and allows S_1 to hear the announcement from SSP_1.
11. T_1 collects S_1's entry and adds *T_1_LN, T_2_LN, and T_1_DN*. Then T_1 encrypts these values with KAC, and sends this encrypted value (S_1_RES) to SSP_1. *Where S_1_RES = KAC [S_1_ID, S_1_PW, S_2_ID, T_1_LN, T_2_LN, T_1_DN]*
12. If the message from SSP_1 is *DISC*, T_1 releases D-Channel and B-Channel and sends *RLSE* message to SSP_1 to confirm the release of the channels.
13. Else, T_1 save T_A until T_C arrives
14. If the message from SSP_1 is *"DISC"*, T_1 releases D-Channel and B-Channel and sends *RLSE* message to SSP_1 to confirm the release of the channel.
15. Else, T_1 save T_C, deletes T_A, and waits for K_{ET1}
16. Once T_1 receives *USER INFO[Key_Dist_Req(K_{ET1})]* from SSP_1, T_1 decrypts K_{ET1} by applying K^*T_1 and *KAC*, respectively, to obtain [K_E, t_E,].
17. T_1 starts encrypting/decrypting the conversation between S_1 and S_2 using K_E
18. When T_1 receives *DISC* message, it releases D-Channel and B-Channel and sends *RLSE* message to SSP_1 to confirm the release of the channel.
19. When T_1 receives *RLCOM* from SSP_1, it deletes K_E and T_C.

9.5 Called Telephone (T_2)

9.5.1 State Diagram

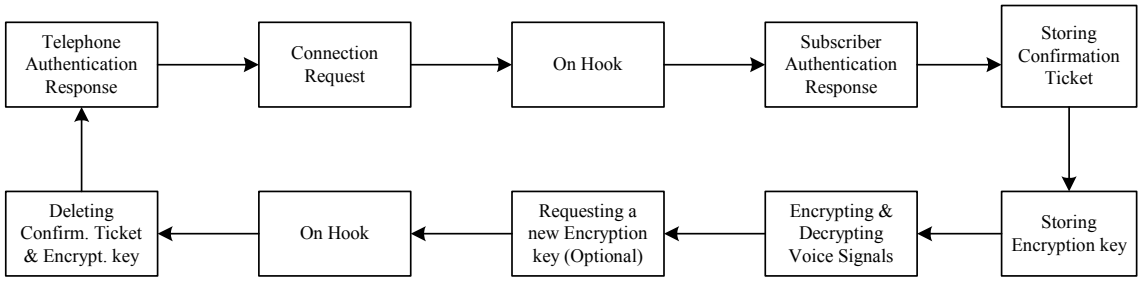

Figure 22. Called Telephone (T_2)

9.5.2 Input and Output Messages

1. When T_2 receives *SETUP* message from SSP_2, it allocates the requested D-channel, and sends *CALPRC* to SSP_2 to confirm the allocation of the channel.
2. If the message from SSP_2 is *DISC*, T_2 releases D-Channel and sends *RLSE* message to SSP_1 to confirm the release D-Channel.
3. Else, T_2 decrypts *KAC(K*AC[R])* by applying KAC to obtain *R*
4. T_2 encrypts T_2_DN and T_2_LN with KAC to obtain *KAC[T_2_DN, T_2_LN]* and then encrypts *KAC[T_2_DN, T_2_LN]* with K*T_2 to obtain *K*T_2(KAC[T_2_DN, T_2_LN], R)*
5. T_2 sends *USER INFO (Tel_Auth_Res {K*T_2(KAC[T_2_DN, T_2_LN], R)})* to AC through SSP_2.
6. If the message from SSP_2 is *DISC*, T_2 releases D-Channel and sends *RLSE* message to SSP2 to confirm the release D-Channel.
7. Else, T_2 starts to listen the assigned B-Channel, alerts S_2, and sends *ALERT* message to SSP_2.

8. When S_2 answers T_2, T_2 sends *CONN* message to SSP_2..
9. T_2 collects S_2's entry and adds T_2_LN, T_2_DN. Then T_2 encrypts these values with KAC and sends this encrypted value (S2_RES) to SSP_2. *Where S_2_RES = KAC[S_2_ID, S_2_PW, T_2_LN, T_2_DN]*.
10. If the message from SSP_2 is *DISC*, T_2 releases D- Channel and B-Channel and sends *RLSE* message to SSP2 to confirm the release of the channels.
11. Else, T_2 saves T_C , and waits for K_{ET2}.
12. Once T_2 receives *"USER INFO[Key_Dist_Req(K_{ET2})]"* from SSP_2, it decrypts K_{ET2} by applying K^*T_2, and *KAC*, respectively, to obtain *[K_E, t_E,]*
13. T_2 starts encrypting/decrypting the conversation between S_1 and S_2 using K_E
14. Assume S_2 hangs up T_2 and then T_2 sends a *DISC* message to SSP_2.
15. When T_2 receives *RLSE* message, it sends *RLCOM* to SSP_2, and deletes K_E and T_C

10 CONCLUSIONS

We have described voice privacy protocol to provide end-to-end voice privacy at the application layer with minimum modification to existing public telephone networks. Voice privacy is achieved by encrypting voice signals between the two end telephones using a symmetric key algorithm and one-time encryption keys. One-time encryption key is used to prevent replay attacks. In addition, we have described authentication protocols for subscribers and telephones. Telephone authentication protocol provides AC the assurance that the telephone at the other end of the connection is what it claims to be. Next, we described how to integrate voice privacy protocol in PSTN. Then, we described a detail of software design for the components of voice privacy. Finally, we provided performance characteristics of the voice privacy protocol. We are in process of developing simulation code to confirm and/or improve the end-to-end delay for voice privacy call, and we are looking on the effect of distributed authentication center may have on voice protocol. The summary of the comparison between the proposed voice privacy protocol and the most widely used security protocols in telephony is given in table 7.

	Proposed Voice Privacy	IS-41 Security	GSM Security	STU III [20]
Devices Authentication	Yes	Yes	Yes	No
Subscriber Authentication	**Yes**	**No**	**Only in Europe**	**No**
Voice Privacy between the telephone and SSP or MSC	Yes	Yes	Yes	Not Applicable
End-to-End Voice Privacy	**Yes**	**No**	**No**	**Yes**
One time encryption key	**Yes**	**Not Always**	**Not Always**	**Not Always**

Table 7. Comparison Between The Proposed and Existing Security Protocols

REFERENCES

[1] Berman, R. K. and Brewster, J. H., "Perspective on the AIN Architecture", IEEE Communications Magazine, 31, No. 2, February 1992.

[2] Black, U., "ISDN and SS7", Prentice Hall PTR, Upper Saddle River, New Jersey, 1997.

[3] Bosse, J. G. von, "Signaling IN Telecommunication Networks", John Wiley & Sons, New York, 1998.

[4] Baum, M. S. and Ford, W., "Secure Electronic Commerce", Prentice Hall PTR, Upper Saddle River, New Jersey, 1997.

[5] Carne, E. B., "Telecommunications Primer, Second Edition", Prentice Hall PTR, Upper Saddle River, New Jersey, 1999.
[6] Chlamtac, I., and Lin, Y., "Wireless and Mobile Network Architectures", John Wiley & Sons, New York, 2001.
[7] Chow, M., "Understanding Telecommunications: Systems, Networks and Applications", Volume 1, Andan Publisher, Holmdel, New Jersey, 2000.
[8] Department of Defense Security Institute, "STU-III Handbook for Industry", http://www.tscm.com/STUIIIhandbook.html, February 1997.
[9] Douskalis, B., "IP Telephony", Prentice Hall PTR, Upper Saddle River, New Jersey, 2000.
[10] Gallagher, M. D. and Snyder, R. A., "Wireless Telecommunications Networking with ANSI-41", Second Edition, McGraw-Hill, New York, 2001.
[11] GR-246-Core, "Telcordia Technologies Specification of Signalling System No. 7", Telcordia Technologies, Vol. 1, 2 & 3, Issue 7, December 2002.
[12] ISAAC security research group, "GSM Cloning", http://www.isaac.cs.berkeley.edu/isaac/gsm-faq.html, http://www.isaac.cs.berkeley.edu/isaac/gsm.html.
[13] ITU-T Recom. E.721, "Network grade of service parameters and target values for circuit-switched service in the evolving ISDN", ITU-T, Geneva, May 1999.
[14] ITU-T Recom. E.723, "Grade of service parameters for signaling system 7 networks", ITU-T, Geneva, June 1992.
[15] ITU-T Recom. I.352, "Network performance objectives for connection processing delays in an ISDN", ITU-T, Helsinki, March 1993.
[16] ITU-T Recom. Q.709, "SS7 – Hypothetical Signalling Reference Connection", ITU-T, Helsinki, March 1993.
[17] Modarressi, A. R., and Skoog, R. A., "Signaling System No. 7: A Tutorial", IEEE Communications Magazine, pp. 19-35, July 1990.
[18] Modarressi, A. R., and Skoog, R. A., "An overview of Signaling System No. 7", Proceeding of the IEEE, Vol. 80, No. 4, pp. 590-606, April 1992.
[19] Noll, A. M., "Introduction to Telephones and Telephone Systems", Third Edition, Artech House, Boston, 1998.
[20] Ramteke, T., "Networks", Prentice Hall PTR, Upper Saddle River, New Jersey, 1994.
[21] Rappaport, T. S., "Wireless Communications", Prentice Hall PTR, Upper Saddle River, New Jersey, 2002.
[22] Rose, G., "Authentication and Security in Mobile Phones", http://people.qualcomm.com/ggr/QC/AUUG99AuthSec.pdf
[23] Russell, T., "Signaling System # 7", Second Edition, McGraw-Hill, New York, 1998.
[24] Schneier, B., "Applied Cryptography", Second Edition, John Wiley & Sons, New York, 1996.
[25] Schwartz, M., "Telecommunication Networks: Protocols, Modeling and Analysis", Addison_Wesley Publishing Company, Reading, Massachusetts, 1987
[26] Scourias, J., "Overview of the Global System for Mobile Communications", http://ccnga.uwaterloo.ca/~jscouria/GSM/gsmreport.html.
[27] SecureLogix, "TeleVPN", http://www.securelogix.com , June 2003.
[28] Sharif, M, and Wijesekera, D., Michael, J. B., "Providing Secure Communication Services on the Public Telephone Network Infrastructure", Proceeding of the 10th International Conference on Telecommunication Systems: Modeling and analysis, pp 214-224, October 3-6, 2002, Monterey, CA, USA.
[29] Sharif, M, and Wijesekera, D., "Providing Voice Privacy over the Public Switched Telephone Network", Proceeding of IFIP TC11 18th International Conference on Information Security (SEC2003), pp 25-36, May 26-28, 2003, Athens, Creece,
[30] Stallings, W., "ISDN: An Introduction", Macmillan Publishing Company, New York, 1989.

[31] Stallings, W., "Cryptography and Network Security", Second Edition, Prentice Hall PTR, Upper Saddle River, New Jersey, 1999.
[32] Tanenbaum, A. S., "Computer Networks" third Edition, Prentice Hall PTR, Upper Saddle River, New Jersey, 1996.

APPENDIX

A VOICE PRIVACY PARAMETERS

This section describes paramertes, which are used in proposed voice privacy architecture and all of these messages are sent on the D-channel on ISND and on the SS7 signaling network.

- *** (Asterisk)**
  Name: Privacy code
  Description: * is used to inform the SSP that the calling subscriber is requesting the privacy call setup or a new encryption key.
- **S**
  Name: Subscriber
  Description: S is a user who is signed up for the voice privacy service. Calling subscriber (S_1) initiates the privacy call request and called subscriber (S_1) accepts the privacy call request.
- **T**
  Name: Telephone
  Description: T is an intelligent device, which is capable of voice encryption and encryption key management. In addition, T_1 supports ISDN network and communicates to the network devices with or without subscriber intervention. S_1 uses calling telephone (T_1) to initiate the privacy call request and S_2 uses calling telephone (T_2) to accept the privacy call request.
- **AC**
  Name: Authentication Center
  Description: AC is responsible to authenticate the telephones and the subscribers. In addition, it responsible to generate and distribute the encryption key which is used to encrypt the voice signals.
- **CA**
  Name: Certificate Authority
  Description: CA is responsible for generating public/private keys, creating digital certificate of public keys, and storing the digital certificates in the publicly available database
- **K, K***
  Name: Public and Private key pair
  Description: The key pair is used to encrypt the communication between the telephone and the authentication center.
- **S_ID**
  Name: Subscriber identification
  Description: S selects subscriber identification (ID) when S signed up for the privacy service. S_ID identifies S.
- **S_PW**
  Name: Subscriber password
  Description: S selects password (PW) when S signed up for the privacy service, and the combination of PW and ID authenticates S

- **T_LN**
  Name: Telephone line number
  Description: Telephone is assigned 10-digit line number T_LN to identify the line directly connected between SSP and T.
- **T_DN**
  Name: Telephone device number
  Description: TDN is a unique serial number that is permanently stored in T, and is not visible to S. T_DN and T_LN uniquely identify T.
- S_1_RES
  Name: Encrypted response from the T_1
  Description: S_1_RES is encrypted with KAC to keep the S_1_PW and T_DN_1 private.
  $S_1\_RES = KAC [S_1\_ID, S_1\_PW, S_2\_ID, T_1\_LN, T_2\_LN, T_1\_DN_1]$
- **S_2_RES**
  Name: Encrypted response from the T_2
  Description: S_2_RES is encrypted with KAC to keep the S_2_PW and T_DN_2 private.
  $S_2\_RES = KAC [S_2\_ID, S_2\_PW, T_2\_LN, T_2\_DN]$
- **t_A**
  Name: Approval timestamp
  Description: t_A determines the period which the approval ticket (T_A) is valid.
- **t_C**
  Name: Confirmation timestamp
  Description: t_c determines the period which the confirmation ticket (T_C) is valid.
- **t_E**
  Name: Encryption key timestamp
  Description: t_E determines the period which the encryption key (K_E) is valid.
- **T_A**
  Name: Approval Ticket
  Description: T_A is a signed ticket and indicates that S_1 is approved to have private communication with S_2. AC encrypts with K*AC to make T_A tamper proof. T_1 can present T_A to the AC to proof that the S_1 has been approved to have private communication with S_2 if T_1 did not receive confirmation ticket (T_C) within a specified time. In addition, T_1 deletes T_A when it receives T_C.
  $T_A = K*AC [S_1\_ID, S_2\_ID, T_1\_LN, T_2\_LN, t_A]$,
- **T_C**
  Name: Confirmation Ticket
  Description: T_C is a signed ticket and indicates that S_1 and S_2 are approved to have private communication. AC encrypts with K*AC to make T_C tamper proof. Either T_1 or T_2 can present T_C to request a new voice channel encryption key. In addition, Both T_1 and T_2 will delete T_C when the voice channel is disconnected.
  $T_C = K*AC[S_1\_ID, S_2\_ID, T_1\_LN, T_2\_LN, t_C]$
- **K_E**
  Name: Voice channel encryption key
  Description: K_E is used to encrypt the voice signals between T_1 and T_2. In addition, both T_1 and T_2 will delete T_C when the voice channel is disconnected to prevent replay.
- **K_{ET}**
  Name: Key distribution for T

Description: K_{ET} is a signed value to assure T, that AC issued K_E. AC encrypts K_E and t_E with KT and then encrypts the encrypted value with K*AC.
$K_{ET1} = K*AC[KT_1(K_E, t_E)]$

- **R**
  Name: Random Number
  Description: AC generates and uses R to authenticate the telephone.
- **STR-I:**
  Name: Send to Resource, Caller Interaction
  Description: SSP connects to Intelligent Network Services Circuit (INSC), to play Password request announcement, and to collect the digits sent by T. In addition, SSP sends the collected digits to AC
- **DISC**
  Name: Disconnect subscriber line
  Description: SSP plays disconnect announcement to inform S that privacy call can not be set up. In addition, SSP disconnects message to T.
- **CONT**
  Name: Continue Call Setup
  Description: CONT informs SSP that T has been authenticated, SSP can continue the privacy call set up

B VOICE PRIVACY MESSAGES

This section describes messages, which are used in proposed voice privacy architecture and all of these messages are sent on the D-channel on ISND and on the SS7 signaling network.

- **Sub_Auth_Req (***parameters***)**
  Name: Subscriber Authentication Request
  Description: The service switching point (SSP) sends this message to authentication center (AC) to request subscriber authentication**.**
  Parameters**:** S_1 *or* T_A
  Example: *Sub_Auth_Req(*S_1*)*
- **Sub_Auth_Res (***parameter***)**
  Name**:** Subscriber Authentication Response
  Description*:* The AC sends this message to SSP to respond to *Sub_Auth_Req*. It includes approval or confirmation ticket if the subscriber is authenticated. Otherwise, it includes "DISC".
  Parameters: T_A *,*T_C *or DISC.*
  Example: *Sub_Auth_Res (*T_A*)*
- **Sub_PW_Req (***parameter***)**
  Name: Calling Subscriber Password Request
  Description: In response to Sub_Auth_Req, AC sends this message to SSP to request authentication information from the subscriber such as subscriber's ID and password.
  Parameters: *STR-I or DISC*
  Example: *Sub_PW_Req (DISC)*
- **Sub_PW_Res (***parameter***)**
  Name: Subscriber Password Response
  Description: This message is sent by the telephone in response to Sub_PW_Req and SSP forwards to AC. It includes encrypted response from the subscriber and telephone such as S_1_RES or S2_RES.

Parameters: *S_1_RES or* S2_RES
Example: *Sub_PW_Res (S_1_RES)*

- **Pri_Call_Req (***parameters***)**
  Name: Privacy Call Request
  Description: The originating SSP sends to the terminating SSP to request privacy call setup. It includes approval ticket.
  Parameters: *T_A , T_1_LN, and T_2_LN*
  Example: *Pri_Call_Req (T_A, T_1_LN, T_2_LN)*
- **Pri_Call_Res (***parameters***)**
  Name: Privacy Call Response
  Description: This message is sent by the terminating SSP. It includes confirmation ticket if the called subscriber is authenticated. Otherwise, it includes "DISC".
  Parameters: *T_C or DISC*
  Example: *Pri_Call_Res (T_C,)*
- **Pri_Call_Approved (***parameter***)**
  Name: Privacy Call Approved
  Description: The originating SSP sends this message to the calling telephone to inform the privacy call setup is been approved. It includes the approval ticket if the calling subscriber is authenticated. Otherwise, it includes "DISC".
  Parameters: *T_A or DISC*
  Example: *Pri_Call_Approved (T_A)*
- **Pri_Call_Confirm (***parameters***)**
  Name: Privacy Call Confirm
  Description The SSP sends this message to the telephone to inform the privacy call setup is confirmed. It includes the confirmation ticket if the called subscriber is authenticated. Otherwise, it includes "DISC".
  Parameters: *T_C or DISC*
  Example: *Pri_Call_Confirm (T_C)*
- **Key_Dist_Req (***parameter***)**
  Name: Key Distribution Request
  Description: This message is sent by the AC to distributed the encryption key
  Parameters: *K_{ET1} or K_{ET2}*
  Example: *Key_Dist_Req (K_{ET1})*
- **Enc_Key_Req (***parameters***)**
  Name: New Encryption Key Request
  Description: The telephone sends this message to request a new encryption key due to time out or compromise. It includes the confirmation ticket, and encrypted value of the telephone line and device number.
  Parameters: *T_C and KAC(T_LN, T_DN)*
  Example: *Enc_Key_Req [T_C, KAC(T_LN, T_DN)]*
- **Enc_Key_Res (***parameters***)**
  Name: New Encryption Key Response
  Description: This message is sent by the AC to respond to Enc_Key_Res message. It includes a new confirmation ticket if the request is approved. Otherwise, it includes "DISC"
  Parameters: *T_C and DISC*
  Example: *Enc_Key_Res (T_C)*
- **New_Conf_Ticket (***parameters***)**
  Name: New Confirmation Ticket Distribution

Description: When SSP receive an updated T_C from AC, it sends this message to the other SSP to forward the updated T_C.

Parameters: T_C

- ***Example:*** New_Conf_Ticket (T_C)
- **Tel_Val_Req (***parameter***)**
  ***Name*:** Telephone Validation Request
  Description: The SSP sends this message to the AC to request telephone authentication. It includes the telephone line number.
  Parameters: *T_LN*
  Example: *Tel_Val_Req (T_LN)*
- **Tel_Val_Res (***parameter***)**
  ***Name*:** Telephone Validation Response
  Description: The AC sends this message to the SSP to respond *Tel_Val_Req (T_LN) message.* It includes "CONT" if the telephone is authenticated, Otherwise, it includes "DISC".
  Parameters: *CONT or DISC.*
  Example: *Tel_Val_Res (CONT)*
- **Tel_Auth_Req (***parameter***)**
  ***Name*:** Telephone Authentication Request
  Description: The AC sends this message to the SSP to authenticate the telephone. It includes a signed random number. Then SSP forwards it to the telephone
  Parameters: *K*AC(R)*
  Example: *Tel_Auth_Req [K*AC(R)]*
- **Tel_Auth_Res (***parameters***)**
  ***Name*:** Telephone Authentication Response
  Description: The telephone sends this message to the AC to respond the *Tel_Auth_Req message*. It includes a signed value that contain encrypted telephone line and device number, and the received random number.
  Parameters: *K*T[KAC(T_LN, T_DN), R]*
  Example: *Tel_Auth_Res [K*T(KAC(T_LN, T_DN),R)]*

C PSEUDO CODE FOR THE COMPONENTS OF VOICE PRIVACY PROTOCOLS

C.1 Originating Switch (SSP_1)

```
/* Structure definition */

Struct PrivacyRequest
{
   OPName: String;
   Arg[1]: char;
   Arg[2]: char;
   Arg[3]: char;
} PriReq, Add;

Struct RemoteRequest
{
   OPName: String;
   Arg[1]: char;
   Arg[2]: char;
}  RemReq;

Struct RemoteRequestResponse
{
```

```
  OPName: String;
  Arg[1]: List;
} RemReqRes, RemRes;


/* Component Part with Class 1 Type */
TCAP.CP1 (Req)

/* ISDN Libraries */.
ISND.SETUP (Add)
ISDN.DISC(DISC)
ISDN.CONN (CONN)
ISDN.CALPRC (CALPRC)
ISDN.USERINFO (Info)

/* Analyzing Digits */
/* Correspond to step 1a in SSP1 Input and Output Messages section */
Add = { “SETUP”, Pri_Code, TLN_1, TLN_2};
PriReq = ISDN.Setup (Add)
IF          PriReq.Arg[1] != Pri_Code
            Perform typical call setup

/* Calling Telephone Authentication */
/* Correspond to step 1b, 2, 3, and 4 in SSP1 Input and Output Messages section */
ELSE        RemReqRes.OPName = “Tel_Val_Req”
            RemReqRes.Arg[1] = PriReq.Arg[2]
            RemRes = TCAP.CP1 (RemReqRes)
            RemReqRes = ISDN.USERINFO (RemRes)
            RemRes = TCAP.CP1 (RemReqRes)

/* B-Channel Setup */
/* Correspond to step 5 in SSP1 Input and Output Messages section */
            IF          RemRes.OPName = “Tel_Val_Res”  && RemRes.Arg[1] = “DISC”
                        ISDN.CALPRC (CALPRC)
                        ISDN.DISC (DISC)


/* Initiating Calling Subscriber Authentication  */
/* Correspond to step 6, 7, 8 and 9 in SSP1 Input and Output Messages section */
            ELSE        ISDN.CALPRC (CALPRC)
                        RemReqRes.OPName = “Sub_Auth_Req”
                        RemReq.Arg[1] = S_1
                        RemRes = TCAP.CP1 (RemReqRes)
                        IF          RemRes.OPName = “Sub_PW_Req” && RemRes.Arg[1] != STR-I
                                    ISDN.CONN (CONN)
                                    Play Pri_Call_Not_Accept_Announcement
                                    ISDN.DISC (DISC)
                        ELSE        ISDN.CONN (CONN)
                                    Play PW Req Announcement
                                    Collect S1_RES
                                    RemReqRes.OPName = “Sub_PW_Res”
                                    RemReqRes.Arg[1] = S1_RES
                                    RemRes = TCAP.CP1 (RemReqRes)

/* Initiating Calling Subscriber Authentication Response & Distribution of Approval Ticket */
/* Correspond to step 10 and 11 in SSP1 Input and Output Messages section */
                                    IF          RemRes.OPName = “Sub_Auth_Res” && RemRes.Arg[1] = DISC
                                                Play Pri_Call_Not_Approve_Announcement
                                                ISDN.DISC (DISC)
                                    ELSE        Play Pri_Call_Approve_Announcement
                                                RemReqRes.OPName = “Pri_Call_Approved”
                                                RemReqRes.Arg[1] = T_A
                                                ISDN.USER INFO (RemReqRes)

/* Informing Privacy call process request to Called Switch */
/* Correspond to step 12 in SSP1 Input and Output Messages section */
                                                PriReq.OPName = “Pri_Call_Req”
                                                PriReq.Arg[1] = T_A
                                                PriReq.Arg[2] = TLN_1
                                                PriReq.Arg[3] = TLN_2
```

```
                                        RemRes = TCAP.CP1 (PriReq)

/* Response from the called switch & Distribution of Confirmation Ticket */
/* Correspond to step 13 in SSP1 Input and Output Messages section */
                                        IF      RemRes.OPName = “Pri_Call_Res” && RemRes.Arg[1] = DISC
                                                Play Pri_Call_Not_Confirm_Announcement
                                                ISDN.DISC (DISC)
                                        ELSE    Play Pri_Call_Confirm_Announcement
                                                RemReqRes.OPName = “Pri_Call_Confirm”
                                                RemReqRes.Arg[1] = T_C
                                                ISDN.USER INFO (RemReqRes)
                                                Delay()

/* Distribution of Encryption Key */
/* Correspond to step 14 in SSP1 Input and Output Messages section */
                                                KeyDist = TCAP.CP1 (RemKeyDist)
                                                IF      KeyDist.OPName = “Key_Dist_Req” && KeyDist.Arg[1] != Null
                                                        ISDN.USER INFO (KeyDist)
END IF

/* Selecting Route */
/* Correspond to step 15 in SSP1 Input and Output Messages section */
No Modification is needed on the current code

/* Authorizing Call Setup */
/* Correspond to step 15 in SSP1 Input and Output Messages section */
No Modification is needed on the current code

/* Trunk Setup */
/* Correspond to step 15 and 16 in SSP1 Input and Output Messages section */
No Modification is needed on the current code

/* Active */
No Modification is needed on the current code

/* Release */
/* Correspond to step 17 and 18 in SSP1 Input and Output Messages section */
No Modification is needed on the current code
```

C.2 Terminating Switch (SSP_2)

```
/* Structure definition */

Struct PrivacyRequest
{
  OPName: String
  Arg[1]: char
  Arg[2]: char
  Arg[3]: char
} PriReq, Add;

Struct RemoteRequest
{
  OPName: String
  Arg[1]: char
  Arg[2]: char
}  RemReq

Struct RemoteRequestResponse
{
  OPName: String
  Arg[1]: List
}  RemReqRes, RemRes


/* Component Part with Class 1 Type */
TCAP.CP1 (Req)

/* ISDN Libraries */.
ISND.SETUP (Add)
```

```
ISDN.USERINFO (Info)

/* Analyzing the request from the SSP1 */
/* Correspond to step 1 in SSP2 Input and Output Messages section */
Add = { “Pri_Call_Req”, T_A, TLN_1, TLN_2 };
PriReq = TCAP.CP1 (Add)
IF          PriReq.OPName != Pri_Call_Req”
            RemRes.OPName = “Pri_Call_Res”
            RemRes.Arg[1] = DISC
            TCAP.CP1 (RemRes)

/* Initiating Called Telephone Authentication  & B-Channel Setup */
/* Correspond to step 1, 2, 3, 4, 5a and 5b in SSP2 Input and Output Messages section */
ELSE        ISDN.SETUP (SETUP)
            RemReqRes.OPName = “Tel_Val_Req”
            RemReqRes.Arg[1] = PriReq.Arg[3]
            RemRes = TCAP.CP1 (RemReqRes)
            RemReqRes = ISDN.USERINFO (RemRes)
            RemRes = TCAP.CP1 (RemReqRes)
            IF          RemRes.OPName = “Tel_Val_Res”  && RemRes.Arg[1] = DISC
                        ISDN.DISC (DISC)
                        RemRes.OPName = “Pri_Call_Res”
                        RemRes.Arg[1] = DISC
                        TCAP.CP1 (RemRes)

/* Initiating Called Subscriber Authentication */
/* Correspond to step 5c, 6, 7, 8, 9, 10, and 11 in SSP2 Input and Output Messages section */
            ELSE        RemReqRes.OPName = “Sub_Auth_Req”
                        RemReqRes.Arg[1] = T_A
                        RemRes = TCAP.CP1 (RemReqRes)
                        IF          RemRes.OPName = “Sub_PW_Req” && RemRes.Arg[1] = DISC
                                    ISDN.DISC (DISC)
                                    RemRes.OPName = “Pri_Call_Res”
                                    RemRes.Arg[1] = DISC
                                    TCAP.CP1 (RemRes)
                        ELSE        ISDN.CALPRC (CALPRC)
                                    ISDN.ALERT (ALERT)
                                    ISDN.CONN (CONN)
                                    Play PW Req Announcement
                                    Collect S2_RES
                                    RemReqRes.OPName = “Sub_PW_Res”
                                    RemReqRes.Arg[1] = S2_RES
                                    RemRes = TCAP.CP1 (RemReqRes)
                                    IF          RemRes.OPName = “Sub_Auth_Res” && RemRes.Arg[1] = DISC
                                                Play Pri_Call_Not_Approve_Announcement
                                                ISDN.DISC (DISC)
                                                RemRes.OPName = “Pri_Call_Res”
                                                RemRes.Arg[1] = DISC
                                                TCAP.CP1 (RemRes)

/* Distribution of Confirmation Ticket */
/* Correspond to step 12 and 13 in SSP2 Input and Output Messages section */
                                    ELSE        Play Pri_Call_Approve_Announcement
                                                RemReqRes.OPName = “Pri_Call_Confirm”
                                                RemReqRes.Arg[1] = T_C
                                                ISDN.USER INFO (RemReqRes)
                                                RemReqRes.OPName = “Pri_Call_Res”
                                                RemReqRes.Arg[1] = T_C
                                                RemRes = TCAP.CP1 (RemReqRes)
                                                Delay()

/* Distribution of Encryption Key */
/* Correspond to step 14 in SSP2 Input and Output Messages section */
                                                KeyDist = TCAP.CP1 (RemKeyDist)
                                                IF          KeyDist.OPName = “Key_Dist_Req” && KeyDist.Arg[1] != Null
                                                            ISDN.USER INFO (KeyDist)
END IF

/* Trunk Setup */
/* Correspond to step 15 in SSP2 Input and Output Messages section */
```

No Modification is needed on the current code

/* Active */
No Modification is needed on the current code

/* Release */
/* Correspond to step 16 and 17 in SSP2 Input and Output Messages section */
No Modification is needed on the current code

C.3 AC Process Provides Support to SSP_1

```
/* Structure definition */

Struct CGSPasswordResponse
{
  Arg[1]: char
  Arg[2]: char
  Arg[3]: char
  Arg[4]: char
  Arg[5]: char
  Arg[6]: char
} S1_Res

Struct ApprovalTicketResponse
{
  Arg[1]: char
  Arg[2]: char
  Arg[3]: char
  Arg[4]: char
  Arg[5]: char
  } ApprovalTicket

Struct RemoteRequest
{
  OPName: String
  Arg[1]: char
  Arg[2]: char
}  RemReq, Req

Struct RemoteRequestResponse
{
  OPName: String
  Arg[1]: char
} RemReqRes, RemRes

Struct RemoteRandomResponse
{
  Arg[1]: char
  Arg[2]: char
} RemRandRes, RemTelRes, DBTelInfo

/* Component Part with Class 1 Type */
TCAP.CP1 (Req)

/* Calling Telephone Authentication  */
/* Correspond to step 1 and 2 in AC_SSP1 Input and Output Messages section */
RemReqRes = TCAP.CP1 (Req)
IF          RemReqRes.OPName != “Tel_Val_Req”
            RemRes.OPName = “Tel_Val_Res”
            RemRes.Arg[1] = DISC
            TCAP.CP1 (RemRes)
ELSE        IF          RemReqRes.Arg[1] = NULL
                        RemRes.OPName = “Tel_Val_Res”
                        RemRes.Arg[1] = DISC
                        TCAP.CP1 (RemRes)
            ELSE        R_AC = Generate RandomNumber
                        RemRes.OPName = “Tel_Auth_Req”
                        RemRes.Arg[1] = Encrypt_AC_Private_Key (R_AC)
                        RemReqRes = TCAP.CP1 (RemRes)
```

```
                    IF      RemRegRes.OPName != “Tel_Auth_Res”
                            RemRes.OPName = “Tel_Val_Res”
                            RemRes.Arg[1] = DISC
                            TCAP.CP1 (RemRes)
                    ELSE    IF      RemReqRes.Arg[1] = Null
                                    RemRes.OPName = “Tel_Val_Res”
                                    RemRes.Arg[1] = DISC
                                    TCAP.CP1 (RemRes)
                            ELSE    RemRandRes = Encrypt_Tel_Public_key (RemReqRes.Arg[1])
                                    R_Tel = RemRandRes.Arg[2]
                                    RemTelRes = Decrypt_AC_Private_Key (RemRandRes.Arg[1])
                                    DBTelInfo = Retrieve Telphone Information from the database
                                    IF      R_AC = R_Tel && RemTelRes = DBTelInfo
                                            RemRes.OPName = “Tel_Val_Res”
                                            RemRes.Arg[1] = CONT
                                            TCAP.CP1 (RemRes)
                                    ELSE    RemRes.OPName = “Tel_Val_Res”
                                            RemRes.Arg[1] = DISC
                                            TCAP.CP1 (RemRes)

/* Calling Subscriber Authentication */
/* Correspond to step 3, 4, and 4a in AC_SSP1 Input and Output Messages section */
IF          RemReqRes.OPName != “Sub_Auth_Req”
            RemRes.OPName = “Sub_Auth_Res”
            RemRes.Arg[1] = DISC
            TCAP.CP1 (RemRes)
ELSE        IF      RemReqRes.Arg[1] != S_1
                    RemRes.OPName = “Sub_Auth_Res”
                    RemRes.Arg[1] = DISC
                    TCAP.CP1 (RemRes)
            ELSE    RemRes.OPName = “Sub_PW_Req”
                    RemRes.Arg[1] = STR-I
                    RemReqRes = TCAP.CP1 (RemRes)
                    IF      RemReqRes.OPName != “Sub_PW_Res”
                            RemRes.OPName = “Sub_Auth_Res”
                            RemRes.Arg[1] = DISC
                            TCAP.CP1 (RemRes)
                    ELSE    IF      RemReqRes.Arg[1] = NULL
                                    RemRes.OPName = “Sub_Auth_Res”
                                    RemRes.Arg[1] = DISC
                                    TCAP.CP1 (RemRes)
                            ELSE    S1_RES = Decrypt_AC_Private_Key (RemReqRes.Arg[1])
                                    Authenticate S1_RES.Arg[1] = S1_ID
                                    Validate other arguments in S1_RES

/* Distribution of Approval */
/* Correspond to step 4b and 4c in AC_SSP1 Input and Output Messages section */
                                    IF      Authentication && Validation are OK
                                            ApprovalTicket.Arg[1] = S1_ID
                                            ApprovalTicket.Arg[2] = S2_ID
                                            ApprovalTicket.Arg[3] = TLN_1
                                            ApprovalTicket.Arg[4] = TLN_2
                                            ApprovalTicket.Arg[5] = time + valid time
                                            T_A = Encrypt_AC_Private_Key (ApprovalTicket)
                                            RemRes.OPName = “Sub_Auth_Res”
                                            RemRes.Arg[1] = T_A
                                            TCAP.CP1 (RemRes)
                                    ELSE    RemRes.OPName = “Sub_Auth_Res”
                                            RemRes.Arg[1] = DISC
                                            TCAP.CP1 (RemRes)
ENDIF
```

C.4 AC Process Provides Support to SSP_2

```
/* Structure definition */

Struct CDSPasswordResponse
{
  Arg[1]: char
  Arg[2]: char
```

```
  Arg[3]: char
  Arg[4]: char
} S2_RES

Struct ConfirmationTicketResponse
{
  Arg[1]: char
  Arg[2]: char
  Arg[3]: char
  Arg[4]: char
  Arg[5]: char
} ConfirmationTicket

Struct RemoteRequest
{
  Arg[1]: char
  Arg[2]: char
 }  RemReq

Struct RemoteRequestResponse
{
  OPName: String
  Arg[1]: char
} RemReqRes, RemRes

Struct RemoteRandomResponse
{
  Arg[1]: char
  Arg[2]: char
} RemRandRes, RemTelRes, DBTelInfo

/* Component Part with Class 1 and Class 2 Types */
TCAP.CP1 (Req)

/* Called Telephone Authentication */
/* Correspond to step 1 and 2 in AC_SSP2 Input and Output Messages section */
RemReqRes = TCAP.CP1 (Req)
IF          RemReqRes.OPName != “Tel_Val_Req”
            RemRes.OPName = “Tel_Val_Res”
            RemRes.Arg[1] = DISC
            TCAP.CP1 (RemRes)
ELSE        IF          RemReqRes.Arg[1] = NULL
                        RemRes.OPName = “Tel_Val_Res”
                        RemRes.Arg[1] = DISC
                        TCAP.CP1 (RemRes)
            ELSE        R_AC = Generate RandomNumber
                        RemRes.OPName = “Tel_Auth_Req”
                        RemRes.Arg[1] = Encrypt_AC_Private_Key (R_AC)
                        RemReqRes = TCAP.CP1 (RemRes)
                        IF          RemRegRes.OPName != “Tel_Auth_Res”
                                    RemRes.OPName = “Tel_Val_Res”
                                    RemRes.Arg[1] = DISC
                                    TCAP.CP1 (RemRes)
                        ELSE        IF          RemReqRes.Arg[1] = Null
                                                RemRes.OPName = “Tel_Val_Res”
                                                RemRes.Arg[1] = DISC
                                                TCAP.CP1 (RemRes)
                                    ELSE        RemRandRes = Encrypt_Tel_Public_key (RemReqRes.Arg[1])
                                                RemTelRes = Decrypt_AC_Private_Key (RemRandRes.Arg[1])
                                                DBTelInfo = Retrieve Telphone Information from the database
                                                R_Tel = RemRandRes.Arg[2]
                                                IF          R_AC = R_Tel && RemTelRes = DBTelInfo
                                                            RemRes.OPName = “Tel_Val_Res”
                                                            RemRes.Arg[1] = CONT
                                                            TCAP.CP1 (RemRes)
                                                ELSE        RemRes.OPName = “Tel_Val_Res”
                                                            RemRes.Arg[1] = DISC
                                                            TCAP.CP1 (RemRes)

/* Called Subscriber Authentication */
```

```
/* Correspond to step 3, 4a, and 4b in AC_SSP2 Input and Output Messages section */
IF          RemReqRes.OPName != “Sub_Auth_Req”
            RemRes.OPName = “Sub_Auth_Res”
            RemRes.Arg[1] = DISC
            TCAP.CP1 (RemRes)
ELSE        IF          RemReqRes.Arg[1] = NULL
                        RemRes.OPName = “Sub_Auth_Res”
                        RemRes.Arg[1] = DISC
                        TCAP.CP1 (RemRes)
            ELSE        Validate T_A
                        IF          T_A is not Validated
                                    RemRes.OPName = “Sub_Auth_Res”
                                    RemRes.Arg[1] = DISC
                                    TCAP.CP1 (RemRes)
                         ELSE       RemRes.OPName = “Sub_PW_Req”
                                    RemRes.Arg[1] = STR-I
                                    RemReqRes = TCAP.CP1 (RemRes)
                                    IF          RemReqRes.OPName != “Sub_PW_Res”
                                                RemRes.OPName = “Sub_Auth_Res”
                                                RemRes.Arg[1] = DISC
                                                TCAP.CP1 (RemRes)
                                       ELSE IF          RemReqRes.Arg[1] = NULL
                                                        RemRes.OPName = “Sub_Auth_Res”
                                                        RemRes.Arg[1] = DISC
                                                        TCAP.CP1 (RemRes)
                                                ELSE    S2_RES = Decrypt_AC_Private_Key (RemReqRes.Arg[1])
                                                        Authenticate S2_RES.Arg[1] = S2_ID
                                                        Validate other arguments in S2_RES
/* Distribution of Confirmation Ticket */
/* Correspond to step 4c and 4d in AC_SSP2 Input and Output Messages section */
                                                IF          Authentication && Validation are OK
                                                            ConfirmationTicket.Arg[1] = S1_ID
                                                            ConfirmationTicket.Arg[2] = S2_ID
                                                            ConfirmationTicket.Arg[3] = TLN_1
                                                            ConfirmationTicket.Arg[4] = TLN_2
                                                            ConfirmationTicket.Arg[5] = time + valid time
                                                            T_C = Encrypt_AC_Private_Key (ConfirmationTicket)
                                                            RemReqRes.OPName = “Sub_Auth_Res”
                                                            RemReqRes.Arg[1] = T_C
                                                            RemRes = TCAP.CP1 (RemReqRes)
                                                ELSE        RemRes.OPName = “Sub_Auth_Res”
                                                            RemRes.Arg[1] = DISC
                                                            TCAP.CP1 (RemRes)

/* Generation of Encryption Key */
/* Correspond to step 5 in AC_SSP2 Input and Output Messages section */
                                                            K_E = Generate_Encryption_Key (K)
                                                            RemReq.Arg[1] = K_E
                                                            RemReq.Arg[2] = time + valid time
                                                            SigKey = Encrypt_AC_Private _Key (RemReq)

/* Distribution of Encryption Key */
K_ET1 = Encrypt_Tel_1_Pub_Key (SigKey)
/* Sending Encryption Key to Telephone_1 */
/* Correspond to step 5a in AC_SSP2 Input and Output Messages section */
RemReqRes.OPName = “Key_Dist_Req”
RemReqRes.Arg[1] = K_ET1
RemRes = TCAP.CP1 (RemReqRes)

/* Sending Encryption Key to Telephone_1 */
/* Correspond to step 5b in AC_SSP2 Input and Output Messages section */
K_ET2 = Encrypt_Tel_2_Pub_Key (SigKey)
RemReqRes.OPName = “Key_Dist_Req”
RemReqRes.Arg[1] = K_ET2
RemRes = TCAP.CP1 (RemReqRes)
ENDIF
```

C.5 Calling Telephone (T_1)

```
/* Structure definition */
```

```
Struct RemoteRequest
{
   OPName: String
   Arg[1]: char
   Arg[2]: char
   Arg[3]: char
}  SetupReq

Struct RemoteRequestResponse
{
   OPName: String
   Arg[1]: char
} TelAuthReq, TelAuthRes, ApprvlTikt, ConfmTikt, KeyDist

Struct Parameters
{
   Arg[1]: char
   Arg[2]: char
} EntNum, K_ET1

/* ISDN Libraries */
ISDN.USER.INFOR ()
ISDN.USER.INFO ()
ISDN.ALERT ()
ISDN.CALPRC ()
ISDN.CONN ()
ISDN.CONACK ()
ISDN.STOPALERT ()
ISDN.DISC
ISDN.RLSE
ISDN.RLCOM

/* Defining Constant Parameters */
Define TLN_1 = Tel_Number
Define TDN_1 = Device_Number
Define Pri_Code = Privacy_Code


/* On Hook */
/* Subscriber enters called and press “Enter”  */

/* Privacy connection request */
/* Correspond to step 1 and 2 in T1 Input and Output Messages section */
EntNum = scan digits
IF          EntNum.Arg[1] = Pri_Cade
            TLN_2 = EntNum.Arg[2]
            SetupReq.OPName = “SETUP”
            SetupReq.t.Arg[1] = EntNum.Arg[1]
            SetupReq.Arg[2] = TLN_1
            SetupReq.Arg[3] = TLN_2
            ISDN.SETUP (SetupReq)

/* Telephone Authentication Response */
/* Correspond to step 4, 5 and 6 in T1 Input and Output Messages section */
            TelAuthReq = ISDN.USER.INFOR (RemReq)
            IF          TelAuthReq.OPName = “Tel_Auth_Req” && TelAuthReq.Arg[1] != NULL
                        R_AC = Decrypt_AC_Public_Key (TelAuthReq.Arg[1])
                        Enc_Tel_Info = Encrypt_AC_Public_Key (TLN_1, TDN_1)
                        TelAuthRes.OPName = “Tel_Auth_Res”
                        TelAuthRes.Arg[1] = Encrypt_AC_Public_Key (R_AC, Enc_Tel_Info)
                        ISDN.USER.INFO (TelAuthRes)


/* Connection Setup for the B-Channel */
/* Correspond to step 8, 9 and 10 in T1 Input and Output Messages section */
                        ISDN.CALPRC
                        ISDN.ALERT
                        ISDN.CONN
                        ISDN.STOPALERT
```

```
                    ISDN.CONACK

/* Subscriber Authentication Response */
/* Correspond to step 11 in T1 Input and Output Messages section */
                    PW_INFO = Collect digits
                    S1_RES = Encrypt_AC_Public_Key (PW_INFO)
                     Send S1_RES  to SSP via B-Channel

/* Storing Approval Ticket */
/* Correspond to step 13 in T1 Input and Output Messages section */
                    AppvlTikt = ISDN.USER.INFO (RemReqRes)
                    IF      AppvlTikt.OPName = "Pri_Call_Approved" && AppvlTikt.Arg[1] != NULL
                            T_A = AppvlTikt.Arg[1]

/* Storing Confirmation Ticket  */
/* Correspond to step 15 in T1 Input and Output Messages section */
                            ConfmTikt = ISDN.USER.INFO (RemReqRes)
                            IF      ConfmTikt.OPName = "Pri_Call_Confirm" && ConfmTikt.Arg[1] != NULL
                                    T_C = ConfmTikt.Arg[1]

/* Deleting Approval Ticket */
/* Correspond to step 15 in T1 Input and Output Messages section */
                                    Delete T_A
                                    Delay()

/* Storing Encryption Key */
/* Correspond to step 16 in T1 Input and Output Messages section */
                                    KeyDist  = ISDN.USER.INFO (KeyDist)
                                    IF      KeyDist .OPName = "Key_Dist_Req" && KeyDist.Arg[1] != NULL
                                            K_ET1 = Decrypt_AC_Public_Key (KeyDist.Arg[1])
                                            IF      (CurrentTime – 60 sec) < K_ET1.Arg[2]  < CurrentTime
                                                    K_E = K_ET1.Arg[1]
                                                    t_E = K_ET1.Arg[2]

/* Encrypting and Decrypting Voice Signals */
/* Correspond to step 17 in T1 Input and Output Messages section */
                                                            Digitized_Voice = Collect digits
                                                            Encrypt_Voice = K_E (Digitized_Voice)
                                                            Send Encrypt_Voice to T_2 via B-Channel
                                                            Decrypt_Voice = Receive Digits from T_2
                                                            Digitized_Voice = K_E (Decrypt_Voice)

/* Requesting a new Encryption Key (Optional) */


/* On Hook */
/* Correspond to step 18 in T1 Input and Output Messages section */
                                                            ISDN.DISC
                                                            ISDN.RLSE
                                                            ISDN.RLCOM


/* Deleting Confirm Ticket & Encryption Key */
/* Correspond to step 19 in T1 Input and Output Messages section */
                                                            Delete T_C
                                                            Delete K_E
                                                            Delete t_E

/* Correspond to step 3, 7, 12 and 14 in T1 Input and Output Messages section */
ELSE    ISDN.DISC
        ISDN.RLSE
        ISDN.RLCOM
END IF
```

C.6 Calling Telephone (T_2)

```
/* Structure definition */

Struct RemoteRequest
{
```

```
   OPName: String
   Arg[1]: char
   Arg[2]: char
   Arg[3]: char
}  PriReq

Struct RemoteRequestResponse
{
   OPName: String
   Arg[1]: char
} TelAuthReq, TelAuthRes,  ConfmTikt, KeyDist

Struct Parameters
{
   Arg[1]: char
   Arg[2]: char
} K_ET2

/* ISDN Libraries */
ISDN.USER.INFOR ()
ISDN.USER.INFO ()
ISDN.ALERT ()
ISDN.STOPALERT ()
ISDN.CONN ()
ISDN.CONACK
ISDN.DISC
ISDN.RLSE
ISDN.RLCOM

/* Defining Constant Parameters */
Define TLN_2 = Tel_Number
Define TDN_2 = Device_Number
Define Pri_Code = Privacy_Code


/* Privacy connection request */
/* Correspond to step 1 in T2 Input and Output Messages section */
PriReq = ISDN.SETUP (RemReqRes)
IF          PriReq.OPName = “SETUP” && PriReq.Arg[1] = Pri_Code

/* Telephone Authentication Response */
/* Correspond to step 3, 4 and 5 in T2 Input and Output Messages section */
            TelAuthReq = ISDN.USER.INFOR (RemReq)
            IF          TelAuthReq.OPName = “Tel_Auth_Req” && TelAuthReq.Arg[1] != NULL
                        R_AC = Decrypt_AC_Public_Key (TelAuthReq.Arg[1])
                        Enc_Tel_Info = Encrypt_AC_Public_Key (TLN_2, TDN_2)
                        TelAuthRes.OPName = “Tel_Auth_Res”
                        TelAuthRes.Arg[1] = Encrypt_AC_Public_Key (R_AC, Enc_Tel_Info)
                        ISDN.USER.INFO (TelAuthRes)

/* On Hook  and Connection Setup for the B-Channel */
/* Correspond to step 7 in T2 Input and Output Messages section */
                        ISDN.CALPRC
                        ISDN.ALERT

/* On Hook */
/* Correspond to step 7 in T2 Input and Output Messages section */
                        ISDN.CONN
                        ISDN.CONACK
                        ISDN.STOPALERT

/* Subscriber Authentication Response */
/* Correspond to step 9 in T2 Input and Output Messages section */
                        PW_INFO = Collect digits
                        S2_RES = Encrypt_AC_Public_Key (PW_INFO)
                         Send S2_RES  to SSP via B-Channel

/* Storing Confirmation Ticket  */
/* Correspond to step 11 in T2 Input and Output Messages section */
                        ConfmTikt = ISDN.USER.INFO (RemReqRes)
```

```
                    IF      ConfmTikt.OPName = “Pri_Call_Confirm” && ConfmTikt.Arg[1] != NULL
                            T_C = ConfmTikt.Arg[1]
                            Delay()

/* Storing Encryption Key */
/* Correspond to step 12 in T2 Input and Output Messages section */
                            KeyDist  = ISDN.USER.INFO (KeyDist)
                            IF      KeyDist .OPName = “Key_Dist_Req” && KeyDist.Arg[1] != NULL
                                    K_ET2 = Decrypt_AC_Public_Key (KeyDist.Arg[1])
                                    IF      (CurrentTime – 60 sec) < K_ET2.Arg[2]  < CurrentTime
                                            K_E = K_ET2.Arg[1]
                                            t_E = K_ET2.Arg[2]

/* Encrypting and Decrypting Voice Signals */
/* Correspond to step 13 in T2 Input and Output Messages section */
                                                    Digitized_Voice = Collect digits
                                                    Encrypt_Voice = K_E (Digitized_Voice)
                                                    Send Encrypt_Voice to T_1 via B-Channel
                                                    Decrypt_Voice = Receive Digits from T_1
                                                    Digitized_Voice = K_E (Decrypt_Voice)

/* Requesting a new Encryption Key (Optional) */


/* On Hook */
/* Correspond to step 14 and 15 in T2 Input and Output Messages section */
                                                    ISDN.DISC
                                                    ISDN.RLSE
                                                    ISDN.RLCOM


/* Deleting Confirm Ticket & Encryption Key */
/* Correspond to step 15 in T2 Input and Output Messages section */
                                                    Delete T_C
                                                    Delete K_E
                                                    Delete t_E

/* Correspond to step 2, 6 and 10 in T2 Input and Output Messages section */
ELSE        ISDN.DISC
            ISDN.RLSE
            ISDN.RLCOM
END IF
```