SECURE BRIDGES: A MEANS TO CONDUCT SECURE TELECONFERENCES OVER PUBLIC TELEPHONES*

Inja Youn and Duminda Wijesekera

Abstract: Many organizations carry out their confidential business over teleconferences with the expectation that only the declared participants are privy to the conversation. But, because public telephones do not encrypt voice, such teleconferences are vulnerable to eavesdropping. As a remedy, we offer secure bridges consisting of architectures and algorithms for eavesdropping resistant teleconferencing as a service over public switched telephone network (PSTN) with minimal additions. Our proposal consists of appropriately placing servers to authenticate telephone equipment and subscribers of the service, and certificate authorities to cross-certify them over 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 suites and the D1 channel of the digital subscriber line (DSL) connecting telephone equipment to the SS7 grid. Using published delays our algorithms take about 4.25 minutes to setup a 10-person teleconference including soliciting and verifying user IDs and passwords.

1. INTRODUCTION

Despite the advent of Internet based net meetings and IP telephony, many organizations still use traditional teleconferences to conduct their sensitive businesses. Consequently, participants of such conferences expect their conversations to be private. Nevertheless, being implemented as intelligent applications without any security over public switched telephone networks

* Partly supported by NSF under grant CCR-0113515 and CSIS at George Mason University

(PSTN), these teleconferences are susceptible to eavesdropping. Software and hardware modules that provide to days teleconferences over the PSTN are known as *bridges,* and specified in ITU-T recommendations Q.954 [4], Q.734 [5] and Q.84 [6]. None of these or existing services provide voice confidentiality as a service over public telephony. As a remedy, we propose *secure bridges* that provide authentication and voice encryption within multi-party calls by minimally changing existing bridges, thereby expanding quoted ITU-T recommendations.

As proposed, a teleconference begins by an initiator (hereafter referred to as the *call master*) dialing a special key sequence, say ** on a telephone set equipped with encryption capabilities to obtain the proposed service. In response, the *interactive voice response (IVR)* comes over the call master's phone and authenticates the caller. Consequent to proper authentication, the IVR comes over again and requests the telephone numbers and identities of other participants (referred to as *slave conferees*). Authenticated participants and their equipment participate in the conference using a one-time voice encryption key. As an enhancement, we allow the call master to add or drop any participants at will, and with the call master's permission, other callers may join an ongoing teleconference. In order to prevent eavesdropping, any changes in the conferee group structure trigger a new encryption key where the *hold services* provided by the PSTN to momentarily suspend the voice stream is used to refresh the encryption key. The teleconference ends when the call master or the last slave conferee hangs up.

The one-time encryption key prevents replay attacks. We also authenticate conferees and their telephones used for voice privacy services. Our authentication protocols use public key cryptography with the use of authentication centers. We show how to integrate our service on PSTN

2. RELATED WORK

The multiparty communication services over the PSTN are standardized in the ITU-T recommendations Q.84, Q.734 and Q.954. Q.84 addresses the general structure, the concept of a *bridge* and the basic calling procedures of adding, dropping, isolating and reattaching. Q.734 describes details of multiparty supplementary services in the context of using ISDN within the SS7 network. Q.954 describes how the conference calls should perform at the user-network interface.

Most multi-party systems do not provide authentication, authorization and non-repudiation but only the confidentiality. The *secure telephone unit third generation (STU III)* is a system designed to work as dedicated pairs through PSTN using symmetric keys stored in handheld telephone units for

voice encryption messages. *SecureLogix's TeleVPN®* is another system that uses 3DES to provide voice encryption between two private branch exchanges (PBXs), but not end-to-end privacy. *Wireless networks* use symmetric keys for encryption, but not provide end-to end voice encryption. Sharif et al. [2] describe a system with end-to-end voice privacy but they do not address complications beyond two-party communications.

Sailer [12] enhances network service interfaces on standardized security services to enable open security, but is tangential to voice privacy. A new application level protocol referred to as the *security services application part (SecAP)* was envisioned to fulfill the need of additional signaling protocols between core network functions and specialized security services functions. Lorenz [10] analyzes SS7 vulnerabilities and presents attack taxonomy.

3. PROPOSED SECURITY ARCHICTURE

Figure 1 (A) shows the proposed architecture using a master bridge that communicates with the call master and slave bridges that manage slave conferees. Every secure bridge (i.e. a traditional bridge with an embedded AC) has its own AC for authentication and key distribution. The main components of the architecture are shown in Figure 1(B).

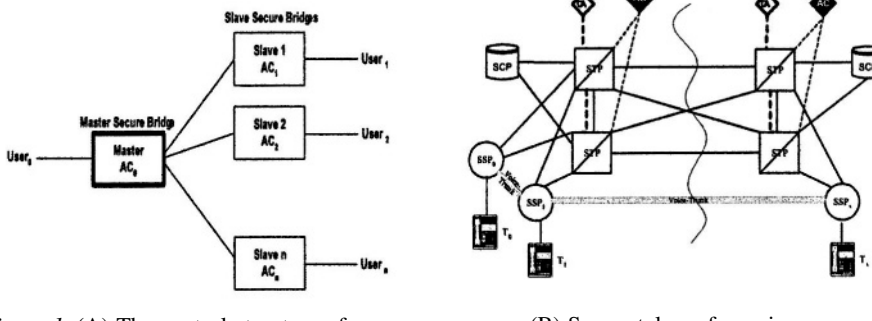


Figure 1. (A) The control structure of secure teleconferencing (B) Secure teleconferencing architecture

- An *authentication center* (AC) authenticates users and their equipment, manages symmetric keys, and issues tickets.
- A *certificate authority* (CA) generates and manages credentials, keys and certificate revocation lists (CRLs). Both AC and CA are *application service elements* (ASE) of the PSTN.
- A *service switching point* (SSP) is a transit exchange providing *advanced intelligent networks* (AIN). SSPs communicate with ACs, CAs, interpret messages and initiate disconnection procedures on request.

- A *service control point* (SCP) is a database with routing tables, billing information etc. For example, *line information databases* (LIDB).
- A *signal transfer point* (STP) is a switch that routes messages.

User telephones are expected to have cryptographic capabilities using symmetric and public keys. Our authentication algorithms use *telephone line numbers* (TLN), *telephone device numbers* (TDN), the public/private keys and the timestamps (to prevent replay attacks) as parameters. All participating CAs are expected to have mutual trust relationships. Our protocols use the *digital subscriber signaling system no 1 (DSS1)* to communicate between the telephones and the local SSPs. *ISDN user part (ISUP)* for communication between SSPs and TCAP for transactions between SSPs, ACs, CAs and SCPs

3.1 Terminology

This section describes the notations used throughout this paper.

- **$User_0$, call master:** The user requesting and controlling the conference.
- **Conferee, $User_1$, ...,$User_n$:** Other conference participants.
- **T_i:** The telephone set of $User_i$
- **Begin:** An action initiating the teleconferencing by $User_0$.
- **Local Exchanges, signal service points: SSP_0, SSP_1..... SSP_n.**
- **Transit Exchange (TE):** One that connect SSPs to national or international exchanges.
- **Secure Bridge (SB):** A bridge with authentication capabilities.
- **Master Secure Bridge (MSB):** The controlling bridge of a teleconference.
- **Slave Bridge (SB):** A non-controlling bridge of a teleconference.
- **En-bloc signaling:** A (IAM) signaling sequence sending an entire telephone number.
- ** **(double Asterisk):** The access code. ** for secure conference calls.
- **CR_0, CR_0*,CR_1...CR_X:** Call Reference [4]. Indicates the referee of a call.
- **TDN:** Telephone Device Number. TND of $User_i$ is TDN_i,
- **TLN:** Telephone Line Number. TLN of $User_i$, $User_1$, $User_2$ ... are denoted by TLN_i.
- **BR:** Bridge. A device connecting network segments for multi part service.
- **MSB:** master secure bridge.
- **SETUP:** Q.931
- **FAC:** DSS1 (Q.932) messages invoking a facility during an active call state.
- **FIE|:** Facility Information Element – data that can be included in an ISDN message to invoke supplementary services.
- **NIE:** Notify Information Element -data used to notify an action within the network.
- **ID_i:** The identifier of $User_i$, **PWD_i:** Password of $User_i$
- **BeginConf.** An FIE included a SETUP message, requesting to begin a secure conference.
- **IAM:** Initial Address Message
- **CALLPRC:** Call proceeding in Q.931.
- **BEGIN:** Begin a TCAP transaction. **END:** End a TCAP transaction.

- **Invoke:** an application (component parting TCAP).
- **CONTINUE:** Continue a TCAP transaction, **Invoke:** Request Result (component part).
- **RR:** Request Results from a TCAP transaction.
- **ALERT:** An ISUP message (DOING WHAT?).
- **ACM;** address completion message. **ANM***:* answer message.
- **CPG:** Call progress message - reports a call setup event.
- **CONN:** Connect. Q.931.
- **Notify:** ISUP (DOING WHAT?)
- **K / K*:** Public / Private Key pairs. K_i / K^*_i belongs to $User_i$.
- K_E, **KOLD,** K_{NEW}**:** initial, old and new shared symmetric voice encryption keys.
- R_i, R^*_i, R'_i**:** Random Number, used for authentication between T_i and AC_i, AC_{br} and T_i, and transmits tickets between AC_{br} and AC_i.
- t_i**:** Timestamps, M1: Messages
- **beginSecureConf-Inv:** A facility message invoking a conference call.
- **userAuth-Req:** User authentication request sent by the MSB to T_0
- **userAuth-RR:** T_0 sends back SSP_0 the result of the request.
- **RemoteAuth-Req:** The slave bridge SSP_s send this message to AC_i, verifying $User_0$.
- **RemoteAuth-RR:** AC_i replies back to SSP_1 containing the public key of the SSB.
- **TelAuth-Req:** SSP_i in slave bridge request T_1 for device authentication.
- **TelAuth-RR:** T_1 replies back.
- **CPG (Key Distribution, Msg)||.** CPG containing the key distribution message.
- **KeyDis:** Key distribution message with the voice encryption key.
- **CPG (Key Distribution-ACK):** CPG containing a key distribution acknowledgement.
- **keyDist-ACK:** Key distribution acknowledgement.
- **Play IVR:** Request to play an IVR.

4. PROTOCOLS

Eavesdropping proof teleconferencing over PSTN require eight protocols categorized as originating and joining, leaving a conference and ending the conference. The first category has three protocols (1) setting up the conference, (2) a new conferee initiated joining and (3) adding conferee by the call master's invitation. The second category consists of (3) dropping a conferee due to call master's request, (4) a slave conferee hanging up, (5) conferee initiated leaving. The third category consists of (6) call teardown when the call master hangs up, (7) when the last conferee hangs up and (8) when all slave conferees either refuse or fail admission criteria incoming connections. Due to space limitations, we describe only the call setup protocol in detail. Salient points of our protocols are that (1) all conferees and their equipment have to be authenticated, (2) all voice communications are encrypted by one key (3) any change of the conferee group triggers a

change of the voice encryption key and (4) the call master retains the right to conferee membership changes and conference termination.

4.1 Protocol 1: Conference Initiation

The call setup protocol consists of three basic phases. In the first phase a customer that subscribes to the proposed service dials an access code (say **) and is prompted for a (ID, password) pair. Those and the telephone equipment information are used to authenticate the call master. Then, the call master is guided by an interactive voice response (IVR) to enter other (say slave) conferees telephone numbers. Then they are contacted and authenticated (with their equipment) by IVR guidance. During the next phase, those authenticated conferees are distributed a common voice encryption key. Thereafter voice circuits are reserved for the conference and finally the conference begins. A step-by-step description follows.

Protocol 1: [conference initiation]
(A) Call Initiation:

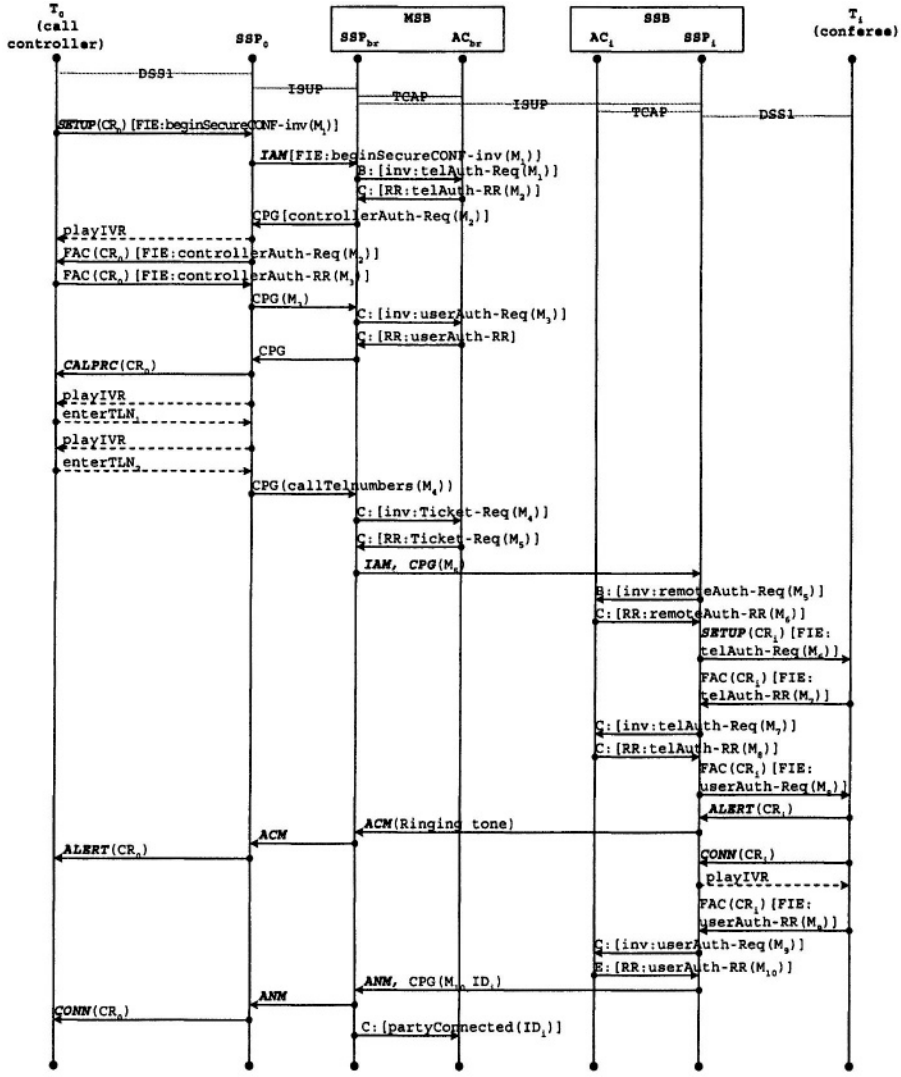
1.1 $[T_0]$ The call master dials the secure teleconferencing access code (**).

1.2 $[T_0 \rightarrow SSP_0]$ T_0 sends a SETUP message $M_1 = K_{br}[K^*_0[TLN_0, TDN_0, t_0]]$ in a FIE to SSP_0 initiating a conference. SSP_0 allocates MSB resources, and if unsuccessful, sends a RELEASE message clearing the connection that is acknowledged by a RELCOM from T_0.

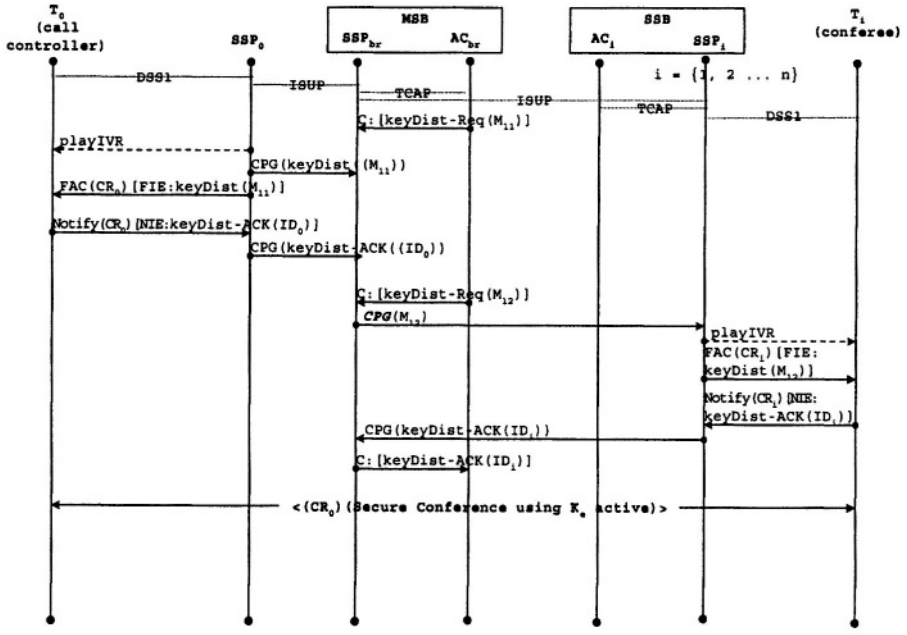
1.3 $[SSP_0 \rightarrow SSP_{br} \rightarrow AC_{br}]$ SSP_0 forwards M_1 to SSP_{br}, in an IAM message. SSP_{br} starts a TCAP transaction with AC_{br} and sends M_1 to AC_{br} together with an authentication request. AC_{br} checks the authenticity of T_0 by decrypting the message with K^*_{br} and verifying the signature with K_0. After extracting contents TLN_0, TDN_0 and t_0, AC_{br} checks t_0, and TDN_0 for freshness and permissions. If either fails, AC_{br} signals connection clearance by ending the transaction with SSP_{br}. Otherwise the AC_{br} initializes the *party_list* with $[ID_0]$ and continue.

B. Authentication and Authorization

1.4 $[AC_{br} \rightarrow SSP_{br} \rightarrow SSP_0 \rightarrow T_0]$ MSB sends a FACILITY message to T_0 containing a FIE request for user authentication together with a random number R^*_0 and a timestamp t_1 in M_2: $= K^*_{br}[R^*_0, t_1]$.



Continued on the next page…



M1 = Kbr[K*0[TLN0,TDN0,t0]]
M2 = K*br[R*0,t1]
M3 = Kbr[ID0,PWD0,R*0,t2]
M4= (nc, Kbr(TLN1, R*0)…, Kbr(TLNnc , R*0))
M5 = K*br[ID0, ID1 … IDnc, R1, t3]
M6 = K*aci[ID0,Kbr,R'i,t4]
M7 = Kaci[K*i[TLNi,TDNi,Ri,t5]]
M8 = K*br[R*i,t6]
M9 = K*aci[IDi, PWDi, R*i,t7]
M10 = K*aci[IDi,R'i,t8,Ki]
M11 = K0[K*br[KE, R*0, t9]]
M12 = Ki[K*br[KE, R*i, t9]]

Figure 2. Beginning the conference

1.5 $[T_0 \rightarrow SSP_0 \rightarrow SSP_{br} \rightarrow AC_{br} \rightarrow SSP_{br}]$ SSP_{br} prompts $User_0$ to enter (ID_0, PWD_0) pair. T_0 send the results to AC_{br} in a FACILITY message $M_3 = K_{br}[ID_0, PWD_0, R^*_0, t_2]$ encrypted with the bridge's public key K_{br}. Upon receipt, if authentication fails, AC_{br} clears the connection to T_0 by ending the transaction with SSP_{br} by sendin an error message, and continues otherwise. The SSP_{br} acts accordingly.

1.6 $[SSP_{br} \rightarrow SSP_0 \rightarrow T_0]$ SSP_{br} signals SSP_0 to send the *Call proceeding* message to $User_0$'s, and SSP_0 obliges and plays the IVR messages *please enter the number of conferees followed by the # sign* ($User_0$ enters $nc < 30$ numbers). Suppose only $n \leq nc$ joins the conference. Then IVR solicits

these number by playing *please enter the first/next conferee's telephone number followed by the # sign* etc. Assume $User_0$ obliges.

1.7 $[T_0 \rightarrow SSP_0 \rightarrow SSP_{br} \rightarrow AC_{br}]$ SSP_0 collects all the numbers and sends them to AC_{br} in message M_4= (nc, $K_{br}[TLN_1, R^*_0)$..., $K_{br}[TLN_{nc}, R^*_0]$). When the AC_{br} receives M_4, decrypts the numbers, and checks the random number for freshness. If authentication fails, the AC_{br} ends the transaction with the SSP_{br} with an error message and continues otherwise.

1.4 Simultaneously execute steps 1 through 10 for each i = 1 to *nc*.

1. $[AC_{br} \rightarrow SSP_{br} \rightarrow SSP_i]$ AC_{br} signals SSP_{br} to send an IAM seizing a trunk between the secure bridge for U_i and SSP_i to establish a bidirectional circuit between them, followed by a CPG message with $M_5 = K^*_{br}[ID_0, ID_1 \ldots ID_{nc}, R_1, t_3]$ signed by the bridge as a parameter. M_5 certifies that U_0 initiated the conference.
2. $[SSP_i \rightarrow AC_i]$ SSP_i forwards M_5 to AC_i for authentication. If fails, AC_i signals SSP_i to drop $User_i$.
3. $[AC_i \rightarrow SSP_i \rightarrow T_i \rightarrow SSP_i]$ If authentication succeeds, AC_i sends the result to SSP_i in the TCAP message $M_6 = K^*_{aci}[ID_0, K_{br}, R'_i, t_4]$. SSP_i sends this result to the T_i in an ISUP message. T_i sends back $M_7 = K_{aci}[K^*_i[TLN_i, TDN_i, R_i, t_5]]$ encrypting and signing TLN and TDN.
4. $[SSP_i \rightarrow AC_i]$ The authentication center checks TLN_i, TDN_i (aginst the database) and the signature of T_i using K_i. If authentication fails, AC_i signals SSP_i to initiate disconnecting U_i by sending a REL message to the MSB.
5. $[AC_i \rightarrow SSP_i \rightarrow T_i]$ AC_i sends $M_8 = K^*_{br}[R^*_i, t_6]$ in a TCAP message as the return result to SSP_i where R^*_i is the random number included in the confirmation ticket sent by AC_i to the MSB.
6. $[SSP_i \rightarrow T_i]$ SSP_i sends M_8 to T_i in a FACILITY message with a FIE containing a user authentication request and solicits $User_i$'s (ID, password) pair through an IVR announcement.
7. $[T_i \rightarrow SSP_i \rightarrow SSP_0 \rightarrow T_0]$ T_i sends an ALERT (CRi) message to SSP_i that sends the ALERT message to SSP_0 which in turn sends an ALERT (CR_0) message to T_0.
8. $[T_i \rightarrow SSP_i]$ When $User_i$ picks up the handset, T_i sends a CONNECT message to SSP_i, which plays an IVR announcement informing U_i of the conference participants. Then SSP_i solicits $User_i$'s ID, password pair by playing an IVR announcement.
9. $[T_i \rightarrow SSP_i]$ $User_i$ enters the (ID, password) pair that is encrypted with AC_i's public key by T_i.
10. $[SSP_i \rightarrow AC_i]$ SSP_i forwards $M_9 = K^*_{aci}[ID_i, PWD_i, R^*_i, t_7]$ to AC_i in a TCAP message. The authentication center verifies ID and password, and if incorrect, initiates connection clearance.

C. Key Distribution

1.9 $[AC_i \rightarrow SSP_i]$ $M_{10} = K^*_{aci}[ID_i, R'_i, t_8, K_i]$ signed by AC_i containing $User_i$'s ID, public key, R'_i from M_6 and t_8 is sent to SSP_i in a TCAP message.

1.10 $[SSP_i \rightarrow SSP_{br} \rightarrow AC_{br}]$ If authentication succeeds SSP_i forwards M_{10} to SSP_{br} in a CPG ISUP message, which SSP_{br} forwards to AC_{br} in a TCAP message. AC_{br} waits until either all users have connected or a timeout occurred, and adds IDs of all connected users to a list.

1.11 Simultaneously execute steps 1 and 2 for i = 0,1, ... *nc* .

1. $[AC_{br} \rightarrow SSP_{br} \rightarrow SSP_i \rightarrow T_i]$ The secure bridge starts distributing the voice encryption key (as described in section 2.6., but details omitted due to space limitation) by sending $M_{11} = K_i[K^*_{br}[K_E, R^*_i, t_9]]$ in a TCAP message from AC_{br} to SSP_{br} in a CPG message from SSP_{br} to SSP_i and in a FACILITY message from SSP_i to T_i.
2. $[T_i \rightarrow SSP_i \rightarrow SSP_{br} \rightarrow AC_{br}]$ T_i decrypts M_{11}, checks the signature, the random number and the timestamp, and recovers the voice encryption key K_E. Then T_i sends an acknowledgement back to AC_{br}.

1.12 When SSP_0 receives a *set up* message contains the FIE for the conferencing call from T_0, SSP_0 recovers FIE, routes and allocates the call to the MSB, which forwards the voice signal to every T_i.

1.13 .

5. PERFORMANCE ANALYSIS

This section computes communication delays of the proposed protocol suit. They use telecommunication connection and encryption/decryption delays for text streams published in [11] and [14] respectively. Switch response time delays in [14] are summarized in Table 1.

Table 1. Switch response delays

Type of Call Segment	*Switch Response time (ms)*	
	Mean	**95% confidence interval**
ISUP Message	205-218	≤ 337-349
Alerting	400	≤ 532
ISDN Access Message	220-227	≤ 352-359
TCAP Message	210-222	≤ 342-354
Announcement/Tone	300	≤ 432
Connection	300	≤ 432
End MF Address – Seize	150	≤ 282

We do not calculate the call teardown delay because it is the same as a normal telephone call without encryption. Under our assumptions, the delays computed for the proposed protocols are given in Table 2.

As Table 2 shows, the worse case teleconference setup time is 255,852 ms (i.e. 4.16 minutes) under the assumption that all the slave conferees are simultaneously authenticated in parallel. At a first glance, this may look excessive, but only 14,882 ms is network delay. The remaining 241,070ms is due to user interactions such as playing IVR messages, entering user ID and password or ringing time before picking up phones. The exact timing analysis is as follows. 765ms after dialing **, the call master is prompted for her ID and password. Assuming that it takes 11782ms to enter the ID and password, it takes 206,717ms for IVR messages to solicit 10 conferees. Then, cross certifying and authenticating remote user take 1,806ms. Then a 10sec IVR message is played for each conferee taking 10seconds to pick-up their phones. Thereafter, authenticating remote users takes 21,966ms. The cross-certification takes another 954ms. Finally, due to 10ms of IVR announcing the beginning of the conference, the key distribution phase takes 11,932ms. Further details delays are shown in Table 3.

Table 2. Conference call delays

Conference Call Phase	**Delay**	**Delay under assumptions: $n = 10$, $p = 10$s, $d = 10$s $a_i = b_i = 50$ms (10,000km / fiber) and x_i=10s**	**Description of the parameters and assumptions**
Call setup	$9492 + 551n + (n+3)p + nd + 5a_0 + 4 \cdot \max\{b_1 \ldots b_n\} + 10 \cdot \max\{a_1 \ldots a_n\} + \max\{x_1 \ldots x_n\}$ ms	255,952 ms	The delay to perform a RSA 1024 encryption/decryption 12ms. The number of conferencing subscribers is n The time needed to play an announcement (IVR) is a constant time of p. The time needed to enter any destination number is a constant d. The signal propagation delay between T_0 and AC_{br} is a_0 and the signal propagation delay between T_i and AC_i is a_i, where $i = 1, 2 \ldots n$. (see ITU-T Recommendation TABLE 1/Q.706). We omit a maximum 2.5ms delay between T_0 and SSP_0 (under the realistic assumption that the distance between T_0 and SSP_0 is less then 500km).
Add user by call controller	$11079 + 4p + d + 4a_0 + 2 \cdot \max\{b_1 \ldots b_n\} + 2 \cdot \max\{a_1 \ldots a_n\} + 2b_n + 7a_n + x_n$ ms	71,929 ms	
Add user by user itself	$10893 + 4p + d + 7a_0 + 2 \cdot \max\{b_1 \ldots b_n\} + 2 \cdot \max\{a_1 \ldots a_n\} + 4a_n + b_n + x_0$ ms	71,693 ms	
Drop user by call controller	$2571 + p + a_0 + 2 \cdot \max\{b_1 \ldots b_{n-1}\} + 2 \cdot \max\{a_1 \ldots a_{n-1}\}$ ms	12,821 ms	

Drop user by user itself	$2571 + p + a_0 + 2 \cdot \max\{b_1 \ldots b_{n-1}\} + 2 \cdot \max\{a_1 \ldots a_{n-1}\} + a_n + b_n$ ms	12,921 ms	The transmission propagation delay between AC_{br} and AC_i is b_i, where $i = 1, 2, \ldots, n$ The pick-up time (the time between T_i is starting ringing and $User_i$ is answering) is x_i, for $i = 0, 1, 2 \ldots n$. It includes the time needed by the subscriber to enter username/password

Adding a user – either initiated by the user or by the call master takes 71,693 to 71,929ms (with only 11,693 to 11,929 ms network delay). Notice that during this period, the existing conference is put on hold by suspending the voice stream. In our protocols, dropping a user also takes a considerable time, 12,821 to 12,921ms. Somewhat surprising is due to the amount of time taken to exchange the new voice encryption keys. The network delay in this phase is 2,821-2,921ms

Table 3. Call setup delays

Parameters	Estimated Delay under Normal Load (ms)	Description of estimated value
Initiating voice privacy call + Telephone authentication delay	$715 + a_0$	• One-way delay $T_0 \rightarrow SSP_0 \rightarrow SSP_{br} \rightarrow AC_{br}$ • 4*12 ms delay for encryption/authentication
User authentication delay	$1682 + 2a_0 + p$	• Round trip delay $AC_{br} \rightarrow SSP_{br} \rightarrow SSP_0 \rightarrow T_0 \rightarrow SSP_0 \rightarrow SSP_{br} \rightarrow AC_{br}$ • 4*12 ms delay for encryption/authentication • 300 ms playing IVR delay
Dialing the conference numbers delay	$1107 + 2a_0 + (551 + p + d)\, n$	• n round trip delays $SSP_0 \rightarrow T_0 \rightarrow SSP_0$ • Round trip delay $AC_{br} \rightarrow SSP_{br} \rightarrow SSP_0 \rightarrow T_0 \rightarrow SSP_0 \rightarrow SSP_{br} \rightarrow AC_{br}$ • n*2*12 ms delay for encryption/authentication • 300*n ms playing IVR delay • n*d delay for dialing the numbers
Cross-certification delay	$686 + b_i$	• One-way delay $AC_{br} \rightarrow SSP_{br} \rightarrow SSP_i \rightarrow AC_i$. • 2*12 ms delay for encryption/authentication

T_i authentication delay	$970 + 2a_i$	• Round trip delay $AC_i \rightarrow SSP_i \rightarrow T_i \rightarrow SSP_i \rightarrow AC_i$ • 6*12 ms delay for encryption/authentication
Remote user authentication delay	$1746 + 5a_i + x_i + p$	• Round trip delay $AC_i \rightarrow SSP_i \rightarrow T_i \rightarrow SSP_i \rightarrow AC_i$ • 6*12 ms delay for encryption/authentication • 400ms delay for ALERT message • 300 ms playing IVR delay • 300ms delay for CONN message
Cross-certification delay	$904+b_i$	• One-way delay $AC_i \rightarrow SSP_i \rightarrow SSP_{br} \rightarrow AC_{br}$ • 2*12 ms delay for encryption/authentication
Key distribution delay	$1682+ 3a_i + 2b_i + p$	• Round trip delay $AC_{br} \rightarrow SSP_{br} \rightarrow SSP_k \rightarrow T_k \rightarrow SSP_k \rightarrow SSP_{br} \rightarrow AC_{br}$, for k=1, 2 … n • 4*12 ms delay for encryption/authentication • 300 ms playing IVR delay • We assume that key generation and distribution is done in parallel for all parties in the conference

6. CONCLUSIONS

Many organizations use teleconferences to conduct private and confidential businesses with the assumption that there are no eavesdroppers on their conversation. Although eavesdropping is a crime in many countries, given that most telephone voice travels without encryption lends itself to eavesdropping. To prevent this, we have developed a fully automated teleconferencing service with corresponding algorithms called *secure bridges.* We did so by strengthening existing telecommunication standards that specify multi-party calls. Using published results we have shown by calculation that a basic secure teleconference can be setup in about 256 seconds – that is 4.25 minutes. We have presented that callers can be added and dropped within approximately 72 and 13 seconds. Our ongoing work addresses enhancing our basic algorithms to include floating – i.e. dynamically changing the call master. This would enable one caller to set up

a conference and another to take over and yet a third to finish the conference. We are also developing detailed simulations our algorithms under different load conditions.

References

[1] AT&T Webpage, www.att.com/technology/technologists/fellows/lawser.html
[2] J. G. von Bosse. Signaling in Telecommunication Networks. *John Wiley & Sons, New York,* 1998.
[3] CPKtec Research Labs web page, http://www.cpktec.com/performance.html.
[4] Specifications of Signaling System No. 7--Message Transfer Part Signaling Performance. *ITU-T Recommendation Q.706,* March 1993.
[5] Specifications of Signaling System No. 7--Signaling performance in the Telephone Application. *ITU-T Recommendation Q.706,* March 1993.
[6] Stage 3 description for multiparty supplementary services using DSS 1. *ITU-T Recommendation, Q.954,* 1993.
[7] Stage 3 descriptions for multiparty supplementary Specifications of signaling system no. 7. *ITU-T Recommendation Q.734,* 1993.
[8] Stage 2 descriptions for multiparty supplementary services. *ITU-T Recommendation Q.84,* 1993.
[9] Specifications of Signaling System No.7--Hypothetical Signaling Reference *Connection. ITU-T Recommendation Q.709,* March 1993.
[10] G. Lorenz, T. Moore, J. Hale, and S. Shenoi. Securing SS7 Telecommunications Networks. In *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security,* June 2001.
[11] T. Russell. Signaling system #7. *McGraw-Hill, New York,* 2002.
[12] R. Sailer. Security in an Open Service Environment. In *Proceedings of the Fourteenth Annual Computer Security Applications Conference,* pages 223–234, December 1998.
[13] M. Sharif, D. Wijesekera. Providing Voice Privacy Over Public Switched Telephone Networks. *Proceeding of IFIP,* pp 25-36, May 26-28, 2003, Athens, Greece, 2003.
[14] Telecordia Technologies Generic Requirements GR-1364-CORE, Issue 1, LSSGR: Switch *Processing Time Generic Requirements,* Section 5.6, June 1995.
[15] Telecordia and ITU-T specification, summarized in IETF Signaling Transport Working Group Internet draft (October 22 1999).