Participants Registration, Validation, and Key Distribution for MBone Conferencing

Rolf Oppliger

University of Berne
Institute for Computer Science and Applied Mathematics (IAM)
Neubrückstrasse 10, CH-3012 Berne, Switzerland
oppliger@iam.unibe.ch

Abstract

There are several tools available that can be used to hold multimedia conference sessions within the Internet Multicast Backbone (MBone). Most of these tools provide means for public and private conferencing, but offer no possibility to register participants of particular conference sessions, or to authenticate them accordingly. In a private conference session, every message is encrypted with a session key and this key must be distributed out-of-band, e.g. by using Privacy Enhanced Mail (PEM) or Pretty Good Privacy (PGP) messages. DiRK (Distributed Registration and Key distribution) is a technique that can be used in a conferencing system, such as the MBone, to handle participants registration and key distribution in a decentralized and fully distributed way. This paper overviews the basic principles of DiRK, and mainly focuses on a prototype implementation of DiRK in a session registration tool (`srt`) for the MBone.

1 Introduction

A conference is held if two or more parties want to discuss a specific topic. The conference is private, if participation is restricted and the conference is held behind closed doors. Otherwise the conference is public. Note that a private conference always requires participants registration and authentication, whereas a public conference may require either participants registration, or both participants registration and authentication. For example, a scientific conference may be public in nature, but require participants registration for billing purposes. In addition to that, the conference may require authentication to provide membership or student discounts. In general, there are four types of conferences to be distinguished:

- Public conferences
- Public conferences with participants registration
- Public conferences with participants registration and authentication
- Private conferences

A conferencing system allows several conference sessions to take place simultaneously. Every conference session is initiated by a particular user, namely the session holder, and it is up to the session holder to specify the conference session type, and to handle participants registration and key distribution accordingly. With respect to participants registration and key distribution, the session holder can follow either a centralized or a decentralized and distributed approach: In a centralized approach, the session holder handles participants registration and key distribution himself, whereas in a decentralized and distributed approach, he nominates agents to assist him. It is assumed that a decentralized and distributed approach better fits the requirements of conferencing systems, and this assumption has driven the design of DiRK (Distributed Registration and Key distribution) [1]. The basic idea of DiRK is to distinguish between active and passive participants in a particular conference session, and to have the active participants assist the session holder in participants registration and key distribution. In principle, any user can register as active participant, and subsequently register new users as (active or passive) participants on the session holder's behalf. In addition to the registration of users, an active participant can also provide the users with a session key that is needed in order to participate in a private conference session.

This paper overviews the basic principles of DiRK, and mainly focuses on a prototype implementation of DiRK in a session registration tool (`srt`) for the Internet Multicast Backbone (MBone). It proceeds as follows: The terminology that is used to subsequently describe the cryptographic protocols that DiRK is based on is introduced in section 2. The protocols are overviewed in section 3, and the `srt` is described in section 4. Conclusions are drawn in section 5.

2 Terminology

A protocol specifies the format and relative timing of messages exchanged between communicating parties. A cryptographic protocol is a protocol that uses cryptography, meaning that all or parts of the messages are encrypted on the sender' side and decrypted on the receiver' side. The following notation is used in this paper:

- $\mathcal{S}$ is used to refer to a particular conference session within a conferencing system. It is assumed that $\mathcal{S}$ overlays a particular (multicast) channel, and that $\mathcal{S}$ is used to refer to this channel, too.
- Capital letters, such as A, B, C, ..., are used to refer to arbitrary users of the conferencing system.
- K is used to refer to a secret key, and (k, k^{-1}) is used to refer to a public key pair. In either case, key subscripts may be used to refer to users of the conferencing system. With respect to key subscripts, capital letters are used for long-term keys, whereas small letters are used for short-term keys.
- The term $\{m\}K$ is used to refer to a message m that is encrypted with the secret key K. The same key is used for decryption, so $\{\{m\}K\}K$ equals m. Similarly, the term $\{m\}k$ is used to refer to a message m that is encrypted with the public key k. The message can be decrypted only with the corresponding private key k^{-1}. In a digital signature scheme, the user's private key is used to digitally sign messages, and the corresponding public key is used to verify signatures. In this case, the term $\{m\}k^{-1}$ is used to refer to a digital signature giving message recovery, and the term $\langle m\rangle k^{-1}$ is used to refer to a digital signature with appendix. Note that in the second case, $\langle m\rangle k^{-1}$ in fact abbreviates $m, \{h(m)\}k^{-1}$, with h being a collision-resistant one-way hash function.
- In accordance with international standardization, the term $X \ll Y \gg$ is used to refer to a certificate that has been issued by X for Y's long-term public key k_Y.
- In addition to that, the term $X < Y, \mathcal{S} >$ is used to refer to a registration certificate that has been issued by X for Y and session $\mathcal{S}$. In this case, X is necessarily registered as an active participant, whereas Y can be registered either as an active or passive participant. If Y is registered as an active (passive) participant, $X < Y, \mathcal{S} >$ in fact abbreviates $\langle X, Y, \mathcal{S}, k_y\rangle k_x^{-1}$ ($\langle X, Y, \mathcal{S}\rangle k_x^{-1}$).

3 DiRK

This section overviews some of the cryptographic protocols that DiRK is based on.

3.1 Session Initialization Protocol

In order to initialize a conference session $\mathcal{S}$, a user A randomly selects a public key pair (k_a, k_a^{-1}) for $\mathcal{S}$, and announces both $\mathcal{S}$ and k_a in public. More precisely, A digitally signs the session announcement with his long-term private key, which is k_A^{-1}, and makes $\langle \mathcal{S}, k_a\rangle k_A^{-1}$ publicly available. Note that k_a^{-1} represents the master key of the session, and that its holder is considered as being the current session holder. If $\mathcal{S}$ is going to be a private conference session, A additionally selects a session key K_S for $\mathcal{S}$.

3.2 Participants Registration Protocol

Having initialized the session $\mathcal{S}$ as described above, A is notably the first active participant of $\mathcal{S}$. If another user B wants to register as participant, it has to be distinguished whether he wants to register as active or passive participant. In the first case, B randomly selects a short-term public key pair (k_b, k_b^{-1}) for $\mathcal{S}$, whereas in the second case, B does not have to do so. In either case, B has to run the following participants registration protocol:

$$\begin{array}{lcl} 1 : \mathrm{B} & \longrightarrow & \mathcal{S} : \mathrm{REG\_REQ}([k_b]) \\ 2 : \mathrm{X} & \longrightarrow & \mathcal{S} : \mathrm{REG\_CONF}(B, X < B, \mathcal{S} >, [\{K_S\}k_b]) \end{array}$$

In step 1, the registration request (REG_REQ) message includes k_b, only if B wants to register as active participant, or if $\mathcal{S}$ is going to be a private session. In principle, any active participant X can grab the registration request, and return a registration confirmation (REG_CONF) message to $\mathcal{S}$ in step 2. This message includes B and a registration certificate $X < B, \mathcal{S} >$ for B. Note that $X < B, \mathcal{S} >$ abbreviates $\langle X, B, \mathcal{S}, k_b \rangle k_x^{-1}$ if B is granted active participation, and $\langle X, B, \mathcal{S} \rangle k_x^{-1}$ if B is granted passive participation. If $\mathcal{S}$ is is going to be a private conference session, the message additionally includes $\{K_S\}k_b$, which is the session key K_S encrypted with B's public key k_b. In this case, it is mandatory that every session participant either registers as an active participant and uses his public key pair accordingly, or uses any other public key pair that he holds. A slight modification of the REG_REQ message is necessary for the participants registration protocol to additionally provide authentication:

$$1 : \mathrm{B} \quad \longrightarrow \quad \mathcal{S} : \mathrm{REG\_REQ}(\langle [k_b], Y \ll B \gg \rangle k_B^{-1})$$

In this case, the message additionally includes a certificate for B's long-term public key, and is digitally signed with the corresponding private key k_B^{-1}. If B happens to be the first user who sends a registration request to $\mathcal{S}$, B receives his registration confirmation from A. Afterwards, if A confirms B as an active participant, they both act as active participants and confirm registration requests that are originated by another user C. Otherwise, if B is not registered as an active participant of $\mathcal{S}$, C must be confirmed by A, too. Note that as a result of this registration scheme a participants registration tree (PRT) is subsequently built for $\mathcal{S}$.

3.3 Registration Validation Protocol

The basic idea of DiRK is to couple the status report that every participant periodically sends to the multicast session with a message that allows other participants to validate the registration of the sending participant. If the sending participant is a passive participant, the coupled message is simply the registration certificate. If the sending participant is an active participant, the coupled message consists of both the registration certificate, and a timestamp that is digitally signed with the private key that corresponds to the public key included in the registration certificate. In either case, the registration validation protocol consists of a registration validation (REG_VAL) message that a participant B periodically sends to $\mathcal{S}$:

$$\mathrm{B} \quad \longrightarrow \quad \mathcal{S} : \mathrm{REG\_VAL}(B, X < B, \mathcal{S} >, [\{T\}k_b^{-1}])$$

Any user C can grab the message and validate the registration of B accordingly. In principle, C has to trace back the PRT from B to the current session holder and validate every single registration certificate accordingly.

4 Session Registration Tool

The Multicast Backbone (MBone) is an experimental network that is overlaid on the existing Internet to carry IP multicast datagrams. Application tools, such as `vat`, `nevot`, `nv`, `ivs`, `vic` and `wb` can be used to hold multimedia conference sessions, and the `sd` and `sdr` tools can be used to announce these sessions in public, as well as to have MBone application tools start up accordingly [2].

In spite of the various threats related to IP multicast, security-related issues for the MBone have rarely been touched upon to date. There is, in general, no control as to who can register a dynamic session address, who can or cannot join a session, who can receive traffic from or inject traffic into the session, or how to deal with the security implications when a host leaves a session.

MBone application tools currently support either public conferencing, or a rather immature form of private conferencing: With regard to public conferencing, there is currently no possibility to register users as participants of particular MBone conference sessions, or to authenticate them accordingly. With regard to private conferencing, some MBone application tools support session encryption. Every datagram that is sent to (or received from) an IP multicast address is encrypted (or decrypted) with a corresponding session key, and this session key is shared among the participants. However, the MBone application tools that support session encryption don't address key distribution either, and relay on out-of-band key distribution mechanisms instead.

In order to fully support private MBone conferencing, key distribution facilities must be added to the MBone application tools. As part of the European MICE project, a corresponding Secure Conferencing User Agent (SCUA) has been developed at GMD Fokus. The idea of the SCUA is to have the initiator of a particular MBone session invite the participants with corresponding PEM messages, and to have these messages carry the session key. Note that this approach allows session keys to be distributed out-of-band, namely with PEM messages, but that PEM is not widely deployed today. In addition to that, it is possible and very likely that conferences will be held that lack an explicit initiator. Nevertheless, a similar approach is being followed by the Lawrence Berkeley Laboratory (LBL). The LBL intends to enhance the `sd` tool with key registration facilities, and to use a secure e-mail systems, such as PEM or PGP, to distribute session keys out-of-band.

Note that DiRK follows a decentralized and fully distributed approach for participants registration and key distribution, and that this approach better fits the requirements of MBone conferencing. There are two possibilities to implement DiRK: one is to implement the protocols that DiRK is based on in a MBone application tool of its own, and another is to build a toolkit that can be used to incorporate DiRK into other MBone application tools. Following the first approach, a session registration tool (`srt`) for the MBone is being prototyped at the University of Berne. The development environment consists of C/C++, Tk/Tcl, and Tcl-DP, as well as SecuDE, which is a cryptographic toolkit developed at GMD. The `srt` is to allow users to create, edit, and possibly announce MBone conference sessions in public, to register as participant of particular conference sessions, and to validate the registration of others.

5 Conclusions

This paper has overviewed the basic principles of DiRK (Distributed Registration and Key distribution), and has mainly focused on a prototype implementation of DiRK in a session registration tool (`srt`) for the Internet Multicast Backbone (MBone). The tool can be used to handle participants registration and key distribution in MBone conference sessions in a decentralized and fully distributed way. The tool will be released for public scrunity later this year. In addition to the prototype implementation, DiRK is also being explored to be used in distance education and teleteaching to set up and run online universities [3]. Further information on DiRK, the `srt`, and applications thereof can be found on the World Wide Web (WWW) by following the URL `http://iamwww.unibe.ch/~oppliger/Research/dirk.html`.

References

[1] R. Oppliger and A. Albanese, "Distributed registration and key distribution (DiRK)," in *Information Systems Security — Facing the information society of the 21st century* (S. Katsikas and D. Gritzalis, eds.), (London, UK), pp. 199 – 208, Chapman & Hall, May 1996. Proceedings of the IFIP TC11 12th International Conference on Information Security (IFIP SEC '96).

[2] M. Macedonia and D. Brutzman, "MBone Provides Audio and Video Across the Internet," *IEEE Computer*, vol. 27, pp. 30 – 36, April 1994.

[3] R. Oppliger, M. Bracher, and A. Albanese, "Distributed registration and key distribution for online universities," in *Communications and Multimedia Security (CMS '96)*, (London, UK), Chapman & Hall, September 1996. Proceedings of an IFIP TC6 and TC11 Joint Working Conference held in Essen, Germany, September 23 - 24, 1996.