RSSP-2

铁路信号安全通信协议

（V1.0）

2008 年 12 月

修订历史

版本号	修改说明	日期
V0.1	初稿。	2008.11.15
V0.2	根据 2008.11.17 日通号公司讨论意见修改。	2008.11.20
V0.3	根据 2008.11.24 日 C3 组会议讨论意见修改。	2008.11.27
V0.4	根据 2008.12.24 日 C3 组及铁科院、交大、卡斯科等单位讨论意见修改。	2008.12.26
V1.0	修改部分文字表述，根据 2008.11.31 日 C3 组会议讨论意见修改。	2008.12.31

目录

1. 简介

1.1. 目的及范围

1.1.1.1. 本文件规定了信号安全设备之间通过封闭式网络或开放式网络进行安全相关信息交互的功能结构和协议。

1.1.1.2. 本规范适用于铁路信号安全设备之间的安全通信接口。

1.2. 参考文献

3.2.1.1.1 名称	日期	描述
EN 50159-1	2001 年 03 月	Railway applications – Communication, signalling and Processing systems – Part 1: Safety-related communication in closed transmission systems <铁道应用：封闭式传输系统中安全通信要求>
EN 50159-2	2001 年 03 月	Railway applications – Communication, signalling and Processing systems – Part 2: Safety-related communication in open transmission systems<铁道应用：封闭式传输系统中安全通信要求>
科技运[2008] 127 号	2008 年	CTCS-3 级列控系统系统需求规范（SRS），v1.0
科技运[2008] 34 号	2008 年	CTCS-3 级列控系统总体技术方案，v1.0
Subset-037	2005 年 10 月 14 日	ERTMS\ETCS-Class1， Subset 037，Euroradio FIS 欧标规范子集 037：欧洲无线功能接口规范，v2.3.0
Subset-038	2005 年 12 月 21 日	离线密钥管理 FIS，v2.1.11
Subset-039	2005 年 08 月 31 日	RBC/RBC 移交 FIS，v2.1.2
Subset-108	2006 年 06 月 08 日	ETCS/ERTMS 1 级，TSI 附录 A，v1.1.0
ITU-T X.214	1993 年 11 月	信息科技，开放系统互联，传送服务定义
ITU-T X.224	1993 年 11 月	信息科技，开放系统互联，提供 OSI 连接模式的传送协议
RFC0791	1981 年 09 月 01 日	网络协议 v4
RFC2460	1998 年 12 月	网络协议 v6
RFC0793	1981 年 09 月 01 日	传输控制协议 v4
ISO/IEC 3309	1991 年 06 月	信息技术——系统间的通信和信息交换——高级数据链路控制程序（HDLC）——框架结构

1.3. 术语和定义

1.3.1.1. 本文件中使用了标准 EN 50159-1 和 EN 50159-2 的定义，并附加使用了以下术语。

1.3.1.2. **应用处理：**描述通信关系的应用层实体。

1.3.1.3. **执行周期：**处理周期以及与其相关的递增计数（基于计算机的恒定处理周期）。

1.3.1.4. 密钥管理规范的缩略语和定义包含在本规范的资料性附录中。

1.4. 缩略语

缩略语	*含义*
ALE	适配及冗余管理层实体
ALEPKT	ALE 数据包，ALE 之间交换的 PDU
ApPDU	应用 PDU
CFM	通信功能模块
EC	执行周期
IP	网际协议
MASL	消息鉴定安全层
PDU	协议数据单元
QoS	服务质量
SaCEPID	安全连接端点识别符
SAI	安全应用中间子层
SAP	服务接入点
SaPDU	安全协议数据单元
SaS	安全服务
SFM	安全功能模块
SL	安全层
SN	序列号
TCEPID	传输连接端点识别符
TCP	传输控制协议
TPDU	传输协议数据单元
TS	传输服务
TSAP	传输服务接入点
TTS	三重时间戳

2. 参考结构

2.1. 综述

2.1.1.1. 封闭式网络在 EN50159-1 中定义为：“可连接设备的最大数量和拓扑结构已知的，传输系统的物理特征是固定的传输系统，并可以忽略未授权访问的风险。”

2.1.1.2. 开放式网络在 EN50159-2 中定义为：“连接设备数量未知的传输系统，它拥有未知的、可变的且非置信的特征，用于未知的通信服务，对此系统应评估未经授权的访问。”

2.1.1.3. 这两种网络类型都应被考虑。

2.1.1.4. 安全通信系统间的总体结构（根据 EN50159-2）如图 1 所示。

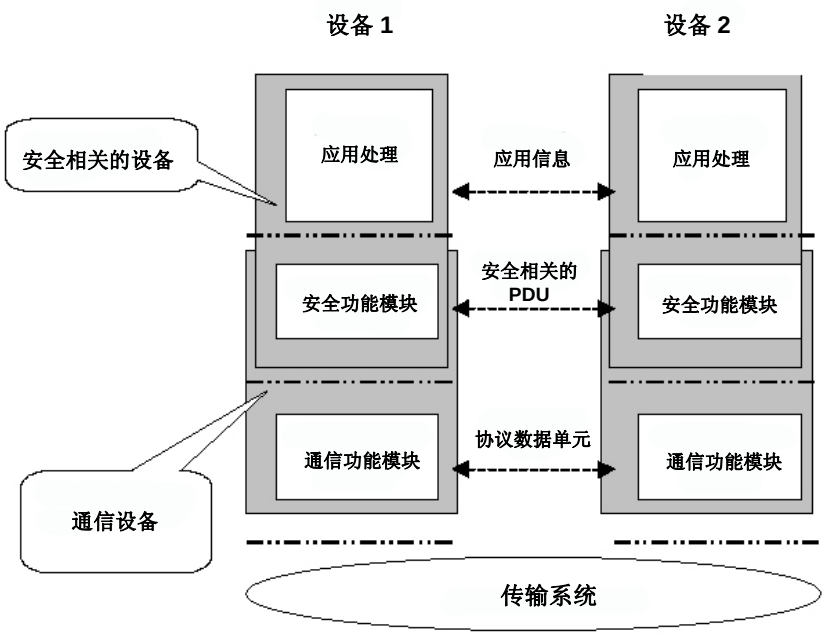


图 1：安全通信系统的总体结构

2.2. 铁路信号安全设备间安全通信接口

2.2.1.1. 本节描述了安全通信系统的功能结构，对内层实现不作限制。不应将其理解为对软件实现的要求。

2.2.1.2. 铁路信号安全设备之间安全通信接口采用分层结构。该接口规范所包含的各层如图 2 中阴影部分所示。

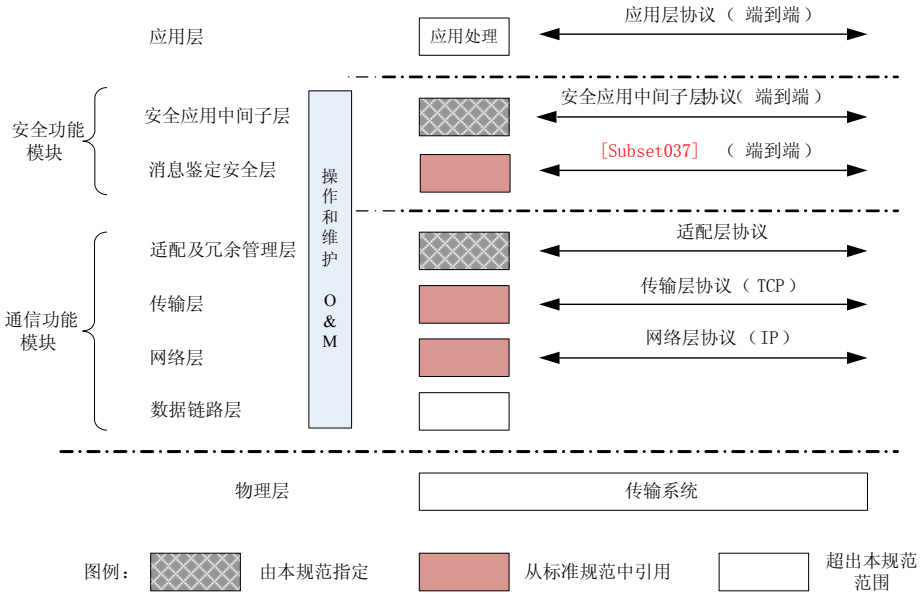


图 2：安全通信系统的结构

2.2.1.3. 安全信息传输的应用协议见各安全设备间应用层协议，不属于本规范范围。

2.2.1.4. 经由非安全低层传输的安全相关信息需要在安全层中进行处理。消息鉴定安全层（MASL）参照［Subset-037］制定，在 MASL 层的基础上增加安全应用中间子层（SAI）。

2.2.1.5. 适配及冗余管理层（ALE）提供 MASL 层和传输层之间的适配和冗余处理。

2.2.1.6. 传输层协议参见 TCP[RFC0793]。重发功能由 TCP 的常规机制来提供。

2.2.1.7. 网络层协议参见 IP［RFC0791］。

2.2.1.8. 本规范不对数据链路层作规定。

2.2.1.9. 本规范不对物理层作规定。

2.2.1.10. 操作和维护（O&M）属于具体实施的范畴，本规范不作规定。

2.3. 各层功能

2.3.1. 安全功能模块（SFM）

2.3.1.1. 本模块提供的安全层必须能够对 EN 50159-1 和 EN 50159-2 中列出的威胁进行检测并提供充分的防护。

2.3.1.2. 安全层实现以下安全相关的传输功能。

- 消息的真实性（源地址和目的地址）；
- 消息的序列完整性；
- 消息的时效性；
- 消息的完整性；
- 安全错误报告；
- 配置管理（安全设备间的安全通信协议栈）；
- 访问保护。

2.3.1.3. MASL 层提供以下功能：

- 消息的真实性（源地址和目的地址）；
- 消息的完整性；
- 访问保护。

2.3.1.4. SAI 层提供所需的其他安全相关的传输功能。

2.3.1.5. 序列错误防护通过在用户数据中增加序列号实现。

2.3.1.6. 消息时效性防护通过在用户数据中增加 TTS 或者 EC 计数实现。

2.3.1.7. EC 和 TTS 对消息时延具有相同的防护等级。

2.3.1.8. TTS 与 EC 计数仅允许一项有效，具体实施中由通信双方协商决定。

2.3.2. 通信功能模块（CFM）

2.3.2.1. 通信功能模块提供非置信的传输。

2.3.2.2. 此模块提供以下功能：

- MASL 层和传输层之间的适配；
- 冗余功能，以满足系统可用性需求；
- 数据的可靠、透明和双向传输；
- 必要时协议数据单元的重传；
- 通道可用性监测。

2.4. 传输系统的分类

2.4.1.1. 为了使本规范具有通用性，不对开放式传输系统类别作限定。本规范参考具有最高安全风险级别的开放式传输系统类别 7［参见 EN 50159-2］来制定。

2.5. 假设

2.5.1.1. 设定条件如下：

- 对[Subset 037]中规定的“高优先级”消息不作要求；

- 目前对多路复用技术不作要求，但是该项功能可以通过在每个逻辑连接中使用一条 TCP 链路来实现；
- 非显式流量控制；
- SFM 检测出的安全相关错误可能在 SAI 层之外进行处理；
- 用户数据长度不超过 1000 字节。

3. 安全功能模块

3.1. 简介

3.1.1.1. 本节规定安全功能模块（SFM）。

3.1.1.2. 本节仅描述相关的功能接口，以确保在安全功能模块层面的互联。

3.1.1.3. 安全功能模块的组成：

- MASL 层；
- SAI 层。

3.1.1.4. 通过这两层结合将对 EN 50159-2 文件中所定义的威胁提供全面的防护。

3.2. 安全功能模块的功能

3.2.1.1. 安全功能模块提供同开放式传输系统类别 7 相适应的安全服务。

3.2.1.2. 本节规定了在安全功能模块中防护技术的具体实现。

3.2.1.3. 根据 EN 50159-2 标准，对于一般的传输系统而言，所有可能的威胁如下（参见 EN 50159-2 的定义）：

- 重复；
- 删除；
- 插入；
- 重排序；
- 损坏；
- 延迟；

- 伪装。

3.2.1.4. 为了减少标准中定义的威胁风险，安全功能模块应提供以下的安全服务（参见 EN 50159-2 的定义）：

- 消息的真实性；
- 消息的完整性；
- 消息的时效性；
- 消息的有序性。

3.2.1.5. MASL 层和 SAI 层的结合为开放式传输系统提供了一种安全保护措施。

3.2.1.6. MASL 层可以预防以下威胁：

- 损坏；
- 伪装；
- 插入。

3.2.1.7. 通过添加安全码（消息验证码 MAC）和连接标识符（源和目的地标识符）提供防护。

3.2.1.8. SAI 层可以预防以下威胁：

- 延迟；
- 重排序；
- 删除；
- 重复。

3.2.1.9. 通过添加延迟防御技术（EC 或 TTS）和序列号（SN）提供保护。

3.2.1.10. 安全功能模块提供的保护如下表所示：

威胁	**防御**							
	序列号 SN	TTS 或 EC	超时	反馈信息	源和目的地标识符	消息识别过程	安全码	加密技术
重复	X							
删除	X							
插入					X			
重排序	X							
损坏								X
延迟		X						
伪装								X

表 1：安全功能模块的防御技术

3.3. 消息鉴定安全层（MASL）

3.3.1.1. MASL 层由［Subset-037］规定。

3.3.1.2. MASL 层参照［Subset-037］，但是不使用其中的高优先级数据业务。所有的数据传输均应使用正常的数据业务来进行。

3.3.1.3. 下表规定 SAI-MASL 接口的 SaS 原语参数的使用。

参数	Subset-037	SFM 的使用	说明
地址类型	5.2.	如果需要的话，可以使用	
网络地址	5.2.	如果使用，可用于识别被叫方的 32 位目的 IP 地址和 16 位目的 TCP 端口号	
移动网络 ID	5.2.	不使用	
主叫 CTCS ID 类型	5.2.	主叫安全设备 CTCS ID 类型	
主叫 CTCS ID	5.2.	用于初始化连接的主叫 CTCS ID	
被叫 CTCS ID 类型	5.2.	被叫安全设备 CTCS ID 类型	
被叫 CTCS ID	5.2.	用于接受连接请求的被叫 CTCS ID	
应用类型	5.2.	参见 Subset-037 中定义的应用类型	
服务质量类型	5.2.	QoS 域用于表示建立连接的服务类型。	服务类型 A 和服务类型 D 可供使用
SaCEPID	5.2.	由本地提供用来辨识各个安全连接的参数	

表 2：SaS 原语参数

3.3.1.4. 由本地实现 SAI 层和 MASL 层之间的接口。

3.3.1.5. 下表规定了安全信号设备间安全通信接口中 MASL 层的配置：

参数	Subset-037	SFM 值	说明
SaS 用户数据的最大长度	5.3.	1000 字节	
T_{estab}	7.2.5.3	最大应用值：40 秒	推荐值为 40 秒，对于安全信号设备间的通信可能采用更低值

表 3：MASL 层的配置

3.4. 安全应用中间子层（SAI）

3.4.1. 概述

3.4.1.1. 安全应用中间子层提供：

- 通过序列号和 TTS/EC 计数对数据进行保护；
- 到应用层接口；
- 到 MASL 层接口。

3.4.1.2. 通过给用户数据添加一个序列号域，防止数据的重复、删除和重排序。

3.4.1.3. 通过给用户数据添加 TTS 或 EC 计数防止数据延迟。

3.4.1.4. 时间戳的防御技术基于发送方和接收方之间时钟偏移的计算。

3.4.1.5. 为了发送方和接收方依据执行初始化程序对时钟偏移进行估算，需要在 SAI 帧头中定义初始化过程的消息类型。

3.4.1.6. 对于由发送方发送给接收方的消息，通过添加以下字段构成 TTS：

- 数据传输时的发送方时间戳；
- 发送方接收的上一条消息中的接收方时间戳；
- 发送方接收上一条消息时的发送方时间戳。

3.4.1.7. 下图描述了 TTS 信息：

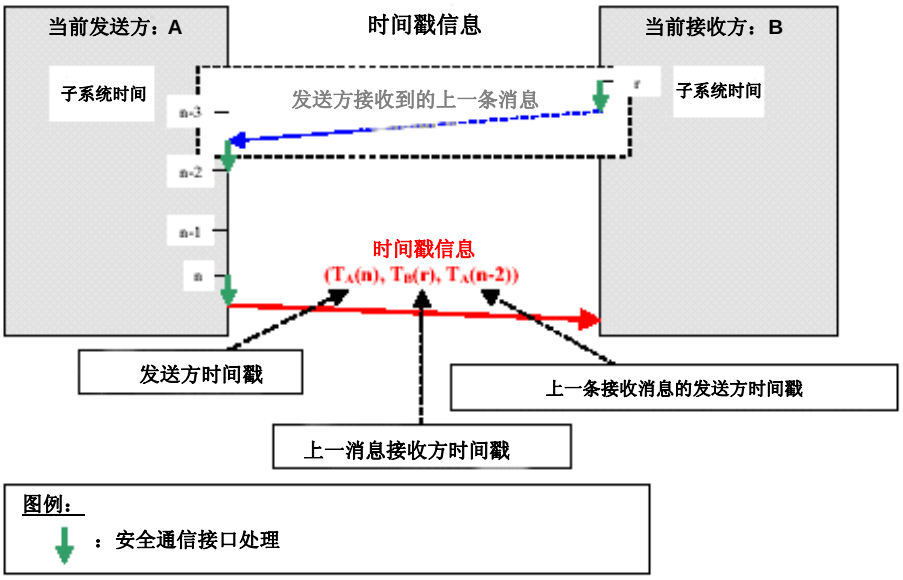


图 3：时间戳信息

3.4.1.8. 当不使用 TTS 的方法时，选择 EC 防御技术。EC 防御技术通过在用户数据上添加 EC 计数来检查消息的寿命。

3.4.1.9. EC 防御技术也要求有一个初始化过程。在此过程中，每一个通信实体的 EC 值被发送给对等实体。

3.4.1.10. EC 和 TTS 的防御技术是相互排斥的。

3.4.2. 到 SAI 层服务接口

3.4.2.1. SFM 通过安全服务接入点上的安全服务原语及其相应的参数，提供安全服务。

3.4.2.2. SAI 层仅提供功能规范，而原语的实施由本地提供，不会影响安全设备的互通。

3.4.2.3. SAI 层连接建立服务：

- SAI-CONNECT.request：用户要求 SAI 建立连接；
- SAI-CONNECT.indication：被叫 SAI 实体收到连接请求后通知被叫 SAI 用户；
- SAI-CONNECT.response：应答 SAI 用户用来接受到 SAI 实体的连接；
- SAI-CONNECT.confirm：主叫 SAI 实体获得被叫对等实体的响应后向主叫 SAI 用户报告 SAI 连接成功建立；

3.4.2.4. SAI 数据传输服务：

- SAI-DATA.request：SAI 用户用来向对等实体传输应用数据；
- SAI-DATA.indication：向 SAI 用户表示来自对等实体数据已成功接收；

3.4.2.5. SAI 连接释放服务：

- SAI-DISCONNECT.request：SAI 用户强制释放 SAI 连接；
- SAI-DISCONNECT.indication：用来通知 SAI 用户 SAI 连接释放；

3.4.3. 到 MASL 层服务接口

3.4.3.1. SAI 层实现的功能在 MASL 层之上。

3.4.3.2. MASL 层的应用约束了 SAI 层的设计，因此为了能适配 MASL 层，SAI 层应能够与 Subset-037 中规定的“安全服务接口”适配。

3.4.4. 消息结构

3.4.4.1. 消息结构如下图所示。

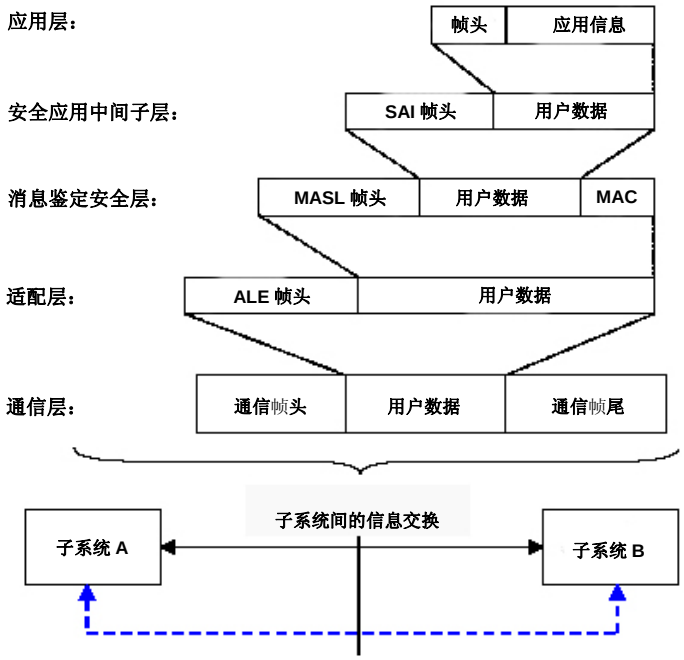


图 4：消息结构

3.4.4.2. 消息类型决定 SAI 帧头结构。

3.4.4.3. 应考虑到两种不同情况：

- SAI 帧头适用于安全数据的传输（参见 Subset-037）；
- 仅 MASL 层用于安全连接管理。

3.4.4.4. 就安全数据传输而言，由 SAI 层添加的帧头结构如下：

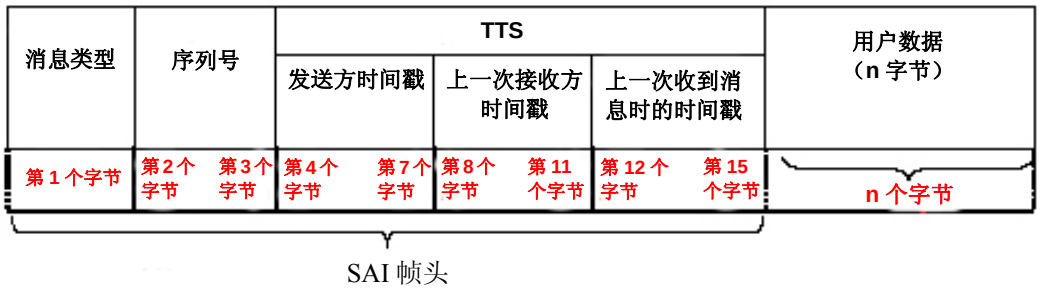


图 5：使用 TTS 时的 SAI 帧头结构

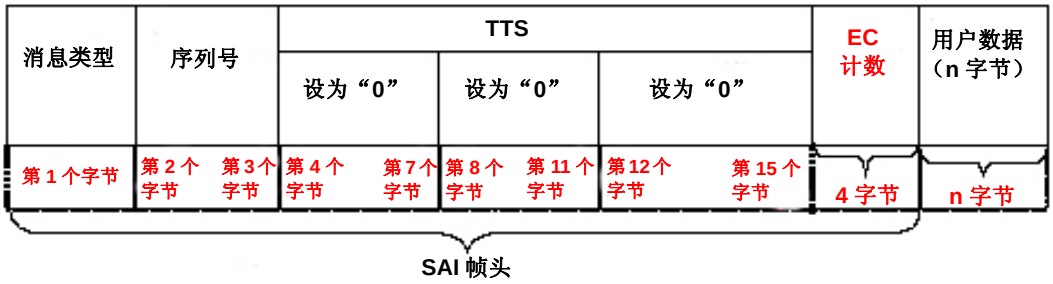


图 6：使用 EC 时的 SAI 帧头结构

3.4.4.5. SAI 层的所有域均使用 Big Endian 方式编码。

3.4.4.6. “消息类型”域用于识别消息类型，使用一个字节进行编码。

3.4.4.7. “序列号”域用于定义消息顺序号，使用两字节编码。

3.4.4.8. 如果使用了 TTS，“发送方时间戳”域应填写消息传送至本地 MASL 层实体时的发送方时间戳。发送方时间戳使用四个字节编码。

3.4.4.9. 如果使用了 TTS，“上一次接收方时间戳”域应填写由“接收方”产生，从接收方传输给发送方的最后一个消息的时间戳，OffsetStart 信息除外（见§5.4.8.4）。本时间戳使用四个字节编码。

3.4.4.10. 如果使用了 TTS，“上一次的消息接收时间戳”域应填写由本地 MASL 层实体上传上一条消息时的本地时间戳，OffsetStart 信息除外（参见§5.4.8.4）。本时间戳使用四个字节编码。

3.4.4.11. 只有使用 EC 防御技术时，才使用“EC 计数”域，并出现在帧头中。

3.4.4.12. EC 计数使用四字节编码。

3.4.4.13. EC 防御技术和 TTS 防御技术互相排斥，若使用 EC 防御技术，则与 TTS 防御技术相关的域将不被检查，此时“TTS”域所有字段的值应被设为 0。

3.4.5. SAI 协议

3.4.5.1. 连接过程

3.4.5.1.1. 两个设备之间的连接过程如下图所示。

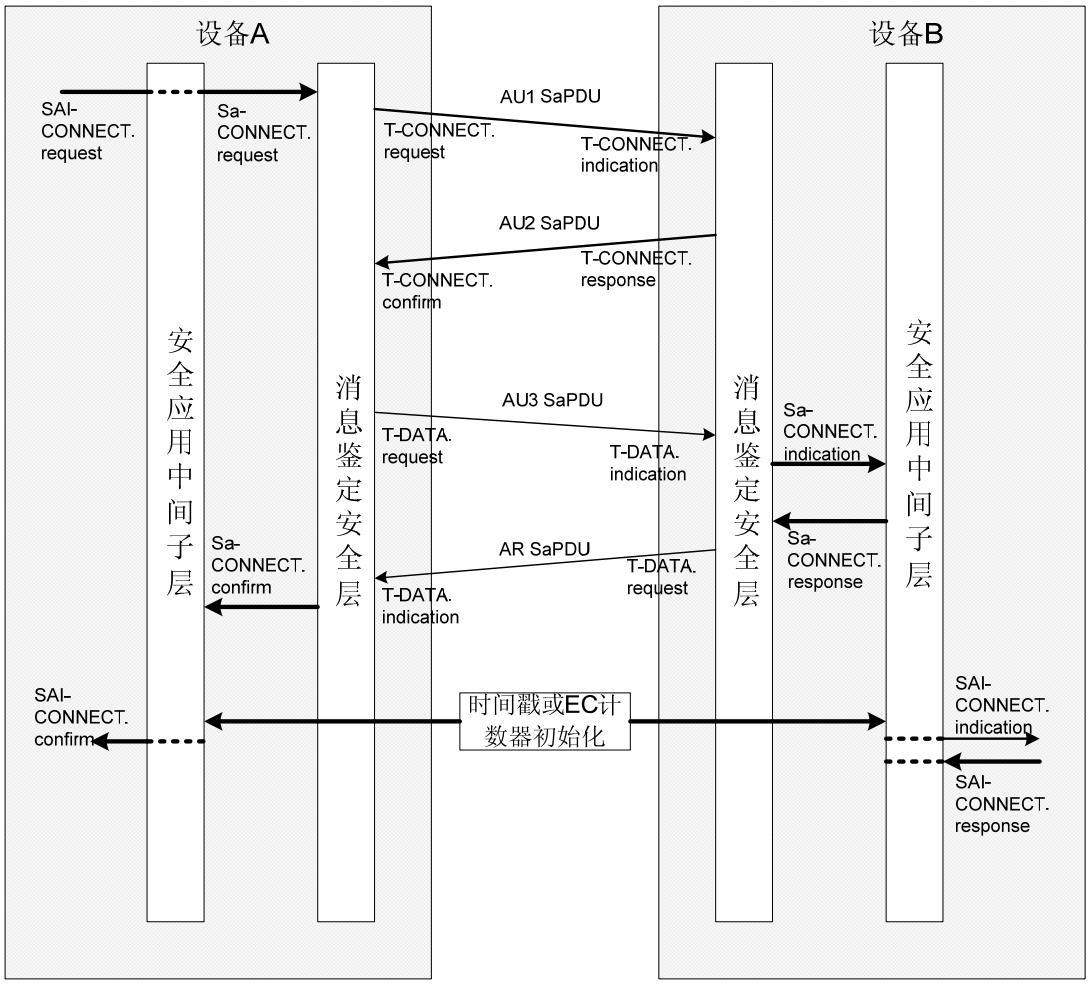


图 7：SAI 连接过程

3.4.5.1.2. SAI 服务原语应被映射到 SA 服务原语上，以用于连接建立。

3.4.5.2. 断开连接过程

3.4.5.2.1. 两个设备之间的连接断开过程如下图所示。

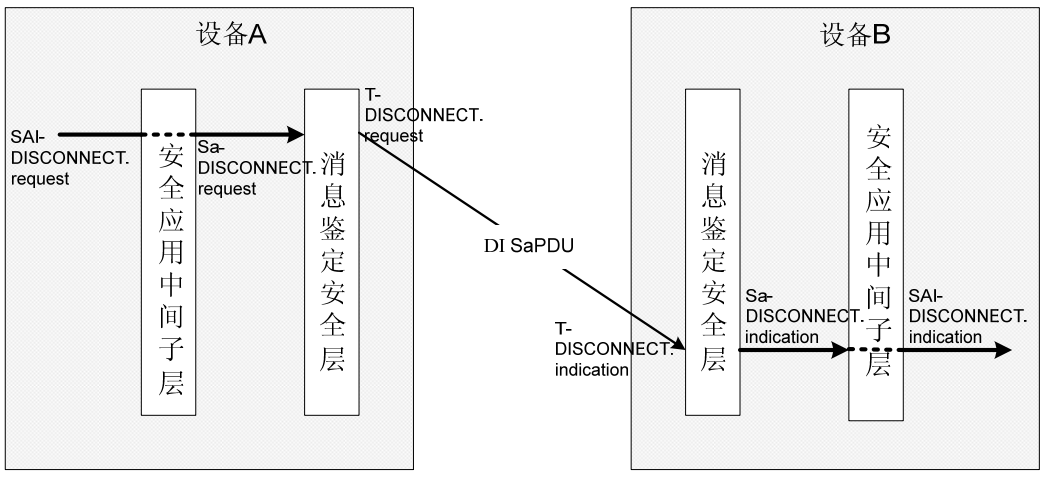


图 8：断开连接过程

3.4.5.2.2. SAI 服务原语应被映射到 SA 服务原语上，以用于连接断开。

3.4.5.3. 应用数据交互过程

3.4.5.3.1. 以下过程用来描述如何传输应用数据消息。

3.4.5.3.2. 通过使用 SAI-DATA.request，应用层应能够将数据发送至对等实体（见 Subset-037）。

3.4.5.3.3. 通过使用 SAI-DATA.request，SAI 层应能够识别所连接的 SaCEPID。

3.4.5.3.4. 在 SAI 帧头中，SAI 层应在应用数据上增加：

- 应用数据交互的消息类型；
- 序列号；
- TTS 域。如果使用 EC 防御技术，则 TTS 设为“0”；
- EC 计数及其版本（仅在使用 EC 防御技术时）。

3.4.5.3.5. SAI 层通过使用 Sa-DATA.request 原语，将其处理完的数据传送至 MASL。Sa 用户数据参数由消息类型，序列号，TTS 域和 EC 域连接组成。只有使用 EC 防御技术时，EC 域才会出现。

3.4.5.3.6. 以下过程用来描述如何接收应用数据消息。

3.4.5.3.7. MASL 层可以使用 Sa-DATA.indication 将数据传送至 SAI 层处理，并由 SAI 层传送到应用层。

3.4.5.3.8. Sa-DATA.indication 应提供 SaCEPID。

3.4.5.3.9. Sa 用户数据参数由下列部分组成：

- 应用数据的消息类型；
- 序列号；
- TTS 域。如果使用 EC 防御技术，则时间戳设为“0”；
- EC 计数（仅在使用 EC 防御技术时）。

3.4.5.3.10. 5.4.5.3.10 “消息类型”应属于为应用数据交互而定义的消息类型之一。

3.4.5.3.11. 5.4.5.3.11 SAI 应在 EC 计数或 TTS 检查前对序列号进行检查。

3.4.5.3.12. 5.4.5.3.12 如果使用 EC 防御技术，则无须检查 TTS 域，而只须检查 EC 域。

3.4.5.3.13. 5.4.5.3.13 如果使用 TTS 防御技术，则必须检查 TTS 域。

3.4.5.3.14. 5.4.5.3.14 若以上所有检查均成功执行，则应使用 SAI-DATA.indication 将应用数据和 SaCEPID 传送至应用层。

3.4.5.3.15. 5.4.5.3.15 应用数据交互如下图所示：

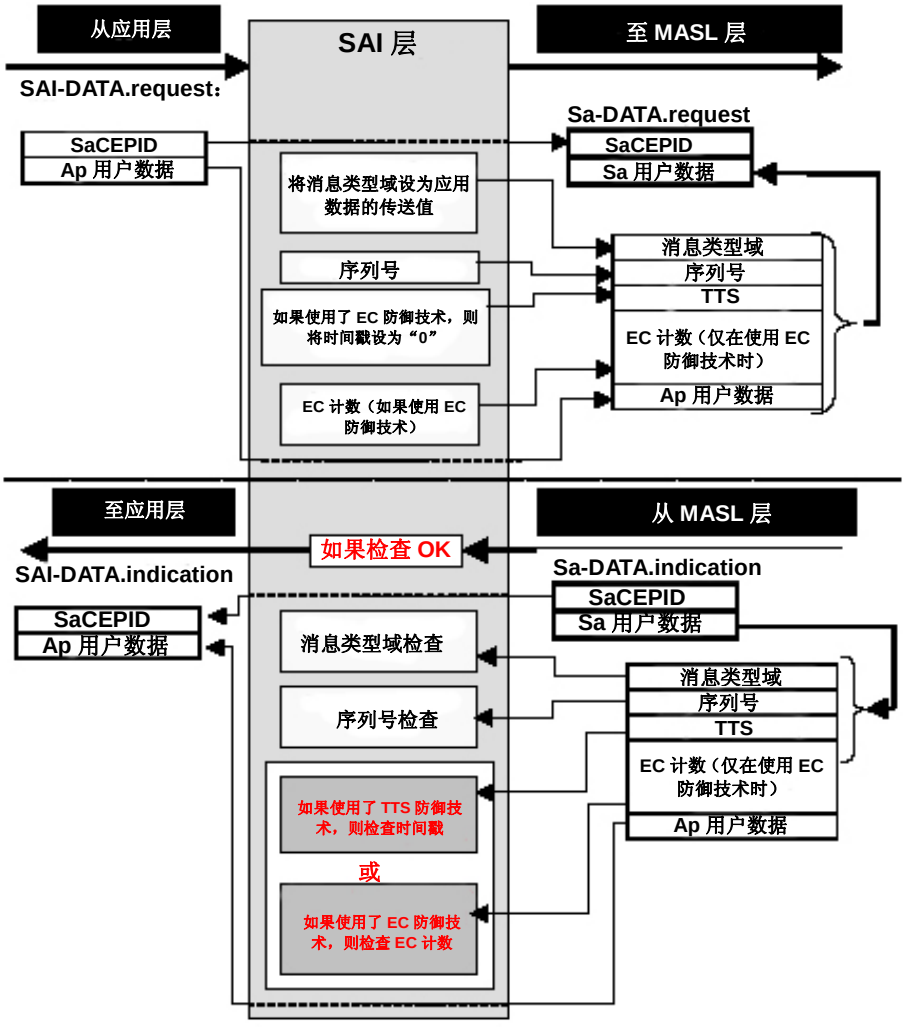


图 9：应用数据交互过程

3.4.5.4. SAI 管理消息

3.4.5.4.1. SAI 执行某些特定程序，通过 TTS 或 EC 计数确保对消息的防护（见第 5.4.8.5 节，5.4.8.7 节，5.4.9.3 节和 5.4.9.6 节）。在执行此类程序期间，双方 SAI 层之间应交互 SAI 管理消息。

3.4.5.4.2. SAI 管理消息通过消息类型域的特定值或无任何应用数据的消息进行识别。

3.4.5.4.3. SAI 管理信息应通过 Sa-DATA.request 和 Sa-DATA.indication 原语在 SAI 层之间进行交换。

3.4.5.4.4. SN，TTS 和 EC 域应采用和应用数据交互过程相同的处理方式。

3.4.5.4.5. 在消息传输中，根据管理消息的类型选择消息类型域值。用户数据是来自应用层还是由 SAI 自行计算，这取决于管理消息的类型。如果没有任何应用数据被传送至对等应用层，则 SAI 层会为管理消息选择适当的 SaCEPID。

3.4.5.4.6. SAI 应根据接收的管理消息类型，决定用户数据是由 SAI 自行处理，或将 SaCEPID 和应用数据一起传送至应用层。

3.4.6. 消息类型域

3.4.6.1. 共定义 10 种消息类型，其中，TTS 防御技术中使用的消息类型有 6 种，EC 防御技术中使用的消息类型有 4 种。

3.4.6.2. TTS 防御技术中使用的消息类型如下所示：

OffsetStart 消息（用于时钟偏移估算的第 1 个消息）：	1
OffsetAnsw1 消息（用于时钟偏移估算的第 2 个消息）：	2
OffsetAnsw2 消息（用于时钟偏移估算的第 3 个消息）：	3
OffsetEst 消息（用于时钟偏移估算的第 4 个消息）：	4
OffsetEnd 消息（用于时钟偏移估算的第 5 个消息）：	5
使用 TTS 保护的应用消息：	6

3.4.6.3. EC 防御技术中使用的消息类型如下所示（十六进制）：

EC 起始消息：	81
使用 EC 保护的应用消息：	86
使用 EC 保护并请求应答的应用消息：	87
使用 EC 保护并含有应答确认的应用消息：	88

3.4.7. 序列号防御技术

3.4.7.1. 序列号以 2 字节进行编码（Big Endian）。

3.4.7.2. 序列号值从 0 到 65535。

3.4.7.3. 每一个通信方向上的序列号应该是独立的。

3.4.7.4. 对序列号不作初始化要求。对于从对等实体处接收到的第一个序列号，接收方不做检查。

3.4.7.5. 接收方无须检查接收到的第一条消息的序列号，只需对后续的消息检查其序列号与上一个序列号之间的差异。

3.4.7.6. 如果序列号值未达到最大值，则在此传输方向上的下一条消息序列号加 1。

3.4.7.7. 一旦序列号值达到最大值，则下一条传输消息的序列号设为“0”。

3.4.7.8. 两个设备间的消息编号如下图所示：

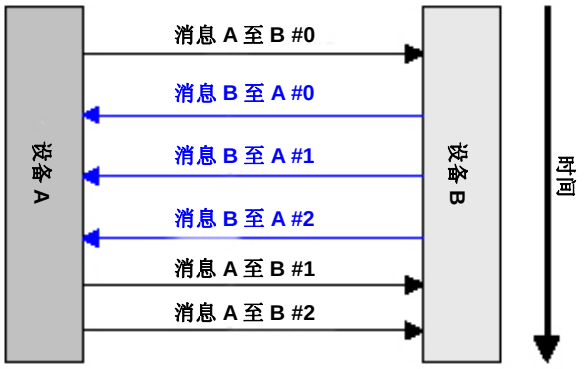


图 10：消息编号

3.4.7.9. 序列号错误

3.4.7.9.1. 序列号可检测到下列错误：

- 消息重复；
- 消息删除；
- 消息重排序。

3.4.7.9.2. 一旦检测到错误，SAI 不能采取任何措施恢复丢失的数据。

3.4.7.9.3. 为消息序列检查而定义参数 N。N-1 是代表允许丢失消息的数量。需要配置 N 值，N 值等于或大于 1。

3.4.7.9.4. 如果接收到的 SN 不等于上次接收消息的 SN＋1，则被认为是消息序列错误。

3.4.7.9.5. 如果接收到的 SN 大于上次接收消息的 SN＋N，则应丢弃此消息，并释放安全连接。

3.4.7.9.6. 如果接收到的 SN 大于上次接收消息的 SN＋1，并小于或等于 SN＋N，则不应丢弃此消息中的应用数据，并根据规定的错误处理方式（见第 5.4.10.1.2 节）来处理这一事件。

3.4.7.9.7. 5.4.7.2.7　　如果接收到的 SN 小于或等于上次接收消息的 SN，则应丢弃此消息。

3.4.7.9.8. 5.4.7.2.8　　需要根据 SAI 错误处理程序（见第 5.4.10 节）对序列号错误作出相应反应。

3.4.7.9.9. 5.4.7.2.9　下图说明当发生重复消息，删除消息或重排序消息时，SN 防御技术所采取的行为。假定 N＝3，则允许丢失消息的数量为 0，1 或 2。

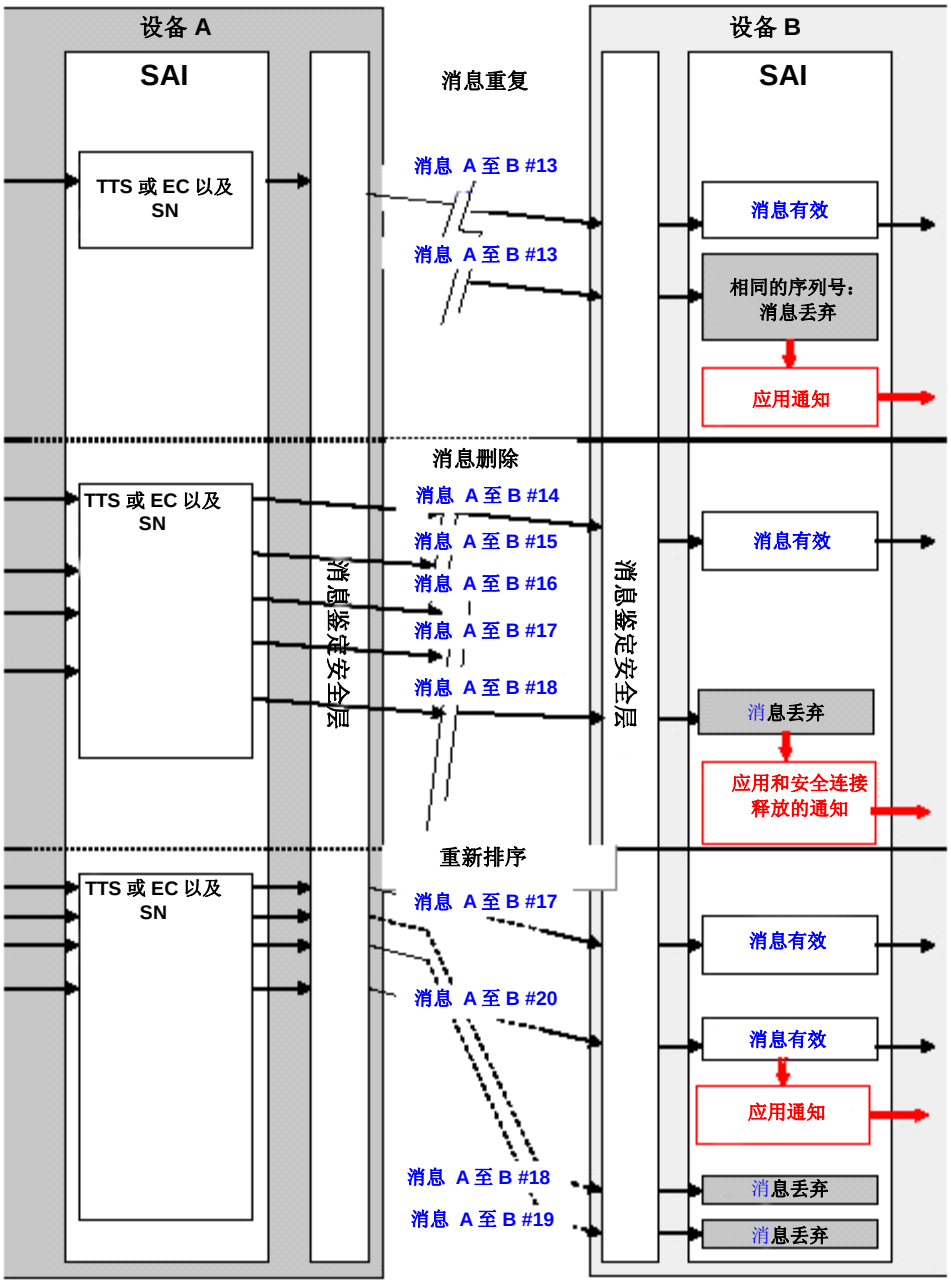

设备 A
SAI
TTS 或 EC 以及 SN
消息重复
消息 A 至 B #13
消息 A 至 B #13
设备 B
SAI
消息有效
相同的序列号：消息丢弃
应用通知
消息删除
TTS 或 EC 以及 SN
消息 A 至 B #14
消息 A 至 B #15
消息 A 至 B #16
消息 A 至 B #17
消息 A 至 B #18
消息鉴定安全层
消息鉴定安全层
消息有效
消息丢弃
应用和安全连接释放的通知
重新排序
TTS 或 EC 以及 SN
消息 A 至 B #17
消息 A 至 B #20
消息 A 至 B #18
消息 A 至 B #19
消息有效
消息有效
应用通知
消息丢弃
消息丢弃

图 11：序列错误

3.4.8. TTS

3.4.8.1. 简介

3.4.8.1.1. 时间戳处理过程基于发送方和接收方之间的时钟偏移估算。通信双方应在建立安全连接后和交换应用数据前，初始化时钟偏移估算。

3.4.8.1.2. 两个设备之间的时钟偏移估算值一旦确定，就能够以一种安全的方式估算应用数据的时间有效性，而无须对远程设备当前周期和（或）网络特性作任何设定。

3.4.8.1.3. 时间戳调整过程（即时钟偏移更新过程）由两个设备之间 5 条消息的交换组成。通过该过程，每个设备都能估算它的内部时钟和对等设备内部时钟之间的时钟偏移。

3.4.8.1.4. 所有应用消息都要标记 TTS。

3.4.8.1.5. 第 2 和第 3 个时间戳仅用于计算和更新时钟偏移估算值。第 1 个时间戳不仅用于估算和更新时钟偏移，也用于计算应用数据的时间有效性。

3.4.8.1.6. 接收到消息后，接收方利用时钟偏移估算值，将发送方所传送的时间戳调整为接收方时钟，然后再计算应用数据的时间有效性。

3.4.8.1.7. 发送方所发送的时间戳消息中的“过零点”，由接收方处理。

3.4.8.1.8. 要定期使用时钟偏移更新程序，对两个系统之间的时钟偏移估算值进行更新。

3.4.8.2. TTS 格式

3.4.8.2.1. 时间戳以 32 位 Big Endian 进行编码。

3.4.8.2.2. TTS 最低有效位的时间值等于 10 ms。

3.4.8.3. 时钟偏移估算原理

3.4.8.3.1. 时钟偏移估算值一旦确定，一个设备就能够对其本身与对等设备之间所交换消息的时间有效性进行估算。

3.4.8.3.2. 在安全连接初始化时进行时钟偏移估算。时钟偏移更新程序应在第一条应用消息交互前执行。

3.4.8.3.3. 由发起安全连接的设备启动时钟偏移更新程序。

3.4.8.3.4. 通过两个实体之间 5 个消息的交换来估算时钟偏移。发起时钟偏移更新程序的实体被称为“发起方”，而另一个实体被称为“应答方”。

3.4.8.3.5. 发起方使用前 2 个消息来计算两个设备之间的最大和最小时钟偏移。

3.4.8.3.6. 应答方使用第 2 和第 3 个消息来计算两个设备之间的最大和最小时钟偏移。

3.4.8.3.7. 第 4 和第 5 个消息用来验证时钟偏移估算值的有效性。

3.4.8.4. 时钟偏移更新消息

3.4.8.4.1. OffsetStart 消息结构如下所示：

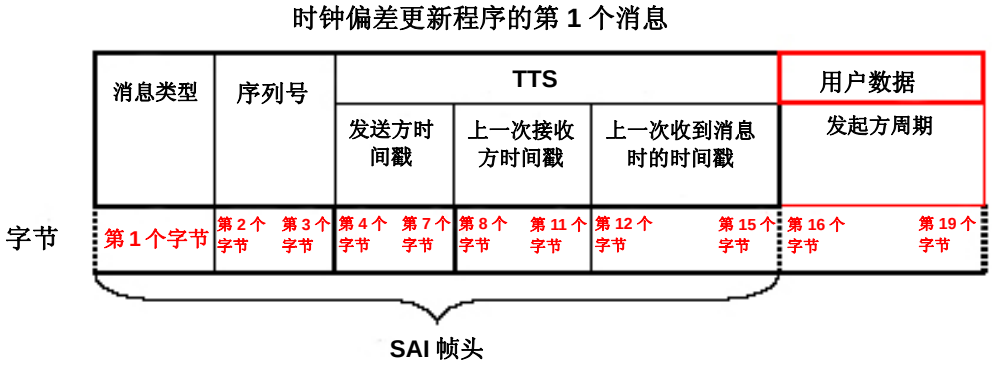


图 12：OffsetStart 消息

3.4.8.4.2. OffsetStart 消息分为不同的域，分别是：

- 消息类型：1（十六进制值）；
- 序列号；
- 发送方时间戳：此域定义了进行时钟偏移估算的发起方时间戳；
- 上一次接收方时间戳：此域给出了上一次应答方传送给发起方的时间戳。由于没有之前应答方给出的时间戳，此值设为“0”；
- 上一次收到消息时的时间戳：此域给出了上一次从应答方处接收

到消息时的时间值。由于没有之前应答方给出的应用消息，此值设为“0”；

- 发起方周期：此域给出了发起方向应答方进行周期性传送消息的传送周期。如果消息不是周期性传送，则将此值设为“0”。“发起方周期”使用的格式和时间分辨率与时间戳域相同。

3.4.8.4.3. OffsetAnsw1 消息结构如下所示：

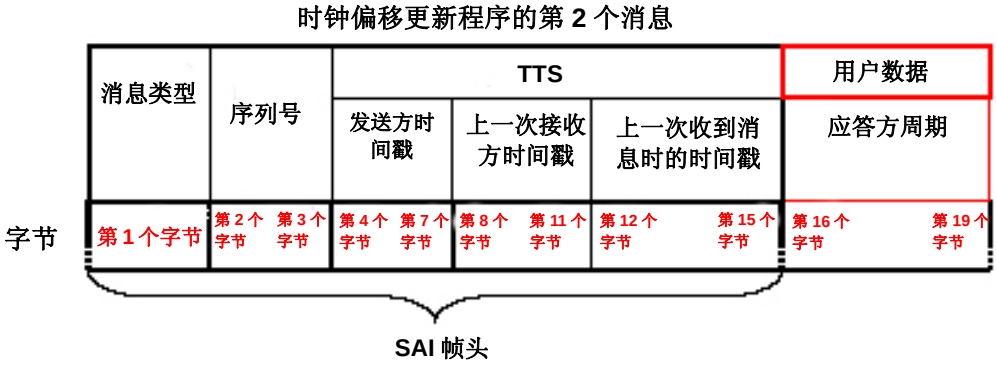


图 13：OffsetAnsw1 消息

3.4.8.4.4. OffsetAnsw1 消息分为不同的域，分别是：

- 消息类型：2（十六进制值）
- 序列号
- 发送时间戳：此域定义了应答方的时间戳。
- 上一次接收方时间戳：此域给出了发起方上次传送给应答方的发起方时间戳。
- 上一次收到消息时的时间戳：此域给出了应答方接收到时钟偏移更新过程的第一个消息时的应答方时间戳。
- 应答方周期：此域给出了应答方向发起方进行周期性传送消息的传送周期。如果消息不是周期性传送，则将此值设为“0”。 应答方周期使用的格式和时间分辨率与时间戳域相同。

3.4.8.4.5. OffsetAnsw2 消息结构如下所示：

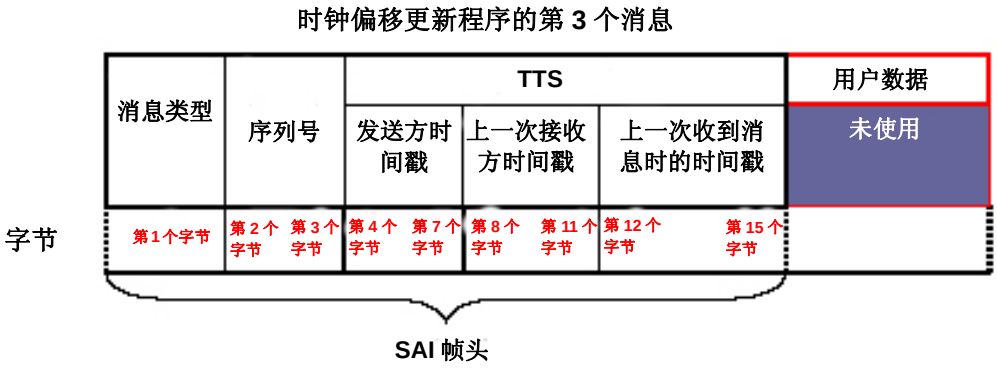


图 14：OffsetAnsw2 消息

3.4.8.4.6. OffsetAnsw2 消息分为不同的域，分别是：

- 消息类型：3（十六进制值）
- 序列号
- 发送方时间戳：此域定义了发起方的时间戳。
- 上一次接收方时间戳：此域给出了应答方上次传送给发起方的时间戳。
- 上一次收到消息时的时间戳：此域给出了发起方上次从应答方处接收消息时的时间戳（即 OffsetAnsw1 消息的接收时间）。

3.4.8.4.7. 消息结构如下所示：

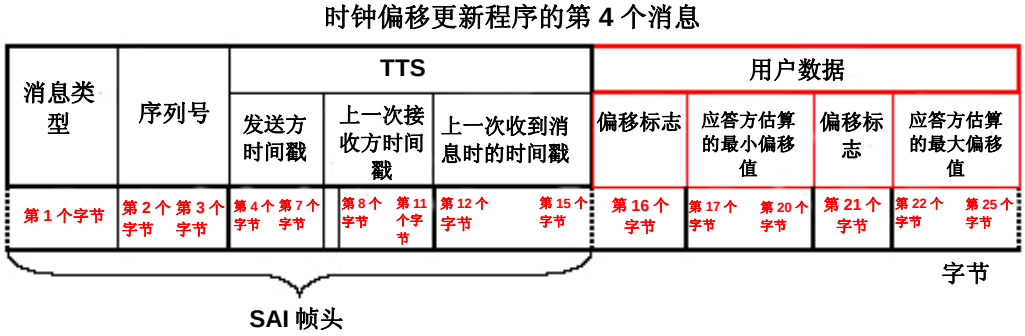


图 15：OffsetEst 消息

3.4.8.4.8. OffsetEst 消息分为不同的域，分别是：

- 消息类型：4（十六进制值）

- 序列号
- 发送方时间戳：此域定义了应答方的时间戳。
- 上一次接收方时间戳：此域给出了发起方上次传送给应答方的发起方时间戳。
- 上一次收到消息时的时间戳：此域给出了应答方接收到的时钟偏移更新过程的第 3 个消息时的时间戳。
- 偏移标志：此域给出了应答方所估算的最小偏移值的正负标志。采用 Big Endian 编码。“0”则表示偏移为正值或空值，“1”表示偏移为负值。
- 应答方估算的最小偏移值：此域给出了应答方所计算的最小偏移值。使用的格式和时间分辨率与时间戳域相同。
- 偏移标志：此域给出了应答方所估算的最大偏移值的正负标志。采用 Big Endian 编码。“0”则表示偏移为正值或空值，“1”表示偏移为负值。
- 应答方估算的最大偏移值：此域给出了应答方所计算的最大偏移值。使用的格式和时间分辨率与时间戳域相同。

3.4.8.4.9. OffsetEnd 消息结构如下所示：

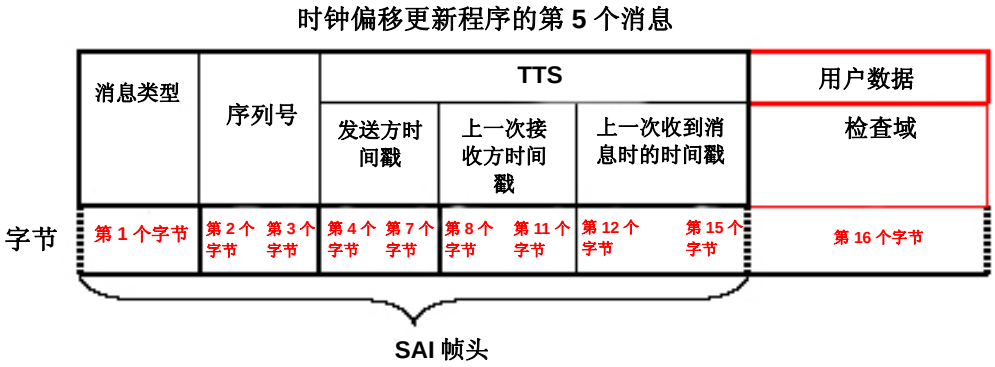


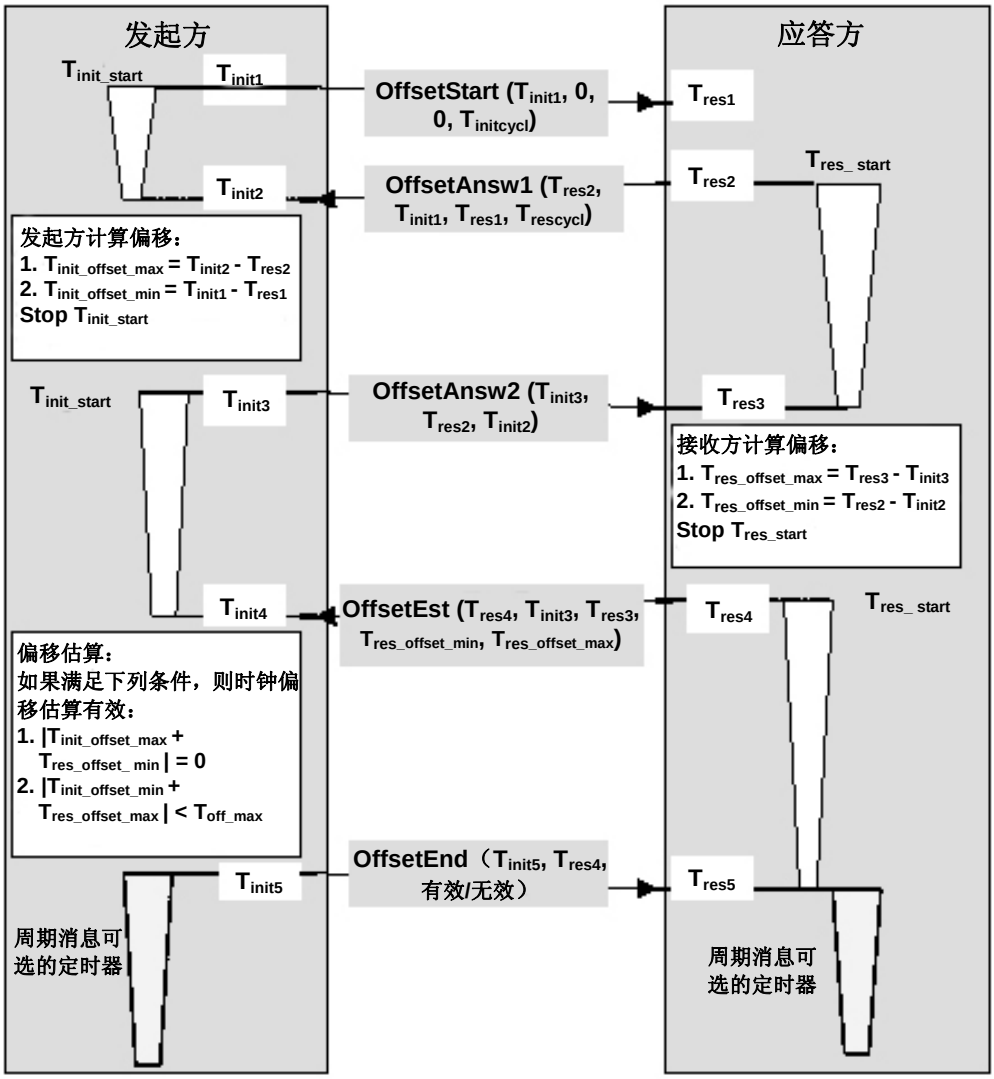
图 16：OffsetEnd 消息

3.4.8.4.10. OffsetEnd 消息分为不同的域，分别是：

- 消息类型：5（十六进制值）
- 序列号：消息编号
- 发送方时间戳：此域定义了应答方的时间戳。
- 上一次接收方时间戳：此域给出了应答方上次传送给发起方的时间戳。
- 上一次收到消息时的时间戳：此域给出了发起方上次从应答方处接收到消息时的时间戳（即 OffsetEst 消息的接收时间）。
- 检查域：此域给出了对时钟偏移值进行检查的结果。经过比较，如时钟偏移值有效，则将检查域值设为“1”，如果无效，则检查域值设为“0”。

3.4.8.5. 时钟偏移更新过程

3.4.8.5.1. 时钟偏移更新过程详见下图所示：



图例说明：

- T_{init_start} 和 T_{res_start}：初始化定时器。如果在收到 Offset Answers 消息前计时终止，则将释放并重启安全连接。
- T_{initX} 和 T_{resX}：发起方和应答方的第 X 个时间戳。
- $T_{rescycl}$ 和 $T_{initcycl}$：应答方和发起方传送消息的周期。如果非周期性发送，则此参数设为“0”。
- $T_{res\_offset\_max}$ 和 $T_{init\_offset\_max}$：发起方和应答方估算的最大偏移值。
- $T_{res\_offset\_min}$ 和 $T_{init\_offset\_min}$：发起方和应答方估算的最小偏移值。
- T_{off_max}：时钟偏移估算值之间的最大差异。
- 有效/无效：时钟偏移检查的结果。

图 17：时钟偏移更新过程

3.4.8.5.2. 发起方通过发送 OffsetStart 消息启动时钟偏移更新程序。

3.4.8.5.3. 发送 OffsetStart 消息时，发起方启动定时器 T_{init_start}。如果 T_{init_start} 终止时，发起方仍未接收到 OffsetAnsw 消息，则根据错误处理程序（见第 5.4.10 节）对该错误进行处理。

3.4.8.5.4. 在接收到 OffsetStart 消息后，应答方通过发送 OffsetAnsw 消息进行应答。

3.4.8.5.5. 发送 OffsetAnsw 消息时，应答方启动定时器 T_{res_start}。如果 T_{res_start} 终止时，应答方仍未接收到 OffsetAnsw2 消息，则根据错误处理程序（见第 5.4.10 节）对该错误进行处理。

3.4.8.5.6. 如果发起方在 T_{init_start} 终止前接收到 OffsetAnsw 消息，则发起方对其与应答方之间的时钟进行最大和最小偏移值的估算。

3.4.8.5.7. 发起方向应答方发送 OffsetAnsw2 消息。发送 OffsetAnsw2 消息时，发起方启动定时器 T_{init_start}。如果 T_{init_start} 终止时，发起方仍未接收到 OffsetEst 消息，则根据错误处理程序（见第 5.4.10 节）对该错误进行处理。

3.4.8.5.8. 如果应答方在 T_{res_start} 终止前接收到 OffsetAnsw2 消息，则应答方对其与发起方之间的时钟进行最大和最小偏移值的估算。

3.4.8.5.9. 应答方向发起方发送 OffsetEst 消息。发送 OffsetEst 消息时，应答方启动定时器 T_{res_start}。如果 T_{res_start} 终止时，应答方仍未接收到 OffsetEnd 消息，则根据错误处理程序（见第 5.4.10 节）对该错误进行处理。

3.4.8.5.10. 如果发起方在 T_{init_start} 终止前接收到 OffsetEst 消息，则发起方对其与应答方分别估算的最大和最小偏移值进行检查。然后，发起方通过 OffsetEnd 消息将检查结果发送给应答方。如果检查中发现错误，则根据错误处理程序（见第 5.4.10 节）对该错误进行处理。

3.4.8.5.11. 在接收到 OffsetEnd 消息后，应答方将对时钟偏移估算的结果进行检查。如果检查中发现错误，则根据错误处理程序（见第 5.4.10 节）对该错误进行处理。

3.4.8.5.12. 如果时钟偏移估算值有效，并且消息为某一方向或双向周期传输，则可以选择启动一个 TTS 周期定时器。此定时器非安全相关，仅用于在早期阶段检测丢失的周期消息。

3.4.8.5.13. 每接收到一个消息时应复位该 TTS 周期定时器。如果定时器结束时仍未收到消息，则根据错误处理程序（见第 5.4.10 节）进行处理。

3.4.8.6. 时间戳标记和检查

3.4.8.6.1. 应用层之间传送应用数据的消息结构如下图所示：

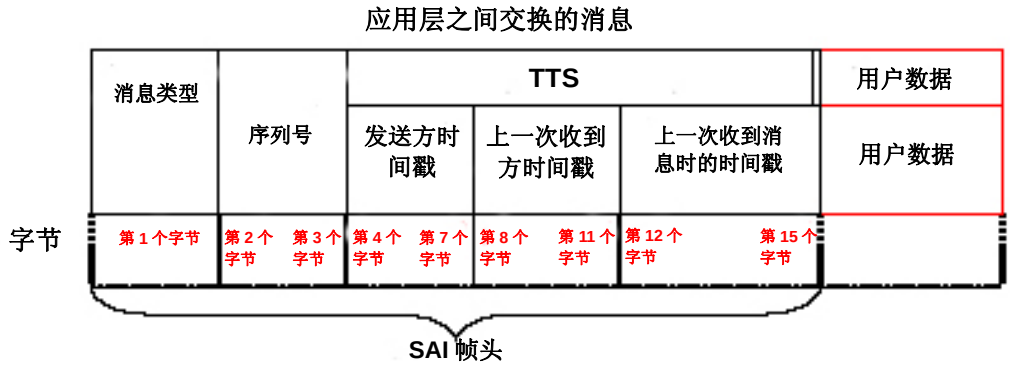


图 18：应用数据消息

3.4.8.6.2. 此消息分别包括以下数据域：

- 消息类型：6（十六进制值）；
- 序列号；
- 发送时间戳：定义发送方的时间戳；
- 上一次接收方时间戳：此域给出接收方上一次传送给发送方的时间戳；
- 上一次收到消息时的时间戳：此域给出上一次从对等实体处接收到消息时的时间戳；

- 用户数据：用户数据域。

3.4.8.6.3. 时间戳原理包括：

- 将发送方本身的时间戳列入发送时间戳域。
- 将上一次从接收方处接收的时间戳列入上一次接收方时间戳域。
- 将发送方上一次从接收方处接收消息时的时间戳列入上一次接收消息时的时间戳域。

3.4.8.6.4. 发送方所发送的时间戳消息中的“过零点”，由接收方处理。下图说明了 TTS 的原理：

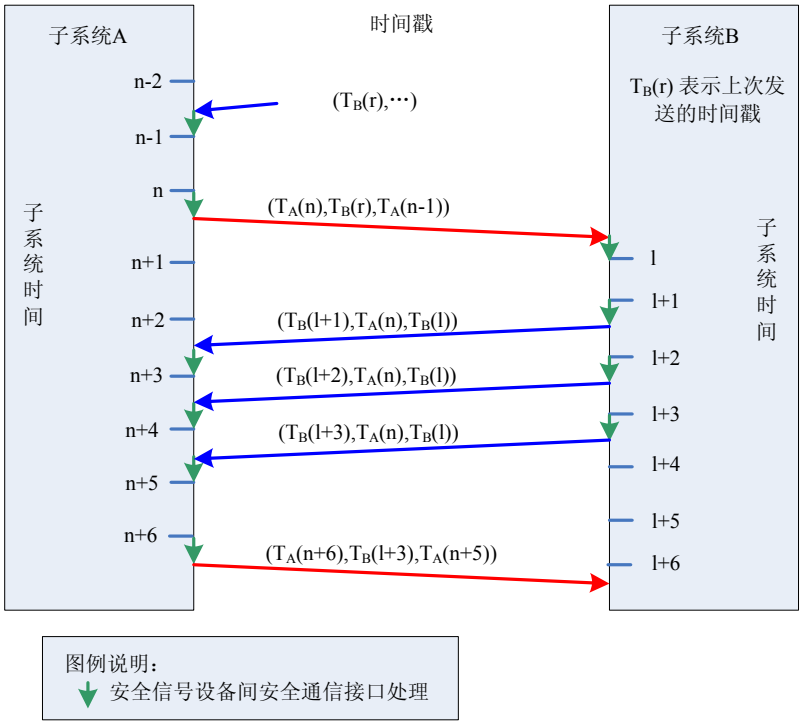


图 19：TTS 原理

3.4.8.6.5. 如果时间戳过程中出错，则根据错误处理程序（见第 5.4.10 节）对该错误进行处理。

3.4.8.6.6. 时间戳更新程序完成后，时钟偏移的最大值和最小值既已确定，并可用于估算应用数据消息的时间有效性。

3.4.8.6.7. 发送方应根据发送方时钟、上次从对等实体处接收到的时间戳以及上次从对等实体处接收消息的时间戳，对发送消息进行时间戳标记。

3.4.8.6.8. 接收方接收到应用数据消息时，应使用接收方估算的最小偏移值以及附加处理延迟的估计值，将消息的发送时间按接收方时钟进行估算。计算表达式如下：

$$T_{receiver}=T_{time\_stamp\_sender} - \triangle T_{extra\_delay} + T_{rec\_offset\_min}$$

3.4.8.6.9. 消息的时间有效性根据接收方估算的消息发送时间与接收到消息时的时间之间的差值（$T_{rec\_current} - T_{receiver}$）进行判断。

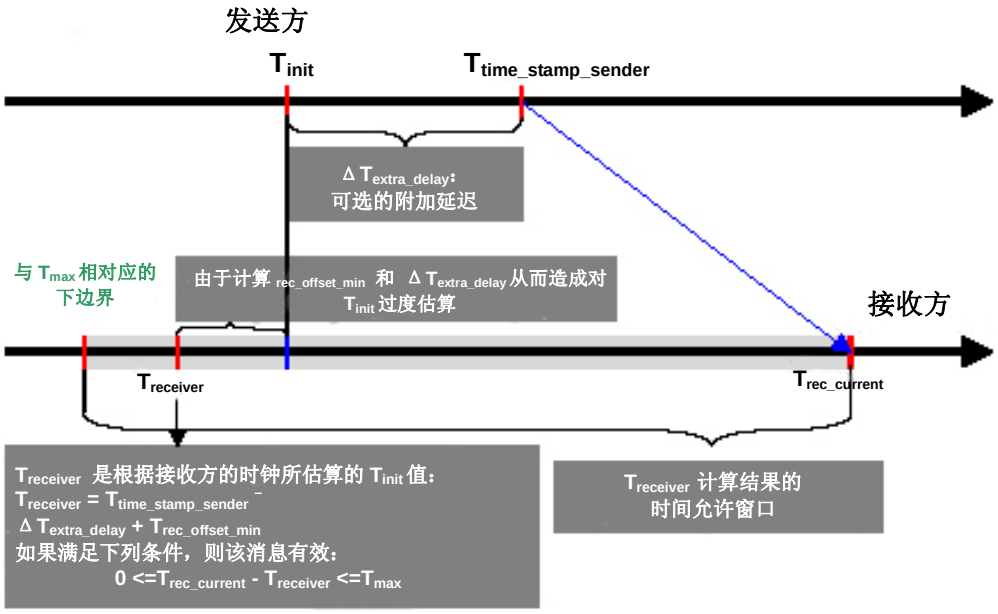
3.4.8.6.10. 如果根据此差值判断传输数据的时间有效，则表示延迟是可以接受的。时间有效性由 T_{max} 定义。T_{max} 为双方协定的配置参数。如果消息的延迟不可接受，则拒绝该消息，并根据错误处理程序（见第 5.4.10 节）对该错误进行处理。

3.4.8.6.11. T_{max} 参数与对等实体发送数据的最大有效时间相一致。

3.4.8.6.12. 通过 T_{max} 的值，可以检测到传输时间的增大，以及双方设备之间的正向时钟偏移。

3.4.8.6.13. 差值结果（$T_{rec\_current} - T_{receiver}$）应为正值。如果为负值，则根据错误处理程序（见第 5.4.10 节）对该错误进行处理。

3.4.8.6.14. 时间有效性检查过程如下图所示。



图例说明：

$T_{time\_stamp\_sender}$：发送方时间戳（TTS 中发送方时间戳域）。

$T_{receiver}$：根据接收方时钟估算的应用数据的传输时间。

$\triangle T_{extra\_delay}$：发送方子系统处理应用数据的附加延迟的总和。

$T_{rec\_offset\_min}$：接收方估算的最小偏移值。

T_{max}：最大有效时间。

$T_{rec\_current}$：接收到消息时接收方的当前时间。

T_{init}：根据发起方时钟，应用数据的传送时间。

图 20：时间戳计算

3.4.8.7. 时钟偏移更新

3.4.8.7.1. 时钟偏移应根据设定的时间，定期进行更新。

3.4.8.7.2. 时钟偏移更新程序详见下图所示：

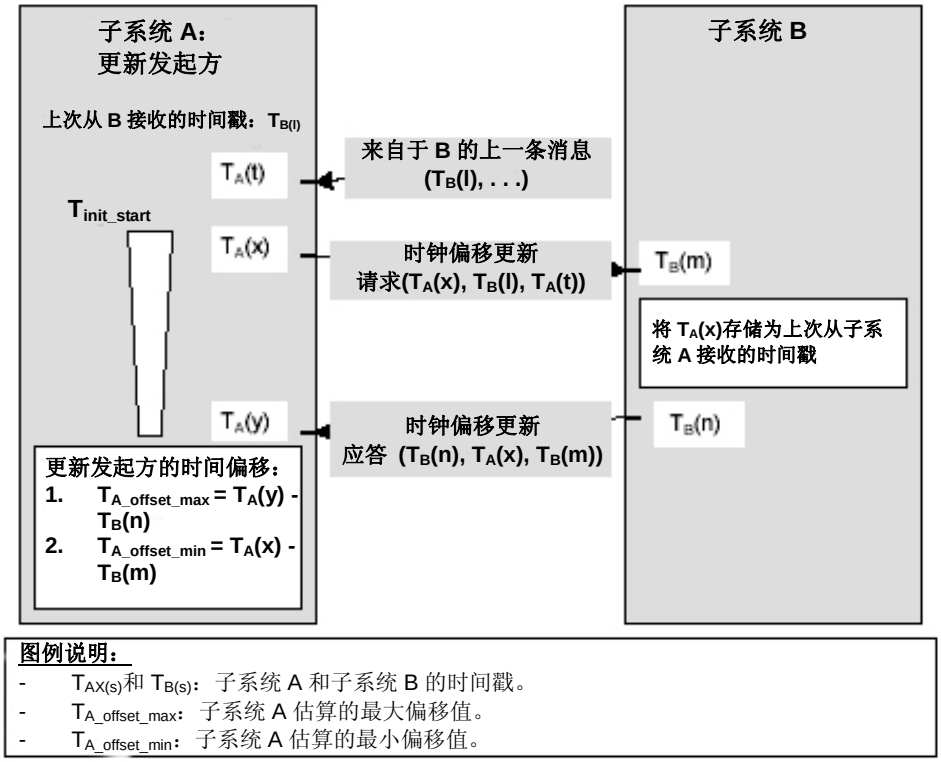


图 21：时钟偏移更新程序

3.4.8.7.3. 时钟偏移更新消息的结构与应用数据消息（无用户数据域）的结构一致。

3.4.8.7.4. 定时器 T_{init_start} 用于监督执行时钟偏移更新程序所需的时间。

3.4.8.7.5. 如果执行时钟偏移更新程序所需时间大于 T_{init_start} 值，则表示传送延迟时间很长，不能有效更新时钟偏移。

3.4.8.7.6. 如果时钟偏移更新失败，则根据错误处理程序（见第 3.4.10 节）对该错误进行处理。

3.4.8.7.7. 在接收到时钟偏移更新的应答后，发起方要检查它是否是对偏移更新请求的应答：该消息中的上一次接收方时间戳应等于偏移更新请求消息中的发送方时间戳。

3.4.9. EC 防御技术

3.4.9.1. 总体概述

3.4.9.1.1. 根据 EN 50159-2 的要求，应用数据消息要防护延迟威胁，EC 的安全防御技术即可用于保护消息的时效性。

3.4.9.1.2. 从对等实体接收到的每一条消息都含有帧头，使用帧头里的 EC 计数可以实现对延迟威胁的防护。

3.4.9.1.3. EC 需要具有固定的周期值。

3.4.9.1.4. 延迟威胁的检测通过对接收到的消息中的 EC 计数值和本地计算的 EC 计数期望值进行比较实现。

3.4.9.1.5. 安全防御技术保障了从上次传输消息开始指定的时间区域内传输消息（如果应用程序没有要求的话，不包含应用程序数据）的有效性，从而将消息传输转换为伪周期模式。这样就可以管理在安全连接另一端的超时数据接收。

3.4.9.2. EC 计数格式

3.4.9.2.1. EC 计数使用 32 位 Big Endian 编码方式。

3.4.9.2.2. EC 计数的值从 0 到 4294967295。

3.4.9.2.3. EC 计数在通信各方向上是独立的。

3.4.9.2.4. EC 计数的无须初始化。发送方发送第一条消息中的任意 EC 值接收方都应接受。

3.4.9.2.5. 如果 EC 计数值未达到最大值，EC 计数将每周期递增 1。

3.4.9.2.6. 一旦 EC 计数值达到最大值，下一个 EC 的计数值应设为 0。

3.4.9.2.7. 在接收方和发送方的子系统中，SN 的处理与 EC 计数的处理是相互独立的。

3.4.9.3. EC 防御技术的初始化过程

3.4.9.3.1. 两实体间的 MASL 层建立安全连接后，双方 SAI 层将通过两条消息交互 EC 防御技术所需的参数。

3.4.9.3.2. 5.4.9.3.2 初始化过程如下图所示

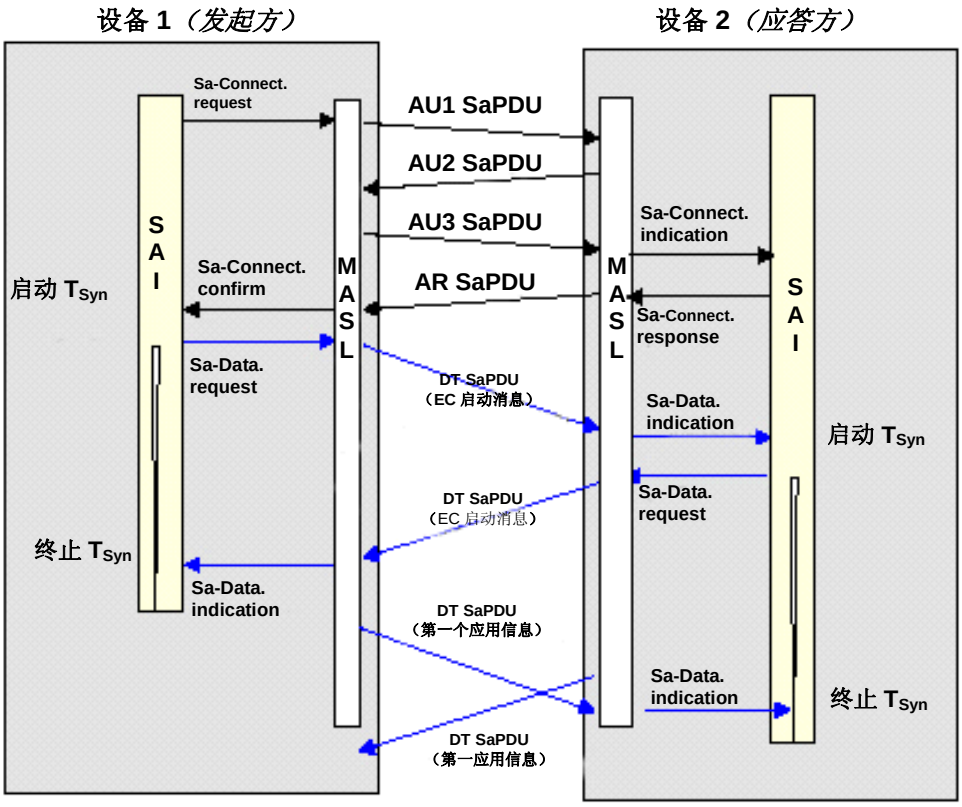


图 22： EC 初始化过程

3.4.9.3.3. 两个实体通过 AU1，AU2，AU3 和 AR SsPDU's 建立安全连接后，连接发起方应发送一条 EC 启动消息，其中包括 EC 计数的初始值和 EC 周期值。

3.4.9.3.4. 其对等实体应通过 EC 启动消息做出应答，其中包括 EC 计数的初始值和 EC 周期值。

3.4.9.3.5. 在从远端接收到 EC 启动消息之后，发起方可以开始发送应用消息，并应使用 EC 防御技术。应答方在发送 EC 启动消息之后的下一个周期可以开始发送应用消息，并应使用 EC 防御技术。

3.4.9.3.6. 每一个对等实体将使用定时器(T_{syn})，来检测不可接受的初始化延迟：

3.4.9.3.7. 发起方的 SAI：定时器自发送 EC 启动消息起开始计时，至接收到应答方执行周期开始消息时停止计时；

3.4.9.3.8. 应答方的 SAI：定时器自发送 EC 启动消息起开始计时，至接收到对方第一个 DT SaPDU（包含第一个应用程序消息）时停止计时。

3.4.9.3.9. 如果一个定时器在接收到预期消息前终止，则根据错误处理程序（见第 5.4.10 节）对该错误进行处理。

3.4.9.3.10. EC 启动消息的结构由下图说明：

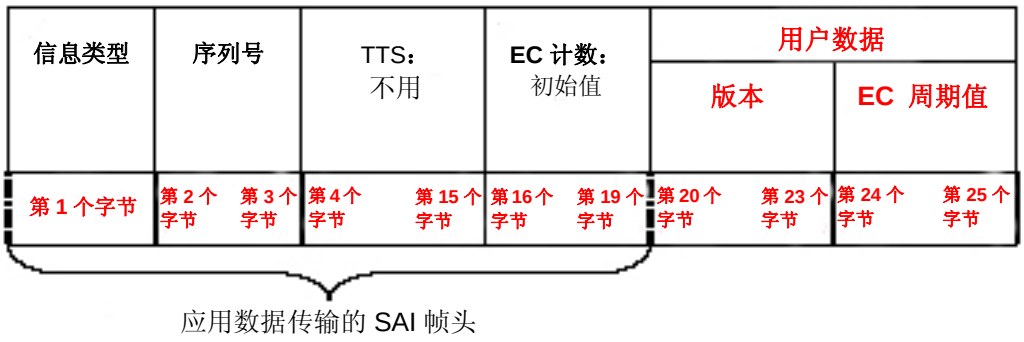


图 23：EC 启动消息

3.4.9.3.11. EC 周期值使用 2 个字节 Big Endian 编码方式编码，取值范围为 0 到 65535。EC 最低有效位时间值等于 1 毫秒（精度为 1 毫秒）。

3.4.9.3.12. 版本域将按 4 个字节 Big Endian 编码方式编码。版本域被用来验证本地和远程的 SAI 软件版本的一致性。

3.4.9.4. EC 计数检查过程

3.4.9.4.1. 一旦 EC 防御技术初始化过程成功完成，此程序将检查对等实体发送的应用数据是否过时。

3.4.9.4.2. 从远程子系统接收的消息中包含 EC 计数值。 通过对此数值与 EC 的期望值（用 Ex 表示）进行比较实现安全防御。

3.4.9.4.3. EC安全防御技术的第一步在于确定EC的期望值（Ex）。为了计算Ex，在EC启动消息之后，每一个SAI实体将用以下公式计算本地和远程EC周期的比值R[1]。

R = 接收方 EC 周期/发送方 EC 周期

注：R 是一个实数。

3.4.9.4.4. 在接收方的每个执行周期都应该更新 EC 的期望值（Ex），计算公式如下：

下一个 Ex = 当前 Ex + R

注：Ex是一个实数[2]。

3.4.9.4.5. 接收方每周期结束时应计算当前状态，而此当前状态仅依赖整形变量 Δ，Δ 使用以下公式计算：

Δ = (Ex –当前周期收到的最后一条消息的 EC 计数值) 向下取整

[1] 比值 R 也可以作为设备配置参数设为固定值。在设备启动前双方必须对 R 参数选择为静态设置还是动态设置进行确定并达成一致。

[2] R 和 Ex 均为实数，同时 EC 周期和 EC 计数均为整数。

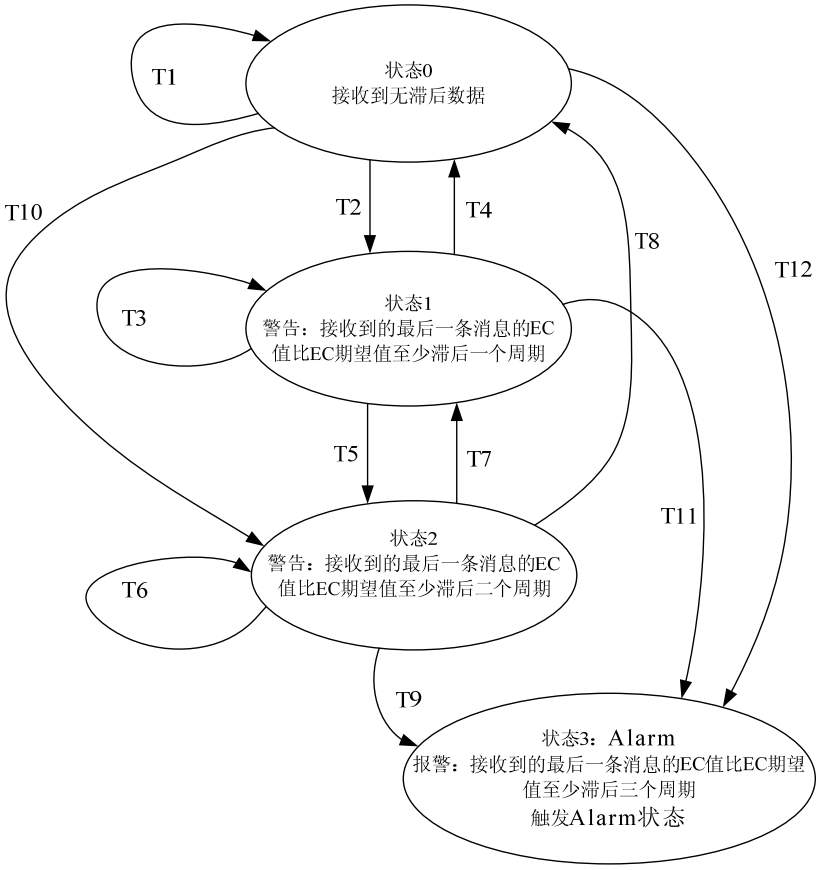
3.4.9.4.6. 如果在某个执行周期中没有收到任何消息，接收方应使用前一个周期得到的最后一条消息中包含的 EC 值来计算 Δ。如果一个周期内收到多条消息，应采用接收到的最后一条消息的 EC 值进行计算。

3.4.9.4.7. 在每一个 EC 上由接收方 SAI 应用下图描述的状态转移图。如果 Δ 大于或等于警报状态（例如下图所描述的状态 3），将会触发错误报警，然后进入错误处理程序处理，（详见 5.4.10）。在以下的例子里，从对等实体接收到的最后一条消息中的 EC 计数比 Ex 有至少 3 个发送周期延迟。如果 Δ 大于或者等于警报状态，该周期所有接收的消息将被删除。

3.4.9.4.8. 当发起方从应答方上接收到EC启动消息，发起方将来自应答方消息中的EC计数初始化Ex，在同一个EC之中，使用 5.4.9.4.4[3] 的公式进行更新。

3.4.9.4.9. 当应答方从发起方接收到第一个应用数据消息时，应答方将使用来自发起方消息中的EC计数初始化Ex，在同一个EC中，使用 5.4.9.4.4[4] 的公式进行更新。

[3] 如果在 EC 启动后的一个周期内收到多条的应用数据消息，最后一条消息中的 EC 将被用来更新下一个 Ex 。
[4] 如果在一个周期内收到多条的应用数据消息，最后一条消息中的 EC 将被用来更新下一个 Ex。



对转移的描述：

- T1：接收方“执行周期”结束，Δ 小于或者等于 0。
- T2：接收方“执行周期”结束，Δ 等于 1。
- T3：接收方“执行周期”结束，Δ 等于 1。
- T4：接收方“执行周期”结束，Δ 小于或者等于 0。
- T5：接收方“执行周期”结束，Δ 等于 2。
- T6：接收方“执行周期”结束，Δ 等于 2。
- T7：接收方“执行周期”结束，Δ 等于 1。
- T8：接收方“执行周期”结束，Δ 小于或者等于 0。
- T9：接收方“执行周期”结束，Δ 大于或者等于 3。
- T10：接收方“执行周期”结束，Δ 等于 2。
- T11：接收方“执行周期”结束，Δ 大于或者等于 3。
- T12：接收方“执行周期”结束，Δ 大于或者等于 3。

图 24：EC 状态机

3.4.9.5. EC 防御技术的后续过程

3.4.9.5.1. 时钟的长期漂移产生的影响及由EC周期的近似值所引起的错误都可能导致EC周期 [5] 十分缓慢的改变（相对于EC启动消息里携带的EC值），也导致R错误（相对于在初始化阶段末期的计算值）。经过一段时间，将会导致接收到：

- 比预期更多的消息，因为发送方子系统比预期的要快（或者接收方子系统比预期的要慢：结果一样）：这将导致长时间滞留在“伪状态 -1”，接着又处于“伪状态 -2”等等，（所有这些伪状态被归入状态 0，因为到达的数据是新的）。处于这种状态下时，基本的 EC 计数检查程序不能及时发现真正的延迟。
- 比预期更少的消息，就算没有消息延迟，因为发送方子系统比预期的要慢（或者接收方子系统比预期的要快：结果一样）： 这将导致状态机长时间滞留在状态 1，接着在状态 2 等等，直到达到报警状态。

3.4.9.5.2. 为了纠正以上所描述的影响，SAI 应采取如下措施：

- 如果 Δ 小于或者等于一个（负）值（配置参数），也就是说接收到比预期（当一个实体从对等实体接收到 EC 启动消息后，当前周期中只期望接收到该“预期”消息）更多的消息，期望 EC 计数值（Ex）使用包含在从等实体接收到的最新的消息里的 EC 来更新：

  下一个 Ex = （从当前周期中接收到的最新 EC 值）+R

- 如果状态机长时间（时间值是一个配置参数）保持在同一状态（不同于状态 0），就通过减小Ex，强制返回上一级状态（1→0，2→1）。在这种情况下，将执行传输延迟检测程序 [6]。

[5] 时钟源的永久误差（时钟精度）一般为几十个 ppm。
[6] 本程序可以在强制返回上一级状态之前或者之后被立即执行,这个选项是一个本地行为。

3.4.9.6. 用于检测传输延迟的过程

3.4.9.6.1. 因为长时间保持在同一状态下可能由传输链[7]中出现的其它问题引起，例如缓慢增长的延迟，不仅是由于R的近似值引起，在 5.4.9.5 中定义的纠正措施可能掩盖这些真实的延迟：将提供进一步的机制来检测它们。

3.4.9.6.2. 发起方子系统和应答方子系统想要检测出可能的传输延迟，需要向对等实体发送“带有ACK请求的应用消息”，该消息与正常的应用消息格式一样，只是消息类型不同（0×87 代替了 0×86）。注：这是一个包含正常用户数据的ApPDU，这条消息将被发送到对等实体[8]。

3.4.9.6.3. “带有ACK请求的应用程序消息”接收到后，子系统将在下一个周期中发送“带有ACK的应用程序消息”以应答对等实体，该消息与正常的应用消息格式一样，只是消息类型不同（0×88）。注：这是一个包含正常用户数据的ApPDU，这条消息将被发送到对等实体[9]。

3.4.9.6.4. 为了进一步的检测不可接受的传输延迟，在发送“带有 ACK 请求的应用程序消息”的子系统中使用定时器 T_{syn}，在发送“带有 ACK 请求的应用程序消息”时定时器开启，从对等实体接收到“带有 ACK 的应用程序消息”时定时器停止。

3.4.9.6.5. 如果定时器在接收到预期消息前终止，将使用错误处理程序处理该错误（详见 5.4.10）。

3.4.9.6.6. 延迟检测过程和 EC 安全防御技术（初始延迟检测）初始化过程中的延迟检测过程是一样的，只是用应用消息代替 EC 启动消息。下图描述这个过程（仅显示一个请求实体）。

[7] 在本文件中，用“传输链”有意指在两个安全对等实体间所有的传输设备和网络单元。
[8] 应用程序消息流和处理不受此机制影响。
[9] 应用程序消息流和处理不受此机制影响。

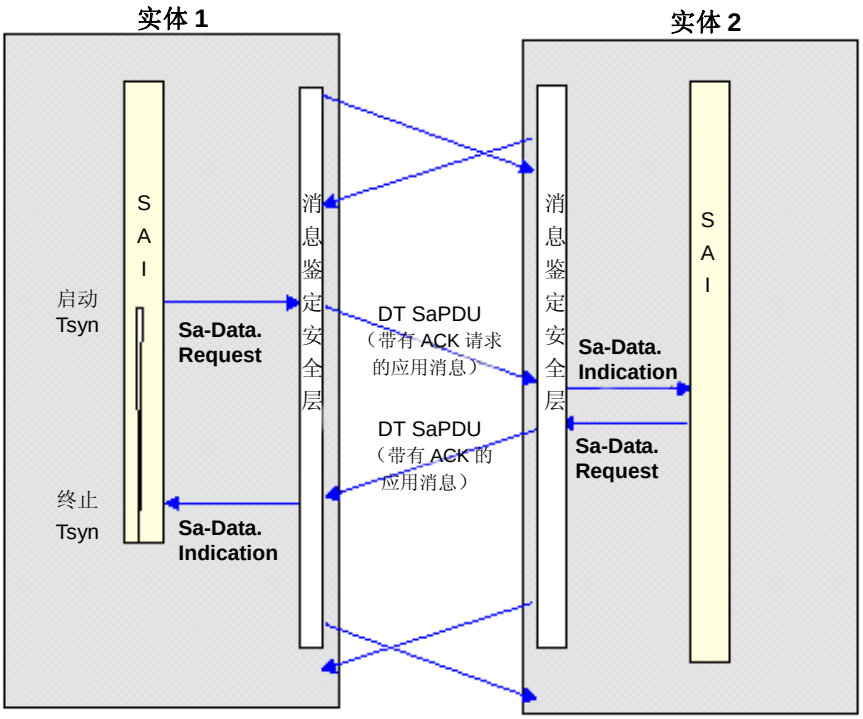


图 25：检测网络延迟的过程

3.4.10. 错误处理

3.4.10.1. 下列错误应被考虑：

- 连接建立期间的错误；
- 序列号错误；
- 与可选 TTS 周期定时器相关的错误；
- 与时钟偏移更新过程相关的错误；
- 传输延迟；
- 延迟的消息。

3.4.10.2. 除了序列错误的消息，其他错误消息中包含的应用数据应被删除（参考 3.4.7.2）。

3.4.10.3. 发生在 SAI 初始化过程中的所有错误（见时钟偏移更新过程和 EC 防御技术的初始化过程）将导致安全连接的释放。

3.4.10.4. 对于序列错误，与可选的 TTS 周期定时器有关的错误，以及与时钟偏移更新程序、传输延迟、延迟消息有关的错误，在达到一个可配置的连续的错误数(T_{succ_er})之后将释放安全连接。

3.4.10.5. 如果时钟偏移更新过程或者传输延迟的检测过程失败，将启用一个独立的错误计数器并立即重复该过程。

3.4.10.6. 在 EC 技术中，如果 Δ 值达到报警状态，连接将被释放。

3.4.10.7. 注：在何处进行错误处理（例如，应用程序，SAI，错误处理者）取决于具体实现。SAI 对需要在其外部进行的错误处理给出错误指示。

3.5. 数据配置及规则

3.5.1. 简介

3.5.1.1. 本节为建立连接提供规则并为确定 SFM 参数提供指导。

3.5.2. 连接初始化规则

3.5.2.1. 概述

3.5.2.1.1. 为在两个轨旁设备之间建立连接，必须定义一些规则以准确确定哪个设备启动连接。

3.5.2.1.2. 在同一网络中不同类型的轨旁设备需要定义不同轨旁设备间启动连接的优先规则。

3.5.2.1.3. 每个轨旁设备应知道与其建立通信的其他轨旁设备的 CTCS-ID 类型和 CTCS-ID。

3.5.2.2. 连接初始化规则

3.5.2.2.1. 网络的每个设备拥有两个表

- 被动连接表；
- 主动连接表。

3.5.2.2.2. 被动连接表包含本设备等待连接请求的远程设备的 CTCS-ID。

3.5.2.2.3. 任何被动连接表之外的远程实体的连接请求应被拒绝。

3.5.2.2.4. 主动连接表包含本设备必须对其发起连接请求的远程设备的 CTCS-ID。

3.5.2.2.5. 在启动过程中或连接丢失后，设备开始试图连接主动连接表中所列的其他设备。

3.5.2.2.6. 如果在轨旁设备初始化期间没有同主动连接表中所包含的设备建立连接，设备将周期性的尝试连接到对等设备。

3.5.2.2.7. 一般情况下，可以通过以下规则确定主动和被动连接：连接通常由后启动的设备发起。

3.5.3. TTS 参数定义

3.5.3.1. T_{max} 值

3.5.3.1.1. T_{max} 值为系统参数值，是两个轨旁设备之间传送应用数据的最大有效时间。

3.5.3.1.2. 此参数取决于以下所举因素：

- 线路布置；
- 性能要求；
- 设备实现；
- 系统规则。

3.5.3.2. ΔT_{extra_delay}

3.5.3.2.1. ΔT_{extra_delay} 参数可以用来考虑从应用数据生成到在 SAI 层形成消息时间戳之间的时间。

3.5.3.2.2. 此参数的默认值为“0”。

3.5.3.3. T_{init_start} 和 T_{res_start}

3.5.3.3.1. 这些定时器的值应等于应答请求所允许的最大时间。

3.5.3.3.2. 确定定时器值的一种方法就是使用以下规则：（余量以%表示）

3.5.3.3.3. 时间值 =（1 + 余量/100）*（2*最大传输时间+设备读取请求与应答传输之间的最大时间）

注：传输时间应考虑：

- 最大网络传输时间；
- 形成消息时间戳与网络上发送该消息之间的时间；
- 接收方请求用来读取消息的时间。

3.5.3.4. T_{off_max}

3.5.3.4.1. 该定时器的值应等于两设备间可接受的最大传输时间。

注：传输时间应考虑：

- 最大网络传输时间；
- 形成消息时间戳与网络上发送该消息之间的时间；
- 接收方请求用来读取消息的时间。

3.5.3.5. 可选 TTS 周期定时器

3.5.3.5.1. 周期消息可选定时器值可使用以下规则估算：

3.5.3.5.2. 可选 TTS 周期定时器 = 对等设备周期 + 最大允许传输时间

注意：传输时间应考虑：

- 最大网络传输时间；
- 形成消息时间戳与网络上发送该消息之间的时间；
- 接收方请求用来读取消息的时间。

3.5.3.6. 时钟偏移更新周期

3.5.3.6.1. 时钟偏移更新用于防止时钟固有的偏移。

3.5.3.6.2. 对每次更新，假设时间精度参数为： *pr*%。

3.5.3.6.3. 对于一个系统，可定义的最大时钟偏移容量：假设为 *X* sec。

3.5.3.6.4. 时钟偏移更新周期可应用以下公式计算：（余量为 *Y*%）：

周期（sec）=（100 * *X* sec/*Pr*）/（1+*Y*/100）

3.5.3.6.5. 例如：Pr=0.1，Y=100%且 X=0.5sec→周期=250sec

3.5.4. EC 参数定义

3.5.4.1. T_{syn}

3.5.4.1.1. 该定时器值应等于应答请求所允许的最大时间。

3.5.4.1.2. 确定该定时器值的一种方法就是使用以下规则：（余量以%表示）：

3.5.4.1.3. 时间值 =（1 + 余量/100）*（2*最大传输时间+设备读取请求与应答传输之间的最大时间）

注：传输时间应考虑：

- 最大网络传输时间；
- 形成消息时间戳与网络上发送该消息之间的时间；
- 接收方请求用来读取和处理消息并产生所需响应的时间。

3.5.4.2. 报警状态

3.5.4.2.1. “报警状态”值是一个系统参数。该参数衡量在两个轨旁设备之间传输的应用数据的最大有效时间。

3.5.4.2.2. 该参数取决于以下所举因素：

- 线路布置；
- 性能要求；
- 设备实现；
- 系统规则。

3.5.4.3. 版本

3.5.4.3.1. 目前只定义一个版本值：01（十六进制）。

3.5.4.4. 传输延迟检测过程

3.5.4.4.1. 传输延迟检测过程应周期性重复。

3.5.4.4.2. 使用与定义可选 TTS 周期定时器类似的方法确定此周期。

3.5.5. 错误处理指导

3.5.5.1. 参数 N

3.5.5.1.1. 如果不允许消息丢失，则 N 设置为 1。

3.5.5.1.2. 对于特定项目和每一个连接都需要制定一个 N 值。

3.5.5.1.3. 如果要连接参数 N 不相同的安全设备，对于两个设备的连接应使用限制更严格的 N 值。

3.5.5.2. 参数 T_{succ_er}

3.5.5.2.1. 如果 D 类服务在 ALE 层中使用并在至少是两个独立的物理链路上实现，则应用独立的路由向对等 SAI 发送两次应用数据。因为在这种情况下，在两个物理链路上的较大干扰导致 SAI 层的错误，推荐在首次错误发生时释放连接：$T_{succ\_er} = 1$。

3.5.5.2.2. 在其他配置中（A 类服务与 D 类服务在一个物理链路上），应用数据仅在一个物理链路上的两个实体之间交换。因为在这种情况下，在这个链路上的一个干扰将导致接收 SAI 时出现错误，推荐在两次连续的错误之后释放连接：$T_{succ\_er} = 2$。如果使用高可用性的网络，则 T_{succ_er} 值可设置为“1”。

3.6. TTS 示例

3.6.1.1. TTS 技术如下图所示：

所有数值以时间戳格式表示（单位 10 msec）。

传输时间的容错窗口：0 到 50 之间。

时钟之间的真正偏移：850（8.5 sec）。当 A 的时钟等于 850 时，B 的时钟等于 0。

A 的周期 ＝50＝>消息输入及产生应答的时间在 50 到 100 之间。

B 的周期 ＝30＝>消息输入及产生应答的时间在 30 到 60 之间

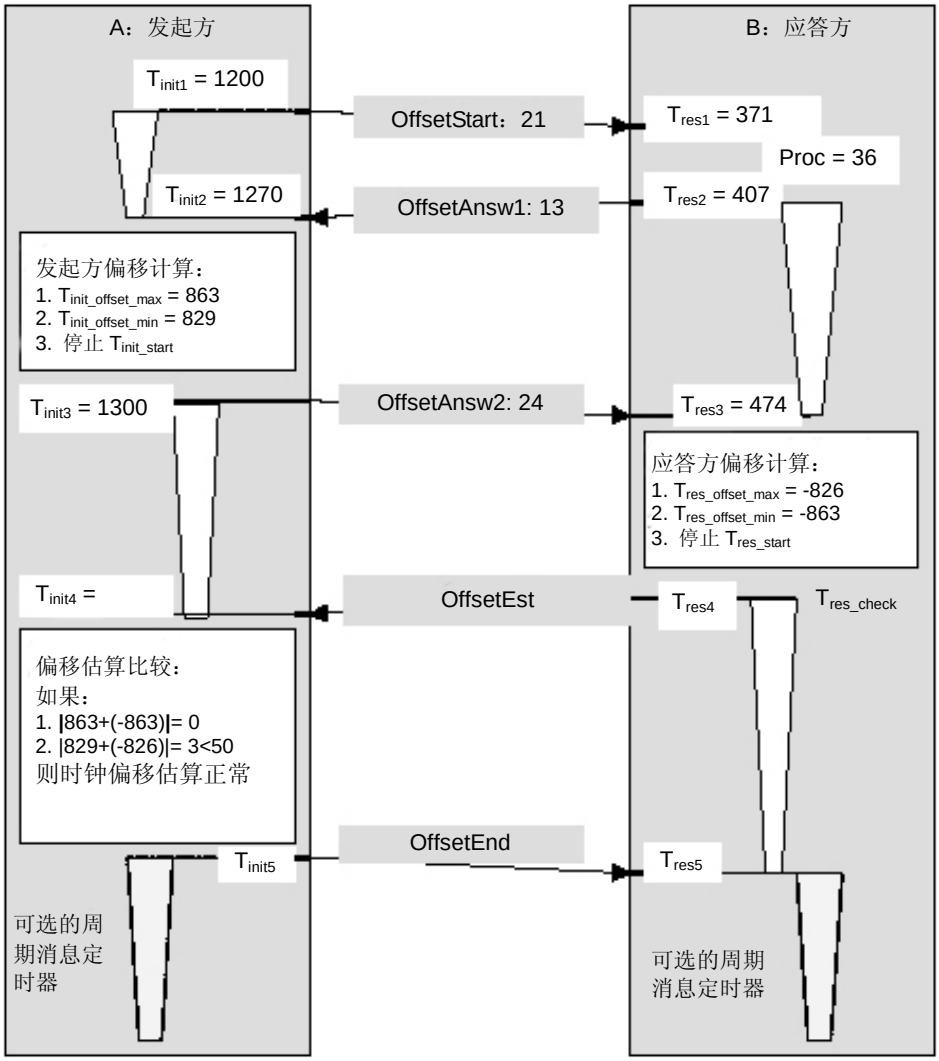


图 26：时钟偏移更新过程举例

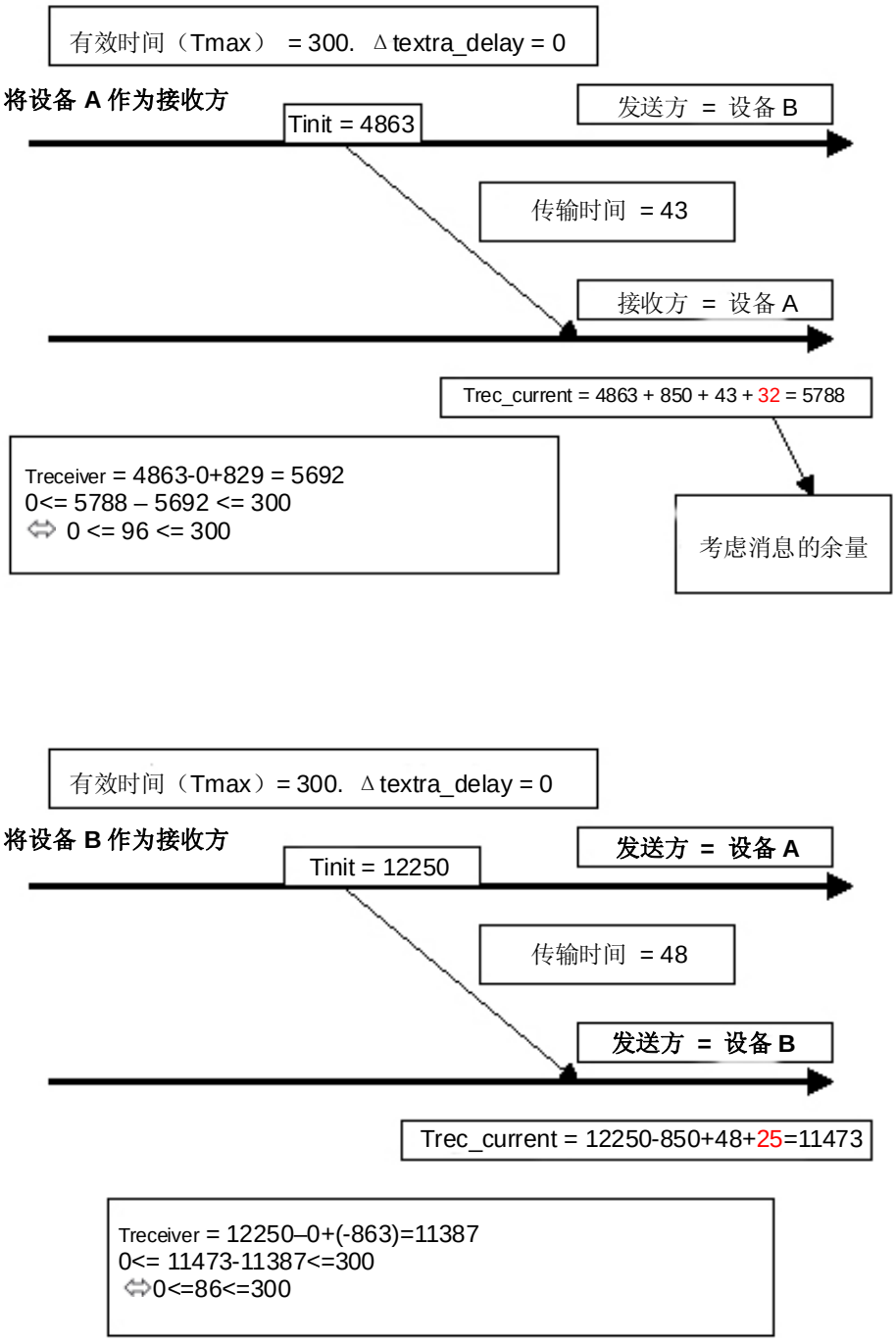


图 27：时间戳程序及检查举例

4. 通信功能模块

4.1. 一般规则

4.1.1.1. 本节描述了安全通信接口中 CFM 使用的通信协议。

4.1.1.2. 本节未定义：

- 具体执行的组件之间交换的信息内容与结构；
- 任何有关接口本身的实施细节；
- 使用的开放网络；
- 通信功能模块的物理架构。

4.1.1.3. 本节仅描述了确保传输层和网络层协议互联有关的功能接口需求，不应被理解为具体的软件结构实现要求。

4.2. 概述

4.2.1. 一般描述

4.2.1.1. 在本 CFM 规范中，两个终端系统均假设为固定的（非移动的）， 并且能够与行业标准的高速网络连接。本规范定义如何使用传输层原语，从而可以获得基于 TCP 的冗余服务。它为铁路安全设备提供了通过国际标准进行通信的方法，且无需了解其它系统的详细特征。

4.2.1.2. CFM 一般特点如下：

- 铁路应用消息的可靠、全双工传输；
- 用户数据的透明传输。信息的内容、格式或编码不受限制。无需为实现传输而了解被传输数据结构和含义；
- 用户数据无限制，也就是说，协议绝不应限制待传输的数据类型；
- 高效的数据传输，CFM 应当支持高性能需求，包括高速、低时延地传输大量数据；
- 对各种已知和当前未知的应用系统开放；
- 支持物理链路冗余性，以增强可用性。

4.3. 功能特性

4.3.1. TCP 与传输 2 类服务和协议的对应关系

4.3.1.1. 本规范旨在适用于各不同供应商提供的地面应用系统。因此，服务的提供采用标准 TCP/IP 作为解决方案，而不是 OSI 或 CENELEC 标准提供的可选协议。因而，本规范采用 TCP，而不是 ISO TP2，来提供一种可靠的面向连接服务。本文将描述所有已知的采用这种途径时带来的限制。

4.3.1.2. 本规范旨在确保能够将灵活的、高效率的、高速而且成本有效的商务现货供应解决方案应用于终端系统，以减低成本和交付周期。

4.3.1.3. 本规范在冗余适配管理层实体（ALE）中定义了传输 2 类服务与 TCP 之间的映射关系，该实体通过离散的、非定长的 ALE 数据包（ALEPKT），实现端到端的数据传输。

4.3.1.4. 下图列出了用于 CTCS 对等应用之间连接的协议栈（本例通过广域网路由链路实现连接）。

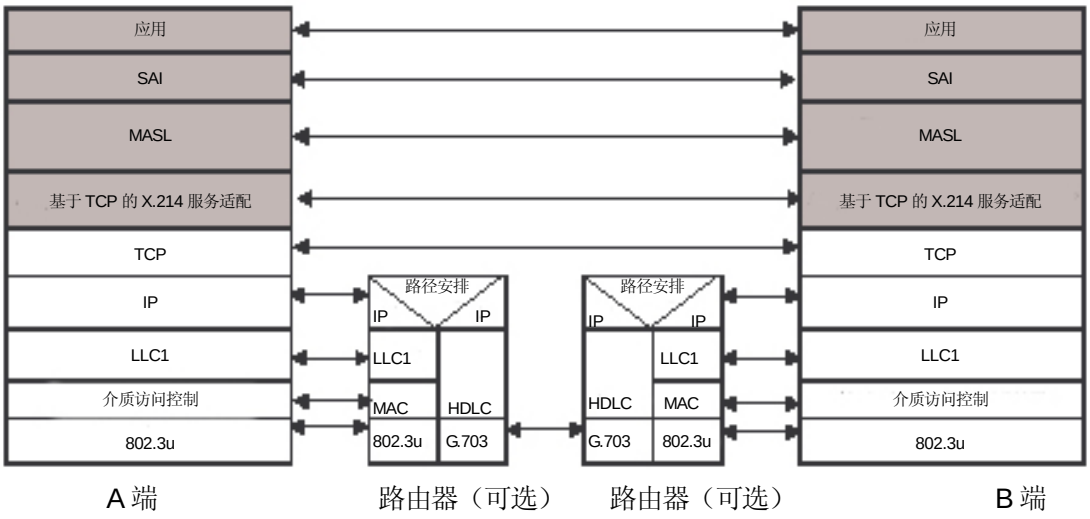


图 28：用于 CTCS 对等应用之间连接的协议栈

注：阴影部分表示需要定制的软件，白色部分则包含可能用于实施的实例。

4.3.1.5. 上图显示了各不同部分之间的关系。负责提供所需功能的应用程序应当使用 Subset-037 中的安全协议原语。此安全协议经由一个与具体环境相关的功能接口访问通信子层，从而实现使用标准原语来访问模拟的 ISO 传输服务。

4.3.1.6. 此类服务实际由 TCP/IP 的传输层和网络层实现，而非 Subset-037 中使用的 ISO TP2 和 T.70 协议。

4.3.1.7. 上图显示了通过广域链路进行系统互连时典型的 IP 路由。

4.3.2. 服务类别

4.3.2.1. 为了实现通用化，本规范通过可选项对要求和提供的服务质量进行分类。现有大量的方法可用于提高本文中定义的服务的可用性，其中之一是可使用独立路由的 TCP 连接来支持同类应用（无论安全与否）之间的单个逻辑连接。

4.3.2.2. 对于不同的服务质量（在 ALE 中称为“服务类别”）说明如下：

4.3.2.3. A 类服务——通常使用两条链路，其中任何一条均可用于数据传输。所有 ALEPKT 均只在一条链路上传输（通常情况下，两条链路均具有相同的性能）；

4.3.2.4. D 类服务——通常使用两条链路，所有 ALEPKT 在两条链路上均被传输。

4.3.2.5. D 类服务的实施是强制性的，而 A 类服务的实施是可选的。

4.3.2.6. 协议支持仅使用一条物理链路进行实施。

4.3.2.7. A 类服务和 D 类服务均可在单物理链路上实现，不会造成任何安全影响。

4.3.2.8. 在连接两端的服务类别应相同。

4.3.3. A 类服务请求

4.3.3.1. 请求 A 类服务时，ALE 实体应与对方建立两条 TCP 链路，其中一条指定的主链路应当在最初即用于数据传输。另一条为次链路，由 ALE 实体开启、维护并监测，但不得用于数据传输，除非主路由发生故障。关于这些链路如何被监测和管理的具体细节参见 6.6 节。

4.3.3.2. ALE层中应定义具体的TCP/IP地址，并根据包含在**T-Connect. Request**原语中的 CTCS-ID 和应用类型进行选择。

4.3.4. D 类服务请求

4.3.4.1. 请求 D 类服务时，ALE 实体应同对方建立两条 TCP 链路，两条链路均用于传输所有数据和控制消息。在一条链路发生故障时，安全连接通过另一条链路继续运行。关于这些链路如何被监测和管理的具体细节参见 6.6 节。

4.3.4.2. 通过单物理链路上的单 TCP 连接请求的 D 类服务也应被接受。

4.3.4.3. ALE层中应定义具体的TCP/IP地址，并根据包含在**T-Connect. Request**原语中的 CTCS-ID 和应用类型进行选择。

4.3.5. TS 用户与 TCP 间的关系

4.3.5.1. 图 29 显示了一个 TS 用户与两个 TCP 通道间的关系。

4.3.5.2. 安全应用可以使用等效的访问原语（位于 ApSAP）访问安全应用中间子层。这些原语和 MASL 层提供（位于 SASAP）的原语相同。

4.3.5.3. MASL 层完成消息防护并通过 TSAP（提供 ISO 传输服务）的服务访问原语向 CFM 以安全协议数据单元（SaPDU）的形式传送消息。

4.3.5.4. ALE 应定义映射功能，从而使用 TCP 协议替代 ISO TP2 协议。

4.3.5.5. 为满足所有的可用性需求，每一个逻辑安全连接应占用一路或多路 TCP 连接。

4.3.5.6. 应使用一个标准的协议栈实现对 LAN 或 WAN（使用 TCP/IP 协议）的接口。

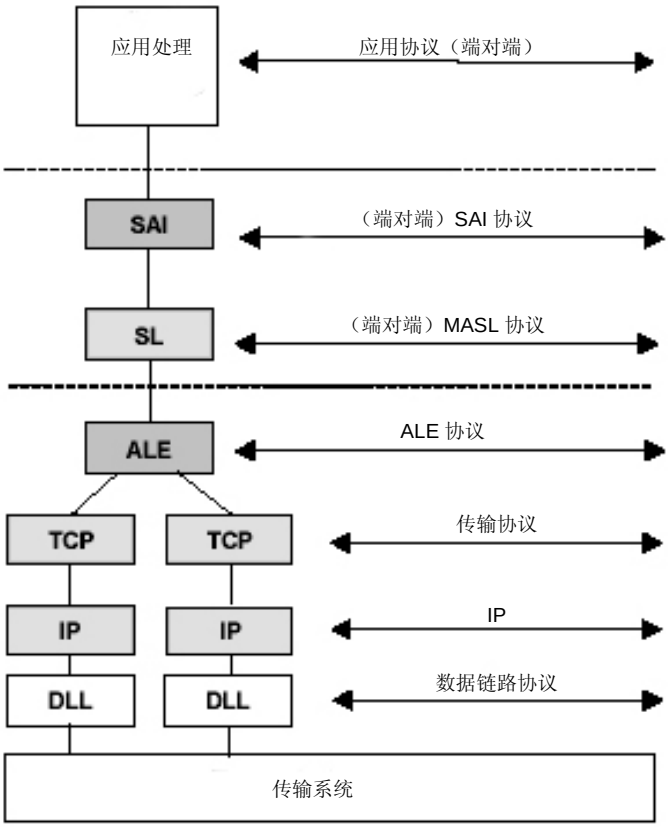


图 29：一个 TS 用户与两个 TCP 通道间的关系

4.3.5.7. 对于该体系结构的每一层，下图中明确了所有服务的用户和提供方，以及每一个协议层的协议数据单元（PDU）结构：

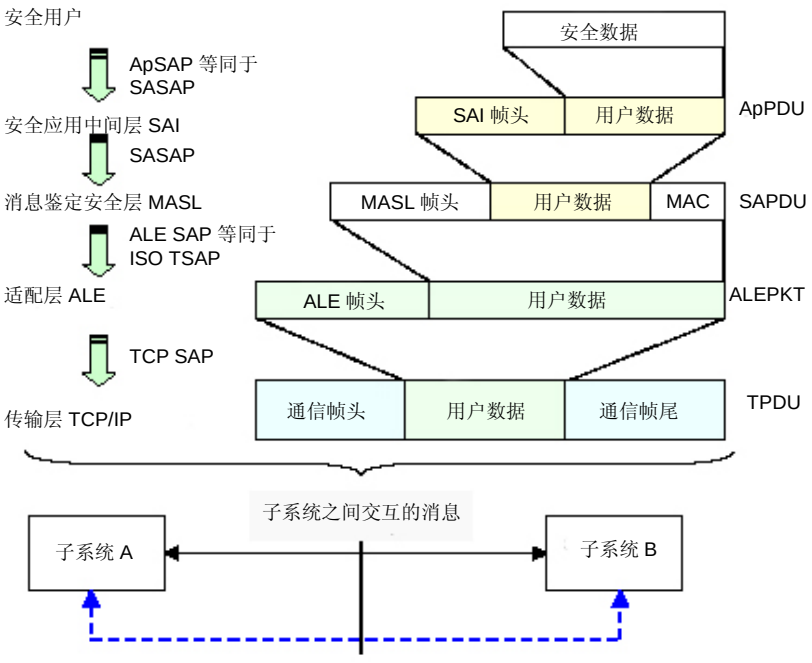


图 30：参考信息结构

4.3.6. 传输优先级

4.3.6.1. 在 Subset-037 中，传输优先级与应用类型有关，具体如下：

- 0 = 不适用；
- 1 = ATP。

4.3.6.2. TCP 不提供也不支持传输优先级。

4.4. 使用适配层实体进行传输层模拟

4.4.1. 概述

4.4.1.1. 本节定义了 CFM 模块提供的传输服务的功能需求。

4.4.1.2. CFM 为 TS 用户提供一个与传输 2 类服务相应的接口，通过使用 TCP/IP 协议实现 ISO 传输功能。

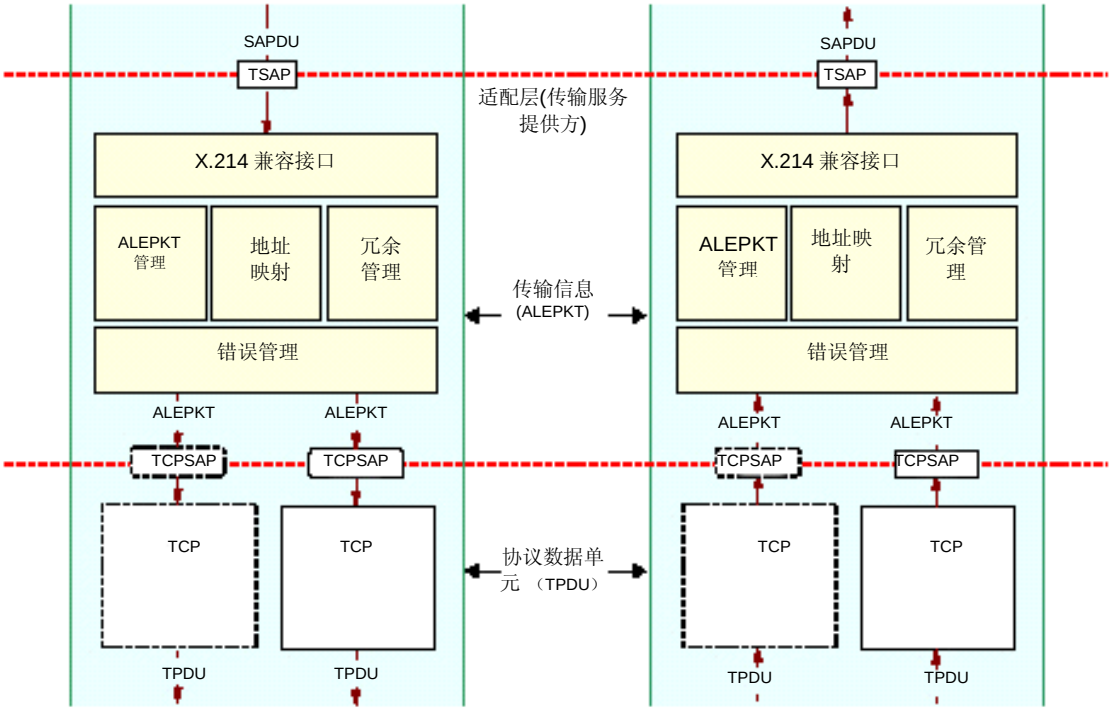


图 31：通信功能模块实例

4.4.1.3. CFM 功能的具体说明如下：

- 在 TSAP（传输 SAP）中定义服务访问原语；
- 通过适配层实现传输 2 类协议（X.214）服务与 TCP/IP 协议原语之间对应关系；
- 定义 ALEPKT 的格式；

- 通过适配层实现 TCP 连接的冗余管理以及相应的功能需求；
- 错误和故障管理。

4.4.2. 接口服务定义

4.4.2.1. ISO 传输服务原语的使用

4.4.2.2. 本节在功能层面上描述了适配层实体向 TS 用户提供的传输服务。

4.4.2.3. 注：在实现终端系统间的互通并确保系统行为不受影响的前提下，如何进行 TS 用户接口的适配属于具体实施的范畴。

4.4.2.4. TS 用户间通过 ALE 传输服务接入点（TSAP）进行信息交互，使用的原语如下表所示：

行为	描述
T-Connet. Request （地址类型， 网络地址， 被叫 CTCS-ID， 主叫 CTCS-ID， 应用类型， 服务质量， 用户数据）	TS 用户（发起方）请求建立一个传输连接。
T-Connet.RESPONSE （TCEPIDA， 应答方 CTCS-ID， 用户数据）	TS 用户（应答方）表示同意连接建立请求。
T-DISCONNECT. REQUEST （TCEPIDXMT， 用户数据）	TS 用户表示传输连接将被关闭。
T-DATA.REQUEST （TCEPIDXMT， 用户数据）	TS 用户发送数据。
事件	**描述**
T-CONNECT. INDICATION （TCEPIDA， 网络地址， 主叫 CTCS-ID， 应用类型， 服务质量， 用户数据）	通知 TS 用户（应答方）传输连接正在建立过程中。
T-CONNECT.CONFIRMATION （TCEPIDA， 应答方 CTCS-ID， 用户数据）	通知 TS 用户（发起方）传输连接已经建立。
T-DISCONNECT. INDICATION （TCEPIDRCV， 原因， 用户数据）	通知 TS 用户传输连接已关闭。
T-DATA. INDICATION （TCEPIDRCV	通知 TS 用户可从传输连接中读取数

用户数据）	据。

表 4：TSAP 服务原语

4.4.2.5. 以上传输服务原语［X.214］的使用应通过下列修正予以支持：

- 移动网络的 ID 参数在此没有意义，应被弃用。
- 无紧急数据，因此无需支持此类服务。
- 传输服务质量参数被重新定义为服务类别来使用。
- 提供非破坏性的传输连接断开服务。

4.4.2.6. X.214 服务中使用的参数值与 FIS［Subset-037］兼容，并具有下列扩展功能。

4.4.2.7. **TCEPID**：（传输连接终点标识符）由本地提供，以区别不同的传输连接。它应有一个到本地 TCP 端口 id 上的映射，通常情况下为 TCP 的 **Local_Connection_Name**。

4.4.2.8. **CTCS 地址类型**：说明 CTCS 地址子参数的类型。

4.4.2.9. **网络地址**：用于识别被叫用户的 32 位目的 IP 地址和 16 位目的 TCP 端口号。

4.4.2.10. 当网络地址由安全层实体提供时，应直接被使用。否则，提供的其它参数应当由适配层实体用来检索相关的 IP 地址和 TCP 端口信息，并由此建立一个至相关远端实体的连接。

4.4.2.11. **被叫 CTCS-ID**：与应用类型一起用于识别被叫传输服务用户。

4.4.2.12. **主叫 CTCS-ID**：与应用类型一起用于识别传输连接发起方。

4.4.2.13. **应用类型**：如 Subset-037 中所述，应用类型字段的前 5 位说明主要应用类型，后 3 位进行进一步的详细说明。

4.4.2.14. **响应 CTCS-ID**：与应用类型一起用于识别接受/响应的传输服务用户，该用户由响应的传输实体进行本地选择。

4.4.2.15. **服务类别**：服务类别不应被协商，请求的服务类别必须被服务提供方和对等应用实体所接受。 否则，连接建立请求将被拒绝。服务类别与应用类型无关。

4.4.2.16. **用户数据**：Subset-037 中定义的 SaPDU。

4.4.2.17. 下表描述了如何使用适配层对原语进行映射。

4.4.3. X.214 原语对 TCP 的映射

4.4.3.1. 注：本节仅作为信息参考。在对互联没有影响的前提下，允许 TCP 使用不用的原语，。

4.4.3.2. 下表概括说明 ISO 传输 2 类原语（行为/事件）如何被适配层映射到 TCP/IP 协议：

行为 X.214	*TCP 原语*	*传送的 ALE 数据包类型*
T-Connect.Request	TCP_OPEN_PORT TCP_SEND_DATA (AU 1 ALEPKT) 或 TCP_ACTIVE_OPEN_WITH_DATA (AU 1 ALEPKT)	1
T-Connect.Response	TCP_SEND_DATA (AU2 ALEPKT)	2
T-Disconnect.Request	TCP_Send_Data (DI ALEPKT) （等待） TCP_Close	4
T-Data.Request	TCP_Send_Data (DT ALEPKT)	3
（无）	TCP_Read_Data	
（无）	TCP_Listen_On_Port	
（无）	TCP_STATUS	
（无）	TCP-Abort	

行为 X.214	*TCP 原语*	*接收的 ALE 数据包类型*
T-Connect.Indication	TCP_Connected TCP_Data_Ready TCP_Read_Data (AU1 ALEPKT)	1
T-Connect.Confirmation	TCP_Connected TCP_Data_Ready TCP_Read_Data (AU2 ALEPKT)	2
T-Disconnect.Indication	（在一个拒绝的 TCP_Open_Port 之后） TCP_Cpnnect_Fails	4
	（在 TCP_Abort 之后） TCP-Errored	
	TCP_Data_Ready TCP_Read_Data (DI ALEPKT) TCP-Close TCP-Closed	
T-Data.Indication	TCP-Data-Ready	3

表 5：X.214 原语对 TCP 的映射

4.4.4. 寻址

4.4.4.1. 地址结构

4.4.4.2. Subset-037 通过原语中提供的 CTCS-ID 地址信息在 **T-Connect.Request** TPDU（CR TPDU）和 **T-Connect.Confirm** TPDU（CC TPDU）中进行传输选择。在上述原语参数中可提供完整的地址信息。

4.4.4.3. 当用于 TCP/IP 连接时，提供的信息应当由适配层实体用来检索相关端口和 IP 地址信息，由此可建立一个至相关远端实体的连接。

4.4.4.4. 应用类型字段的前 5 位说明了主要的应用类型。后 3 位进行进一步的详细说明。6.5.2 节描述了如何传输到远端系统。对于特定的连接，主叫和被叫系统的应用类型应当相同。发送方应使用适当的应用类型，而接收方可加以忽略。

4.4.4.5. 为了与基于 ISO 传输服务的车地无线安全服务保持共性，适配层使用的应用类型参数时格式应当保持不变，其格式和适用数值在 Subset-037 的 8.2.4.6.4 节，表 37 中进行了说明。

4.4.5. 适配层数据包格式

4.4.5.1. TCP 和 ISO 传输服务之间的功能区别之一就是，TCP 管理一个连续的字节流而没有明显的边界，而 TP2 处理的则是具有良好边界的 TPDU。

4.4.5.2. 为了适应这种区别，适配层使用一种简单的打包方案来划定 TS 用户交换的信息。

4.4.5.3. 接收方 ALE 能够对已经被嵌入到连续字节流（在 TCP 层）的每一个信息包进行重新组织。

4.4.5.4. ALE 层处理的信息包称为 ALE 信息包（ALEPKT），如下文所述，ALEPKT 为长度可变的、由整数字节构成的对象 。除非特别说明，所有字段均按 Big Endian 顺序排列。ALEPKT 包括两个部分：

- 包头；
- TS 用户数据 PDU（用户数据）。

4.4.5.5. 帧头格式是恒定不变的：

包长度	版本	应用类型	传输序列号	N/R 标志	包类型	校验和	用户数据
2 字节	1 字节	1 字节	2 字节	1 字节	1 字节	2 字节	长度可变

其中：

字段名称	*长度*	*描述*
包长度	2 字节	去除本字段后，整个 ALEPKT 的长度（以字节为单位）。
版本	1 字节	用于识别适配层为其它方提供的服务程序。它作为配置的一部分须在对等实体之间达成一致。ALEPKT 中的版本参数可以被接收的 ALE 忽略。
应用类型	1 字节	根据 6.4.4.1 节的规定，用于识别应用类型。ALEPKT 中的应用类型参数可以被接收 ALE 忽略。
传输序列号	2 字节	传输序列号用于接收方适配层管理两个活动的 TCP 连接的切换。传输序列号字段不用于 A 类服务。在 A 类服务中，传输序列号的数值被设为“0”。
N/R 标志	1 字节	用于发送方 ALE 说明发送 ALEPKT 的正常或冗余链路。在配置阶段，链路的“正常”或“冗余”属性是固定的。链路应当进行正常-正常、冗余-冗余连接。N=1；R=0。
包类型	1 字节	包类型描述（用于接收方适配层采取合适的解析方式）。如果此项不属于选定的服务类别中指定的包类型的一种，接收方 ALE 应丢弃该包，此时也可以关闭该 TCP 连接。
校验和	2 字节	按照 FCS-16 计算的 CRC（针对前 6 个字段的 8 个字节）用于确定是否存在数据传输错误。生成多项式为 CRC-CCITT（$x^{16}+x^{12}+x^5+1$）。循环冗余校验（CRC）算法在参见 ISO/IEC3309。 CRC 的最终结果在发送或者接收实体中无须取反。 计算得到的 CRC 校验和的最高项对应该字段的最低有效位。CRC 结果的实例详见资料性附录 7.5 节表 14。 CRC 校验失败将导致 ALE 断开（重连）TCP 连接。 A 类服务中的生命信息不需要进行 CRC 检验。
用户数据	变量	长度合法的用户数据。

表 6：ALEPKT 信息结构

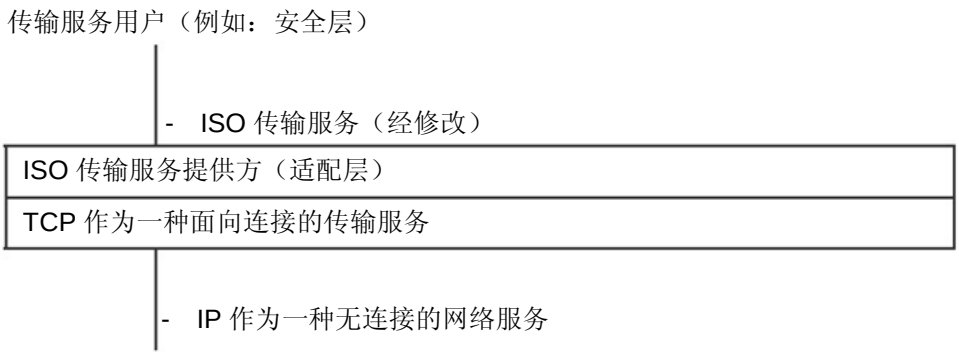
4.5. 接口协议定义

4.5.1. 运用 TCP/IP 提供 ISO 传输 2 类协议

4.5.1.1. 本规范中提供的模型规定了一种简单的封装机制，允许使用 TCP 来传输 SaPDU。这种封装的 PDU 被称为 ALE 包（ALEPKT）。

4.5.1.2. 对 X.214 传输服务进行了微小的相关改动，并由此规定了一种与其类似的基于 TCP 的服务。为避免混淆，CFM 不使用 X.214 和 X.224。它们仅用于说明有待提供的功能。

4.5.1.3. TCP 用于代替 X.224，以提供一种类似于 CONS 的服务。这些标准之间的关系图示如下：



4.5.1.4. 基于 TCP 的仿传输 2 类服务

4.5.1.4.1. ISO 传输 2 类提供所需的功能，用于通过协商建立连接，分段传输数据以及报告协议错误。它提供带有流控制的传输连接，该流控制功能基于 NS 提供方。它提供显式断开传输连接的功能。

4.5.1.4.2. 下表规定了在使用 TCP 时各种 ISO TP2 功能是否与之相关并被提供。与服务操作有关的缺失功能则将在后文进行详细讨论。

4.5.1.5. TP2 一般规程要素的映射

协议机制	X.224 的 X-ref 中的章节	变量或选项	TP2 类	基于 TCP 使用	不基于 TCP 使用	备注
网络连接的分配	6.1.1		X	*		适配层应当包含一个表，允许将本地安全实体发起的呼叫

						映射到一个 TCP 端口+IP 地址的组合。如果 TCP 连接还没有建立，则应当创建该连接。
TPDU 传输	6.2		X	*		安全层提交的数据应当通过标准数据传输特性进行传输。
分段与重组	6.3		X		*	长消息应当通过多个 MTU 进行传输。 由于 TCP 上传输的是字节流，而不是分开的数据单元，因此，有必要对 SaPDU 进行识别和分界
串联和分离	6.4		X		*	本功能与 TCP 无关。
建立连接	6.5		X	*		使用一般 TCP 协议
拒绝连接	6.6		X	*		使用一般 TCP 协议
正常释放	6.7	显式	X	*		使用一般 TCP 协议
错误释放	6.8		X	*		使用一般 TCP 协议
TPDU 与传输连接的关联	6.9		X	*		TCP 端口地址应被用于区分各连接。适配层应当为各连接分配端口号（TP2 中一般使用 SRC-ref 和 DST-ref）。
TPDU 编号	6.10	一般 扩展			* *	当使用 TCP 时，通过其他功能来恢复数据、控制流量并进行重排序。
紧急数据传输	6.11	网络加急			*	在此不支持。适配层应当拒绝 HP-DATA 请求原语
失败之后重新分配	6.12		N/A		*	不用于 TP2 中。连接应当由适配层进行管理。
TPDU 的保持和确认	6.13	接收确认	N/A		*	不适用于 TP2。当使用 TCP 时不相关

协议机制	X.224 的	变型或	TP 类	基于 TCP	不基于	备注

	X-ref 段	选项	别 2	使用	TCP 使用	
再同步	6.14		N/A		*	不用于 TP2。
多路复用和解复用	6.15		X		*	通过每个逻辑连接单独使用一个 TCP 端口来实现。
显式流量控制	10.2		X		*	使用一般 TCP 控制。
校验和	6.17		X		*	在使用 TCP 时不相关。TCP 本身的算法可保护数据。
固定参考	6.18				*	在使用 TCP 时不相关。
超时重传	6.19		N/A		*	一般不被 TP2 使用。 通过 TCP 超时和超时处理参数进行处理。
重排序	6.20		N/A		*	不在 TP2 中使用。 TCP 进行处理。
休眠控制	6.21		N/A		*	对于检测无信号时网络的丢失非常重要（由远端实体的中断或丢失导致）。通过 TCP 超时和超时参数来进行处理。
协议错误处理	6.22		X	*		TCP 使用 STATUS 和 ERROR 原语
分割与重组	6.23		N/A		*	TP2 不使用。 在使用 TCP 时不相关。

表 7：TP2 一般规程要素的映射

注：

X　　表示此规程包含在 TP 2 类中。

N/A　　不适用于 TP 2 类。

4.5.1.5.1. TCP 协议服务接入点（TCP SAP）向服务用户（TCP-User）提供一系列特定的服务访问原语。规范 RFC0793 描述了一种这些原语可能的实现方式，它们在下表中进行了重新说明，仅供参考。在互联互通的前提下，也可以采用其他的实现方式（例如，通过标准的“socket”接口）。

行为	描述
TCP_Listen-On-Port	TCP-用户要求以被动方式打开一个指定端口。
TCP-Open-Port	TCP-用户要求以主动方式打开一个指定端口。

TCP-Read-Data	TCP-用户从连接中读取数据。
TCP-Send-Data	TCP-用户向连接中发送数据。
TCP-Close	TCP-用户关闭连接（待处理数据被发送）。
TCP-Abort	TCP-用户关闭连接（待处理数据丢失）。
TCP-Status	TCP-用户查询连接状态相关信息。
事件	**描述**
TCP-Connected	成功建立连接（主动或者被动）。
TCP-Connect-Fails	主动打开失败。
TCP-Data-Ready	数据已准备好，可以从连接中读取。
TCP-Errored	连接发生错误，并且现在已被关闭。
TCP-Closed	已经完成了一次正常的断开连接。

表 8：TCP 中的服务原语

4.5.2. ALE 操作

4.5.2.1. 创建一个连接

4.5.2.1.1. 为创建 TS 用户实体之间的连接，必须在这些实体之间按照特定的顺序交换消息。这可以使用 TCP 服务，通过下列方法在适配层之间启动一个“传输”连接来实现：

- 可以根据应用类型按照优先次序建立连接。
- 如果需要，应当通过适配层来进行地址解析。
- 每一个连接请求应当通过新的独立 TCP 连接来建立。

4.5.2.1.2. 使用标准程序来建立 TCP 连接，以便处理如果对等实体同时向对方请求连接时所产生的问题。适配层及其 TCP 连接的参数描述如下。

4.5.2.2. 地址映射

4.5.2.2.1. 地址映射属于本地实现的问题。在资料性附录 7.2 中举例提供了一种可能的实现方式。

4.5.2.2.2. 寻址详情必须得到对等实体之间的一致认可。

4.5.2.3. 建立连接

4.5.2.3.1. 两个 TS 用户实体之间的连接可以使用原语 T-Connect.Request 来实现。此时使用 TCP 服务的双方适配层实体之间将建立一个或多个传输连接。

4.5.2.3.2. TS 用户实体之间建立的每个连接均可根据下列内容进行标识：

- 连接双方的 CTCS-ID
  主叫 CTCS-ID
  被叫 CTCS-ID（响应 CTCS-ID）
- 应用类型。

4.5.2.3.3. 除非网络地址已在 T-Connect.Request 中确定，适配层应当从本地配置数据中获取地址信息（IP 地址 +端口）。

4.5.2.4. 详细协议顺序

4.5.2.4.1. 在连接建立阶段适配层实体之间的 ALEPKT 交互说明如下：

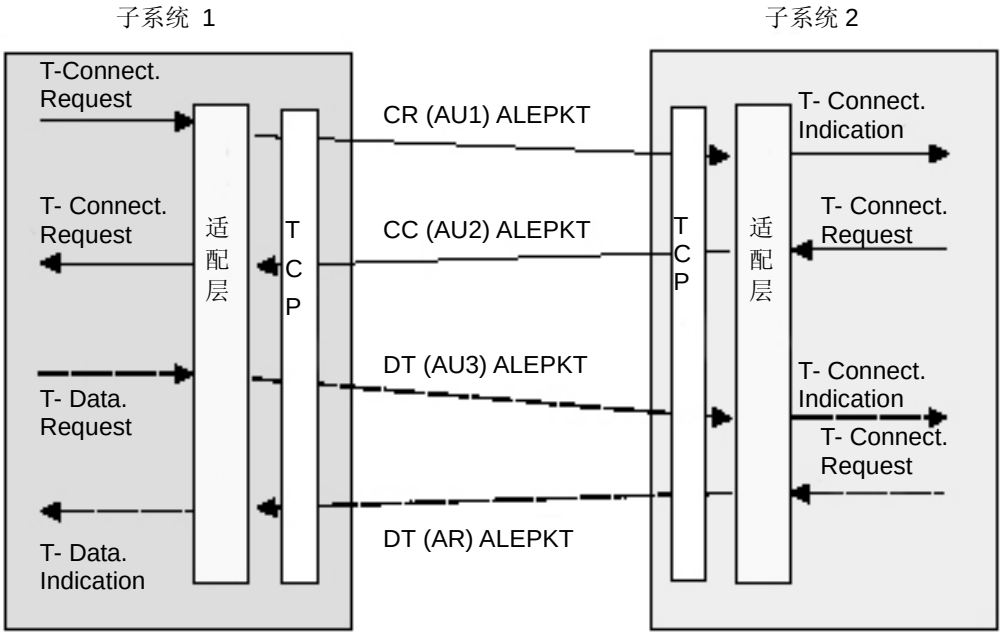


图 32：安全连接建立详细协议顺序

4.5.2.4.2. AU1 ALEPKT（CR）格式如下：

AU1 ALEPKT — 发起方至应答方（包类型 1）				
ALEPKT 帧头	适配层连接消息			ALEPKT 用户数据
帧头字段	主叫CTCS-ID[10]	被叫 CTCS-ID[1]	服务类别	AU1 SaPDU
10 字节	*4 字节*	*4 字节*	*1 字节*	

[10] 包括两个字段：CTCS-ID 和 CTCS-ID 类型，如 Subset-037 所述。

4.5.2.4.3. 发起方 TS 用户应当说明自身（主叫 CTCS-ID）和应答方（被叫 CTCS-ID）的唯一标志符，以及建立连接的应用类型。

4.5.2.4.4. 适配层不处理用户数据，也不对其进行管理。如果存在用户数据，应直接发送给相应的 TS 用户。

4.5.2.4.5. AU1 ALEPKT 应按照 Subset-037 中指定的格式在用户数据 AU1 SaPDU 中包含验证信息。

4.5.2.4.6. 应答方应当能够通过 T-Connect.Response 或 T-Disconnect.Request 拒绝或者接受连接请求，并将其唯一标识符（响应 CTCS-ID）发送给发起方。

4.5.2.4.7. 如果连接被接受，则发送 AU2 ALEPKT（CC）被发送，格式如下：

AU 2 ALEPKT - 发起方至应答方（包类型 2）		
ALEPKT 帧头	适配层连接消息	ALEPKT 用户数据
帧头字段	应答 CTCS-ID[1]	AU2 SaPDU
10 字节	*4 字节*	

4.5.2.4.8. 完成以上消息的交互之后，传输连接即已建立。

4.5.2.4.9. TS 用户进一步进行 ALEPKT 交换，以实现相关安全连接的建立:

DT ALEPKT - 发起方至应答方（包类型 3）	
ALEPKT 帧头	ALEPKT 用户数据
帧头字段	AU3 SaPDU
10 字节	

DT ALEPKT - 发起方至应答方（4 类包）	
ALEPKT 帧头	ALEPKT 用户数据
帧头字段	AR SaPDU
10 字节	

4.5.2.4.10. 从适配层的角度来看，后一个交互是普通的用户数据传输，而不属于传输连接建立的范围。

4.5.2.5. 基于 TCP/IP 的映射

4.5.2.5.1. 注：本节所述内容仅为参考信息。在不影响互联互通的前提下，允许使用其他的原语。

4.5.2.5.2. 6.5.2.5.2 OSI 连接模式要求传输提供方获得传输用户的明确许可，以便建立连接。因此，对于 TS 用户实体之间的每一个连接，作为发起方的主叫 CTCS-ID 应当发送 T-Connect.Request 做出请求，作为应答方的被叫 CTCS-ID 应当等待 T-Connect.Indication 发生，并通过 T-Connect.Request 来进行明确的回复。

4.5.2.5.3. TCP 连接模式与 ISO 模式不同。在 TCP 模式中，响应传输用户是被动的，并且不会明显干涉 TCP 的连接过程。

4.5.2.5.4. 这意味着，在收到 T-Connect.Request 之前，应答方适配层必须通过一个 IP 地址加端口上的 TCP_LISTEN_ON_PORT 监听连接请求。

4.5.2.5.5. 一般情况下，接收的呼叫被分配到对应的套接字上，同时监听行为可以继续保持，以便接受其它呼叫请求。

4.5.2.5.6. TCP 也允许在两个同时发布连接请求的传输用户之间建立连接。ISO 则不是，OSI 模式中的同步对称连接请求将导致产生两个传输连接。

4.5.2.5.7. 一般而言，这种情况仅具有理论意义，因为发起方和应答方的主动与被动特性可以避免同步对称连接。

4.5.2.5.8. 对于一次成功的连接，ISO TP2 与 TCP 之间行为/事件的相关顺序如下：

ISO 发起方	*TCP 客户端*	数据	TCP 服务器端	ISO 应答方
			TCP_Listen_On_Port	
T_Connect. Request				
	TCP_Open_Port			
			TCP_Connected （接受）	
	TCP_Connected			

	TCP_Send_Data	CR AU1 ALEPKT		
			TCP_Data_Ready	
			TCP_Read_Data	
			（ALEPKT 校验）	
				T_Connect. Indication
				T_Connect. Response
		CC AU2 ALEPKT	TCP_Send_Data	
	TCP_Data_Ready			
	TCP-Read_Data			
	（ALEPKT 校验）			
T_Connect. Confirm				

4.5.2.5.9. 对于一次不成功的连接:

ISO 发起方	*TCP 客户端*	数据	TCP 服务器端	ISO 应答方
			TCP_Listen _On-Port	
T_Connect. Request				
	TCP_OPEN_PORT			
			TCP_Close （拒绝）	
	TCP_CONNECT-FAILS			
T-Disconnect.Indication				

表 9：建立连接时 TP2 行为/事件与 TCP 之间的映射

4.5.2.6. 创建连接

4.5.2.6.1. 两个子系统间的每个连接均通过一个或两个传输连接来实现。若要求两个传输连接，在需要保证高可用性时，应当通过分散的路由实现，此项属于本地实施的范畴。

4.5.2.6.2. 对于每一个连接请求，无论是主动（T-Connect.Request）或是被动（T-Connect.Indication），适配层均须为发送的 ALEPKT 分配一个顺序计数的传输序列号。

4.5.2.6.3. 每一个传输序列号均使用 2 字节的 Big Endian 编码（16 位无符号整型），取值范围在 0 到 65535 之间。

4.5.2.6.4. ALEPKT 传输序列号应在开始连接之前初始化为“0”。因此，AU1 和 AU2 ALEPKKT 的传输序列号应当始终等于 0。对于 D 类服务，每个传输链路上的 ALEPKT 序列号的计数均每次递增 1。

4.5.2.6.5. 如果 T-Connect.Request 失败，发起方应重新尝试建立连接。

4.5.3. 数据传输

4.5.3.1. 简介

4.5.3.1.1. 数据传输通过原语 T-Data.Request 和 T-Data.Indication 使用以下描述的 ALEPKT 格式实现。

4.5.3.1.2. 在包类型为 3 的 ALEPKT 的用户数据域，TS 用户可以填写任何格式的用户数据。

4.5.3.1.3. PDU 的数据量存在限制，为了将来能够插入更详细的帧头信息，允许传输的用户数据的最大字节数不得超过 65000 字节。

4.5.3.2. 详细协议顺序

4.5.3.2.1. 在数据传输阶段，两个适配层实体之间 ALEPKT 包的交互如下图所示：

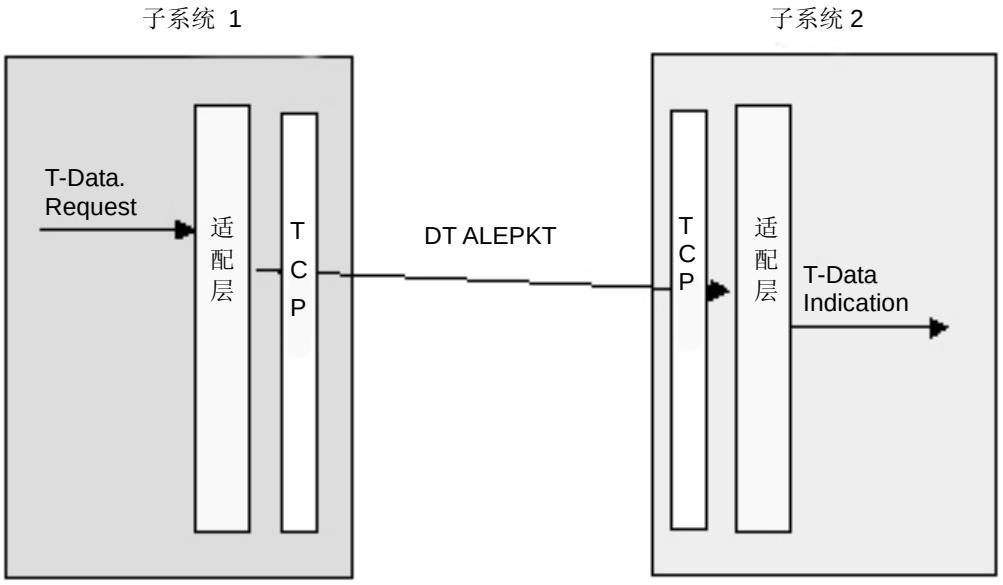


图 33：详细数据传输协议顺序

4.5.3.2.2. DT ALEPKT（数据传输）具有下列格式：

DT ALEPKT - 发送方至接收方（包类型 3）	
ALEPKT 帧头	ALEPKT 用户数据
帧头字段	DT SaPDU
10 字节	

4.5.3.3. 基于 TCP 的映射

4.5.3.3.1. 注：本节所述内容仅为参考信息。在不影响互联互通的前提下，允许使用其他的原语。

4.5.3.3.2. TCP 数据传输模式与 ISO 模式不同，其事件不包含接收到的数据（T-Data.Indication），而仅仅发送数据可用性信息（TCP_DATA_READY），数据读取由该行为的第二步负责执行（TCP_READ_DATA）。

4.5.3.3.3. ISO TP2 与 TCP 之间行为/事件的相关顺序如下：

XmtISO	*Xmt TCP*	数据	Rcv TCP	Rcv ISO
T-DATA.REQUEST				
	TCP_SEND_DATA	DT ALEPKT		
			TCP_DATA-READY	
			TCP_READ_DATA	
			（ALEPKT 校验）	
				T-Data.Indicati

				on

4.5.3.3.4. DT ALEPKT 包通过 T-Data.Request 和 T-Data.Indication 在 TS 用户实体之间的流动应当根据 TS 用户请求的服务类别来实施。

4.5.3.3.5. T-Data.Request：

- ALEPKT 帧头中包含的传输序列号在每次发送时以 1 为单位递增。
- 传输序列号达到最大值（65535）后应当归 0。

4.5.4. 连接释放

4.5.4.1. 简介

4.5.4.1.1. 释放 ISO 连接需要使用两条原语:T-Disconnect.Request 以及相应的 T-Disconnect.Indication。

4.5.4.1.2. ISO 连接释放服务 T-Disconnect.Request 用于：

- 取消连接建立；
- 释放一个已建立的连接；
- 表示建立连接失败；
- 表示保持连接失败。

4.5.4.1.3. 连接释放允许在任何时候进行，并且释放请求不应被适配层拒绝。

4.5.4.1.4. 连接可以使用两种方式断开：

4.5.4.2. 正常断开连接

4.5.4.2.1. “正常断开连接”要求所有 TPDU 队列（指此前由 TS 用户发送给本地 TS 提供者的 TPDU）应当在 TCP 连接关闭之前发送给远端 TS 用户。

4.5.4.2.2. 为实现这一目标，适配层（TS 提供者）应在输出缓冲区中排列 DIALEPKT 并传输，然后释放连接。数据传输一旦完成，远端适配层则关闭连接，并向本地用户发送一个原因代码为“正常”的 T-Disconnect.Indication。

4.5.4.3. 异常断开连接

4.5.4.3.1. 通信系统故障造成中断时将导致“异常断开连接”。这种断开连接的方式不保证此前的数据传送。

4.5.4.3.2. 在这种情况下，双方 TS 提供者（适配层）应向各自的 TS 用户发送一个包含“原因”和“子原因”代码的 T-Disconnect.Indication。

4.5.4.3.3. TS 用户通过 T-Disconnect.Request 要求释放连接时，适配层必须采用正常断开连接的方式。

4.5.4.4. 详细协议顺序

4.5.4.4.1. 如果根据 TS 用户的要求予以中断，则两个适配层之间应通过以下方式释放连接：

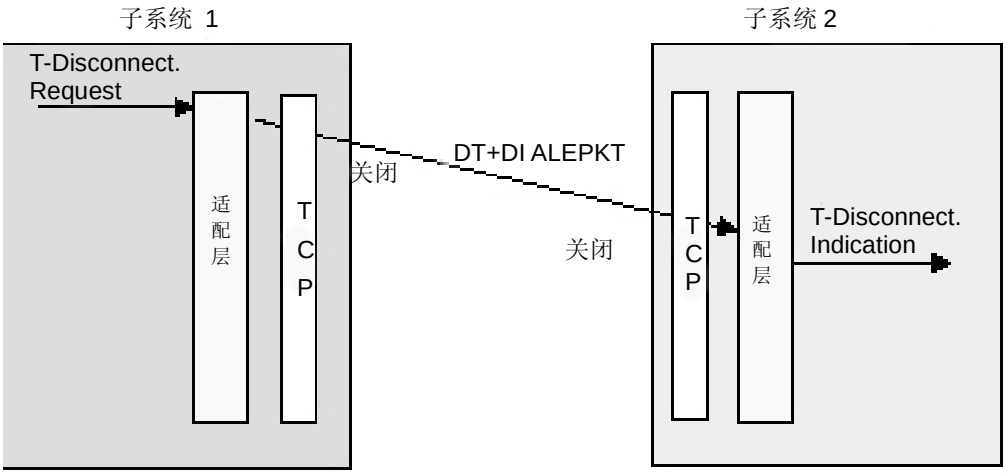


图 34：正常连接释放的详细协议顺序

4.5.4.4.2. DI ALEPKT（断开指示）格式如下：

DI ALEPKT-发起方至应答方（包类型 4）	
ALEPKT 帧头	ALEPKT 用户数据
帧头字段	
10 字节	DI SaPDU

4.5.4.4.3. 通信系统出现故障时（异常断开连接），流程如下：

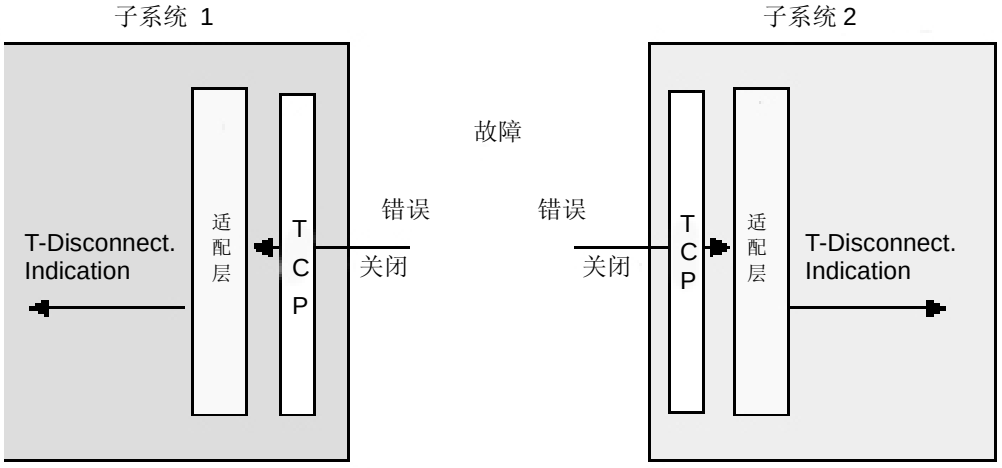


图 35：异常连接释放的详细协议顺序

4.5.4.5. TCP 映射

4.5.4.5.1. 注：本节所述内容仅为参考信息。在不影响互联互通的前提下，允许使用其他的原语。

4.5.4.5.2. ISO 连接释放服务类似于 TCP 连接中断服务（TCP_CLOSE）。不过，与 TCP 不同的是，ISO 不支持有序的连接释放。一旦进入释放阶段，传输提供方不保证任何用户数据的传送。而 TCP 的 释放形式可以保证缓冲区内所有数据在连接被释放之前被发送到另一方的 TS 用户。ISO 仅保证缓冲数据被发送给目的地的 TS 提供者。

4.5.4.5.3. TCP 有序释放服务要求每一个传输用户在连接断开之前必须同意释放该连接，在此之前，TCP 传输提供方应保持连接。一旦连接被传输用户释放，则不再发送数据，但用户可以继续接收数据。此前传送的任何数据均可发送给目的地传输用户。

4.5.4.5.4. 适配层提供的 ISO 连接释放服务应当使所有（一个或多个）相关的 TCP 连接被释放。

4.5.4.5.5. ISO TP2 与 TCP 之间的行为/事件相关序列如下：

ISO	*TCP*	数据	TCP	ISO
T-Disconnect. Request				
	TCP_SEND_ DATA	DI ALEPKT		
	（等待）		TCP_Data_Ready	
			TCP_Read_Data	
	TCP_Close		（ALEPKT 校验）	
			TCP_Close	
	TCP_Closed		TCP_Closed	T_Disconnect. Indication

4.5.4.5.6. 远端 TCP 实体向 ALE 层实体报告连接已经通过一个 TCP_CLOSE 事件被关闭：如果 TCP 连接失败或者被拒绝，则 TCP 向 ALE 层实体报告连接已经通过 TCP_CONNECT_FAILS 事件被关闭。

4.5.4.5.7. 如果适配层认为一条链路因为其它的原因不可用，则应当使用原语 TCP_ABORT 来删除这一链路，并将 TCP_ERRORED 事件发送到连接的另一端。

4.5.4.6. TS 用户请求正常断开

4.5.4.6.1. 当一个 TS 用户希望断开其与远端 TS 用户的关连时，可以使用 T-Disconnect.Request。一旦收到这一原语，本地适配层应当通过 TCP_CLOSE 关闭所有在连接请求时被打开的 TCP 连接。

4.5.4.6.2. 一旦所有数据均被传输（包括连接断开请求），则另一方向的 TCP 连接应当关闭。

4.5.4.6.3. 注：为了触发远端实体的对等 T-Disconnect.Indication 事件，所有与该安全连接相关的 TCP 连接必须关闭。远端 TCP 实体向本地适配层实体报告连接已经通过 TCP_CLOSE 事件关闭。当正在关闭的 TCP 链路为最后一个用于特定 ISO 连接的链路时，适配层应向本地的 TS 用户发送一个具备相关原因代码的 T-Disconnect.Indication。

4.5.4.7. 适配层异常释放

4.5.4.7.1. 在本地适配层实体出现异常而断开连接时，所有 TCP 相关的链路也应当运用 TCP_CLOSE 来释放。一旦执行最后一个 TCP_CLOSED，就应当生成包含有相应的“原因”和“子原因”参数的 T-Disconnect.Indication 事件通知 TS 用户。

4.5.4.8. TCP 异常释放

4.5.4.8.1. TCP 链路异常释放的情况更为复杂。

4.5.4.8.2. 如果最后一个 TCP 链接被释放，则应当发送带有相应错误的 T-Disconnect.Indication 事件给 TS 用户。

4.5.4.8.3. 如果两条链路之中任何一条 TCP 通道因为某些原因被释放，而另一条仍然有效使用，则发起方的适配层应当尝试恢复与远端适配层的连接而无须通知 TS 用户。

4.5.4.8.4. 如果重连失败，则适配层必须尝试再次连接，连接次数超过最大次数限制后，必须通过带有相应错误代码的 T-Report.Indication 将问题报告给 TS 用户或者管理实体。

4.6. 不同服务类别的实施和冗余管理

4.6.1. A 类服务

4.6.1.1. 定义

4.6.1.1.1. 如果在 ALE 实体之间使用两条链路，则应该固定配置一条为正常链路，另一条为冗余链路。

4.6.1.1.2. 在 ALE 实体之间使用两条链路时，如果其中一条链路是用于建立/释放 ALE 连接并交换含有应用数据 ALEPKT 的，则该链路被称为“活动链路”。另一条链路被称为“非活动链路”。当在两种链路之间进行切换时，则活动链路变成非活动链路，反之亦然。

4.6.1.1.3. “正常/冗余”和“活动/非活动”为两种不同的概念。正常或冗余链路均可作为活动或非活动链路。正常或冗余在物理上是固定的，而活动或非活动则为动态分配的。

4.6.1.1.4. A 类服务代码为 0x00。

4.6.1.2. 建立 ALE 连接

4.6.1.2.1. 如果正常链路可用，则应最先使用这一链路来建立 ALE 连接并在对等实体之间传输所有用户数据 ALEPKT。这一通道应当将 ALEPKT 帧头的 N/R 标识设置为数值 1。

4.6.1.2.2. 如果条件允许，两个对等实体间的第二条 TCP 连接也应尽快的在不同的物理链路上发起连接。如有必要，这一连接应当被用作冗余链路。在这一连接发送的任何消息的 N/R 标志都应设为 0。

4.6.1.2.3. 如果正常链路不可用，则应使用冗余链路来建立 ALE 连接，并在对等实体之间传输所有用户数据。在这一连接上发送的任何消息的 N/R 标识均应设为 0。

4.6.1.2.4. 发起方选择的链路将用于交换 1 类包和 2 类包—这条链路将作为发起方的活动链路。1 类包和 2 类包不应当在非活动链路上交换。当应答方从一条链路上接收到 1 类包时，该条链路就作为应答方的活动链路。2 类包应当在活动链路上传送。发起方应当核对 2 类包是否在活动链路上接收。如果核对失败，ALE 层应释放连接。

4.6.1.2.5. 在 ALE 连接建立（1 类包和 2 类包交换）之前，不允许使用 3 类、251 类、253 类、254 类和 255 类包。

4.6.1.2.6. A 类服务应能够在一条物理链路和一个 TCP 连接上无冗余运行。在这种情况下，N/R 标识设为 1。

4.6.1.3. 数据传输

4.6.1.3.1. 如 4 .5.3 所述，ALEPKT 应当在活动链路上传输。当在活动链路上没有可供传输的数据包时，并且为这一连接所配置的“生命保持”定时器已耗尽时，则包类型设为“255”（KAA = 活动链路的生命信息）的 ALEPKT 帧头应当在这一链路上传送。

4.6.1.3.2. 为了检查非活动链路的可用性，不管为这一连接配置的定时器何时耗尽，“生命保持”包也应当在这一链路上双向传送。包类型应当设置为“254”（KANA =非活动链路的生命信息）。没有用户数据。

4.6.1.3.3. 如果非活动链路可用，则在非活动链路上的“非活动链路的生命信息”包（254 类包）应当在建立 ALE 层连接之后在两个对等实体之间交换。

4.6.1.3.4. 当接收到消息时：

活动链路上的 254 类包（非活动链路的生命信息）、非活动链路上的 255 类包（活动链路的生命信息、非活动链路上的 3 类包（数据消息）。则应向对方发送以下交互包：

- 251 类包（表示数据将切换到冗余链路上传送）。
- 或 253 类包（表示数据切换到正常链路上传送）。

4.6.1.4. 冗余管理

4.6.1.4.1. 当活动链路失效时，数据传输应当切换到非活动链路。

4.6.1.4.2. 如果活动链路为正常链路，应通过对等实体之一在冗余链路上发送一个包类型为 251 的 SwitchN2R ALEPKT 来实现切换，用户数据可以包含在该包中正常发送。这使得先检测到错误的一方可以立即实现通道切换，从而减少了故障之后的切换次数。在收到 251 类包之后，接收方应通过冗余链路发送和接收所有的数据包。

4.6.1.4.3. 确定在冗余链路上传输数据之后，连接发起方的适配层应当试图重建正常链路连接，并在成功建立之后，通过“非活动链路的生命信息”包来监测正常连接。此时，除非在冗余链路上检测到故障，不应将数据传输恢复到正常链路上。在需要恢复时，应当通过任何一方的适配层在正常链路上发送一个 SwitchR2N ALEPKT（类型 253）来实现，用户数据可以包含在该包中正常发送。

4.6.1.4.4. 如果活动链路为冗余链路并且发生故障，则应当通过任一方在正常链路上发送一个包类型设置为 253 的 SwitchR2N ALEPKT 来实现切换。用户数据可以包含在该包中正常发送。在收到 253 类包之后，接收方应通过正常链路发送和接收所有的数据包。

4.6.1.4.5. 确定在正常链路上传输数据之后，连接发起方的适配层应当试图重建冗余链路连接，并在成功建立之后，通过“非活动链路的生命信息”包来监测冗余链路连接。此时，除非在正常链路上检测到故障，不应将数据传输恢复到冗余链路上，。在需要恢复时，应当通过任何一方的适配层在冗余链路上发送一个 SwitchN2R ALEPKT（类型设置 251）来实现。

4.6.1.4.6. 生命信息的消息类型取决于链路是否被选择用于传输数据，而不是取决于是否做为指定的正常或冗余通道。用于传送数据的链路始终使用 255 类包，作为备用的链路始终使用 254 类包。此时，如果切换消息丢失或延迟，系统就能够得以快速恢复。（如果在切换消息发送之后，在新的活动链路上接收到 254 类包，则应发送另一个切换消息，直至在活动链路上接收到数据或 255 类包）

4.6.1.5. 正常释放

4.6.1.5.1. 当安全层实体希望中止其与远端实体的关系时，可以通过 T-Disconnect.Request 来实现。此时 DI ALEPKT 将在在活动链路上进行交互。一旦收到 T-Disconnect.Request 原语，本地适配层应当通过 TCP-CLOSE 原语关闭所有相关的 TCP 连接。一旦所有数据（包括断开请求消息）传输完毕，TCP 连接应当在另一方向上也被关闭。

4.6.1.5.2. 4 类包（DI）仅允许在下列情况下被发送：

- 由发起方在发送 1 类包之后进行；
- 由应答方在接收 1 类包之后进行。
- 注：为了触发远端实体的对等 T-Disconnect.Indication 事件，所有与该安全连接相关的 TCP 连接必须关闭。远端 TCP 实体向其适配层实体报告连接已经通过 TCP_CLOSE 事件关闭。当正在关闭的 TCP 链路为最后一个用于该安全连接的链路时，适配层应发送一个具备相关原因代码的 T-Disconnect.Indication 到安全层。

4.6.1.6. 异常释放

4.6.1.6.1. 如果本地安全层实体异常导致连接释放，则所有相关的 TCP 连接均应通过 TCP-CLOSE 释放。最后一个 TCP-CLOSING 事件应立即触发远端的适配层向其用户发送 T-Disconnect.Indicatio（如 4.7 所述的正常释放，但具体代码不同）。

4.6.1.6.2. TCP 链路的异常释放情况更为复杂。

4.6.1.6.3. 如果最后一个 TCP 链接被释放，则应当发送带有相应原因代码的 T-Disconnect.Indication 给 TS 用户

4.6.1.6.4. 如果释放的连接为非活动链路（活动链路仍然可用），则适配层应当试图重新连接到远端实体，而无须通知适配层用户。

4.6.1.6.5. 如果关闭的连接为活动链路，而非活动链路仍然在运行，则适配层应首先将所有数据传输切换到非活动链路，然后试图重新建立原活动链路的连接。

4.6.1.6.6. 如果向适配层报告丢失的 TCP 连接已经是最后一个链路，则应触发适配层发送带有相关原因代码的 T-Disconnect.Indication，此时，所有与该安全连接相关的 TCP 连接都应该已经被关闭。远端 TCP 实体应向其适配层显示连接已经通过 TCP-CLOSING 事件关闭，如果 TCP 连接错误，则显示连接已经通过 TCP-ERROR 事件关闭。

4.6.1.6.7. 如果适配层认为连接由于某种原因而不能使用，则可以通过 ABORT 原语删除该连接。除非该连接为最后一个可用的 TCP 连接，该事件不应当通知安全层。

4.6.2. D 类服务

4.6.2.1. 连接

4.6.2.1.1. 对于 D 类服务，所有的 ALEPKT 均在全部的 TCP 连接上被传输，以用于支持两个实体间的一个安全连接，所有这些 TCP 连接应同时在各自的物理链路上被创建。

4.6.2.1.2. 可以使用一条，两条或更多的物理链路。本节中均假设存在两条物理链路。在使用其他数量的物理链路时，下文中的需求应根据实际情况予以修正。

4.6.2.1.3. 例如：在使用一条物理链路（非冗余）并只建立一个 TCP 连接时，N/R 标识应当设置为‘1’。

4.6.2.1.4. 包含用户数据的连接建立包 AU1 ALEPKT（CR）和 AU2 ALEPKT（CC）的传输序列号为 0，在两条 TCP 传输连接上以相同的方式发送。

4.6.2.1.5. 在连接建立阶段，ALE 实体应当使用接收到的首个 AU1 和 AU2 ALEPKT，并丢弃此后接收到拷贝。

4.6.2.1.6. D 类服务的代码为 0x03。

4.6.2.2. 数据传输

4.6.2.2.1. 所有的数据 ALEPKT 都应当在两个 TCP 连接上传输，并在两个通道中使用相同的传输序列号。

4.6.2.2.2. 接收方应当在两个 TCP 连接上检查是否接收到 ALEPKT。

4.6.2.2.3. 任何 ALEPKT，如果其传输序列号小于或等于与已经向 TS 用户传送过的 ALEPKT 的传输序列号，都应被接收方丢弃。

4.6.2.2.4. 在消息接收侧，至少存在以下两种可能的处理方式：

a） 只要收到消息的传输序列号大于上一次向用户传送的消息中的传输序列号，就向用户传送该消息。

b） 如果消息的传输序列号等于上一次向用户传送的消息中的传输序列号加 1，则向用户传送该消息，如果传输序列号大于最后一个传输序列号加 1，则丢弃该消息。

4.6.2.2.5. a）和 b）可以选择配置。

4.6.2.2.6. 如果不允许 ALEPKT 丢失，则必须选择 b）。

4.6.2.2.7. 一个 TCP 通道上的数据传输发生故障时，适配层不需要解决这一问题，而是应当简单地释放连接，并拒绝接受该通道的数据；在实施中，可以选择将故障报告给 ALE 用户。

4.6.2.3. 冗余管理

4.6.2.3.1. 当发送方不能够在某 TCP 连接上发送数据，或者 TCP 连接出错时，该 TCP 连接应当被关闭并继续通过其它链路传送数据 ALEPKT。连接发起方的 ALE 应试图重建任何失效的 TCP 连接。

4.6.2.3.2. 在接收方，任何被视为已经失效的 TCP 连接都应当关闭，此外，除非其为连接的发起方，不应再采取进一步的动作，如果是连接的发起方，则应试图重新建立该连接。

4.6.2.3.3. 所有 ALEPKT 都应当从余下的 TCP 连接中获得。

4.6.2.4. 连接监测

4.6.2.4.1. D 类服务中应使用 TCP 提供的标准生命保持特性来执行连接与物理链路监测。

4.6.2.4.2. 双方的 TCP 实体都应当具备生命保持特性。

4.6.2.4.3. 注：当应用程序采用周期发送数据的形式时，不要求 TCP 层进行连接监测。

4.6.3. ALEPKT 概述

4.6.3.1. 本节概括了 ALE 使用的所有 ALEPKT。

ALEPKT 类型	ALEPKT 名称	作用	用户数据	链路	服务类别
1	AU1（CR）	请求到对方的连接	AU1	D 类：正常&冗余链路 A 类：活动链路	A&D
2	AU2（CC）	从对方接受连	AU2	D 类：正常&冗余	A&D

		接		链路 A 类：活动链路	
3	DT	传输用户数据	SaPDU	D 类：正常&冗余链路 A 类：活动链路	A&D
4	DI	释放连接	DI SaPDU	D 类：正常&冗余链路 A 类：活动链路	A&D
251	SwitchN2R	将数据通信切换到冗余链路	无或 SaPDU	冗余链路	A
253	SwitchR2N	将数据通信切换到正常链路	无或 SaPDU	正常链路	A
254	KANA	非活动链路的生命信息	无	非活动链路	A
255	KAA	活动链路的生命信息	无	活动链路	A

表 10：ALEPKT 一览表

4.7. 适配层管理—ALEPKT 错误处理

4.7.1.1. 监测/诊断

4.7.1.1.1. 适配层一般对 TS 用户实体屏蔽通信故障，只有在故障不能解决时才通知用户。

4.7.1.1.2. 来自网络报告的不可恢复错误应当经由适配层通知安全层。

4.7.1.1.3. 检查到 ALEPKT 校验和错误时，适配层实体应重启相应的 TCP 连接。

4.7.1.1.4. A 类服务的正常连接中 ALEPKT 出错时，应当根据 4.6.1 节所述的原则进行处理。

4.7.1.1.5. 当适配层不能维持透明服务时，应当对下列错误加以处理，并向用户报告。表 11 提供了一些可能的“原因”和“子原因”描述，并说明了在传输层采取的相应的错误处理措施。在具体的实施中，也可以采取其它处理措施，并且“原因”和“子原因”的编码也属于本地实施的范畴。

原因		子原因		处理措施
编码	描述	编码	描述	
0	正常释放	0		向用户发送包含此原因代码的 T-Disconnect.Indication
1	网络错误	1	未分配的号码； 号码格式无效	本地适配层生成错误报告，并通过发送包含此原因代码的

		2	通道的服务质量不可接受	T-Disconnect.Indication 通知用户
		3	由于其它原因导致无法建立物理连接	
		4	地址信息与请求的服务质量不兼容。	
2	网络资源不可用	1	没有可用的通道	适配层生成时临时错误报告，并通过发送包含此原因代码的 T-Disconnect.Indication 通知用户
		2	网络拥塞	
		3	其它原因	
3	服务或选项暂时不可用	1	无法获得指定的服务质量	适配层生成时临时错误报告，并通过发送包含此原因代码的 T-Disconnect.Indication 通知用户
		2	无法获得指定的承载能力	
5	原因未知	0		被呼叫方适配层生成错误报告，并通过发送包含此原因代码的 T-Disconnect.Indication 通知其用户
6	不支持的应用	1	被叫地址不支持请求的应用类型。	被呼叫方适配层生成错误报告，并向外发送以此原因代码作为用户数据的 DI-ALEPKT（4 类包）。 主呼叫方适配层应当通过带有相应原因与次级原因编码的 **T-Disconnect.Indication** 将错误报告给主叫应用。
		2	被叫用户未知（例如：无回复）	
		3	被叫用户不可用（例如：忙碌）	
7	内部错误	1	强制元素的丢失（例如：原语参数）	对错误进行记录。 丢弃无效的消息。 通过发送包含此原因代码的 T-Disconnect.Indication 通知用户
		2	状态错误	
		3	其他	
备注：保留其它所有原因/子原因。				

表 11：错误报告

4.8. 底层协议栈

4.8.1. 简介

4.8.1.1. 本规范不约束底层网络解决方案的实施，仅说明了实现互通的最低标准。

4.8.2. TCP 参数协商（强制性）

4.8.2.1. 为建立 TCP 连接，适配层必须考虑以下参数：

4.8.2.2. 最大传输单元（MTU）长度

4.8.2.2.1. 广域网链路的默认值为 576 字节，以太局域网为 1500 字节。这一参数直接映射到 TCP 参数中，并可确定消息传输单位的最大长度。超过这一 长度的消息应当通过多个 IP 数据报文传送。在不同的物理媒介中，为了优化性能，该项的取值可能是不同的。

4.8.2.2.2. 如果希望一个 ALEPKT 能够通过单个 IP 数据报传送的话，在使用 576 字节的 IP 默认数据报时（允许 64 字节用作帧头和帧尾—可参阅有关 D 比特和分段的备注）， ALEPKT 应当小于 512 字节，在使用以太局域网时，ALEPKT 应当小于 1460 字节，此时 IEEE 802.3 帧尺寸位于 64 与 1518 字节之间（假设帧头和帧尾总计达 58 字节）。

4.8.2.2.3. 这一参数不会限制能够传输的最大消息长度，即 65536 字节减去上述帧头尺寸。在本规范中，建议最大消息长度为 65000 字节（略低于允许的最大 SaPDU 尺寸—64kB）。注：也可以在 TCP 实体之间设定一个最大段长度（MSS），用于定义允许传输的最大 TCP 帧。

4.8.2.3. 生命信息确认定时器（仅针对 D 类服务）

4.8.2.3.1. 此项等同于 TCP 的超时参数，用于通知应用程序网络超时。此参数应根据应用需求进行取值。

4.8.3. 网络服务定义

4.8.3.1. 应在符合所有相关 RFC 标准的 IP 网络层上使用标准的 TCP 服务。

4.8.3.2. 本规范不限制使用较新的 IP 版本，如 IPv6，参见［RFC2460］。

4.8.4. 网络协议

4.8.4.1. 协议的实施应当符合描述了 IPv4 的 IP 标准［RFC0791］。

4.8.4.2. 不超过 4.8.2.2 所述 MTU 长度的消息不应在 IP 层进行分割，而应在一个传输单元中传输。

4.8.4.3. 为了达到最佳性能，应将 MTU 长度设置为与通道介质相关的最小长度，而不应该由安全设备连接本身决定其大小。

4.9. 适配层配置与管理

4.9.1. 总则

4.9.1.1. 本节定义了执行适配层协议与适配层管理所需的参数。

4.9.2. 定时器参数

4.9.2.1. “最大连接建立延迟”参数在建立连接期间用于监测不可接受的延迟。该参数值由应用需求决定，在本文中默认值为 40 秒（与 Subset-037 中定义的值相同，详见 5.3.1.1.5 节）。

4.9.3. 呼叫与 ID 管理（适配层和 TCP）

4.9.3.1. 6.9.3.1.1 寻址约定与映射原则应在具体项目中由双方协商确定。

5. 资料性附录

5.1. TCP 参数协商

5.1.1. TCP 服务选项

5.1.1.1. TCP/IP 的其它可选特性可用于提供特定服务等级。不过，应当注意的是，达到怎样的服务质量是一个实施的问题，而不是具体规定的。也就是说，满足服务质量的要求导致一系列可能的行为，包括：

- 在终端系统之间创建多重连接；
- 在 TCP 内使用选项，例如：push 和 urgent 标志（可能用于高优先级数据）；
- IP“服务类型”选项的设置，例如：

  优先 ：3 比特，标志 8 个优先级的其中一个。

  低延迟：D 标志—每一段均应选择具有最小延时的路由。

  高吞吐量 ：T 比特—路由器应当选择具有最高吞吐量的路径。

  高可靠性 ：R 比特—如果条件允许，网络应使用面向连接的链路。

  “成本”最低化：C 比特—（如可能）选择最低成本的路线。

5.1.1.2. 必须注意的是，“服务类型”参数仅能在中间系统（如路由器）支持这项功能时才起作用。

5.1.1.3. 从 TCP 传输到 IP 的下列参数也可能使用：

- DF 比特（不分段）——选择一个能够处理整个报文而不是分割处理的网络路由。注：如果设置之后，接收主机应要么接收整个数据报中的数据，要么什么也不接收。
- MF 比特（分段）——当消息被分段时，使用这一参数表示还要到来更多分段消息。

5.2. 地址映射

5.2.1.1. 本节提供了地址映射的一个可能实施的实例。

5.2.1.2. 在收到 TS 用户实体的连接请求时（带有 4.4.4 节所述的地址信息），适配层采取下列行为。

5.2.1.3. 如果 **T-Connect.Request** 包含全部寻址信息（即安全层用户连接请求的“网络地址”域含有一个非空非“0”的数值），适配层则会试图运用这一信息建立传输连接。（注：为了成功连接，必须定义一个正确的 TCP 地址。）

5.2.1.4. 如果在 **T-Connect.Request**（网络地址为空）中仅包含 CTCS-ID，适配层进行必要的查找（通过地址表或者其它数据库），以获得连接所需的完整 TCP 地址。

5.2.1.5. 如果 **T-Connect.Request** 中的“应用类型”确定，则本地适配层试图建立两个连接至远端适配层。

5.2.1.6. 如果网络地址或者 CTCS-ID 没有包含在 **T-Connect.Request** 原语中，或者存在一个映射错误，则必须采用一种常规做法向最适合的系统来建立呼叫（或者通过发送给监督管理系统一个适当的消息来拒绝）。

5.2.1.7. 网络地址（端口和 IP 地址）本身并不足以识别一个特定的远端传输服务用户实体。因此有必要使用一个特定的标识符或者地址限定（包含在每个 ALEPKT 中的“应用类型”）来表明请求的传输服务用户实体类型。

5.2.1.8. 适配层实体和服务用户实体被绑定在 TSAP 中。每一个适配层服务用户实体可能被绑定在一个或者多个 TSAP 中。这属于实施的范畴。TSAP 和多路复用无关，每一个连接，无论其 TSAP id 是什么，都必须映射到一个独立 TCP 端口，以便建立一个真实的连接。如果存在 A 类或 D 类服务关系，第二个 TCP 连接也必须建立和使用，如 4.6 节所述。

5.2.1.9. 如果一个传输服务用户实体（例如：安全层实体）希望与另一个传输服务用户实体建立连接，则必须向被叫传输服务用户提供寻址信息（通常为 CTCS-ID 和应用类型，可能为网络地址）。此信息必须被映射为 TCP 服务所需的格式和结构，以便建立连接，这由安全层实体在一个或多个 TSAP 处接入的适配层实现。

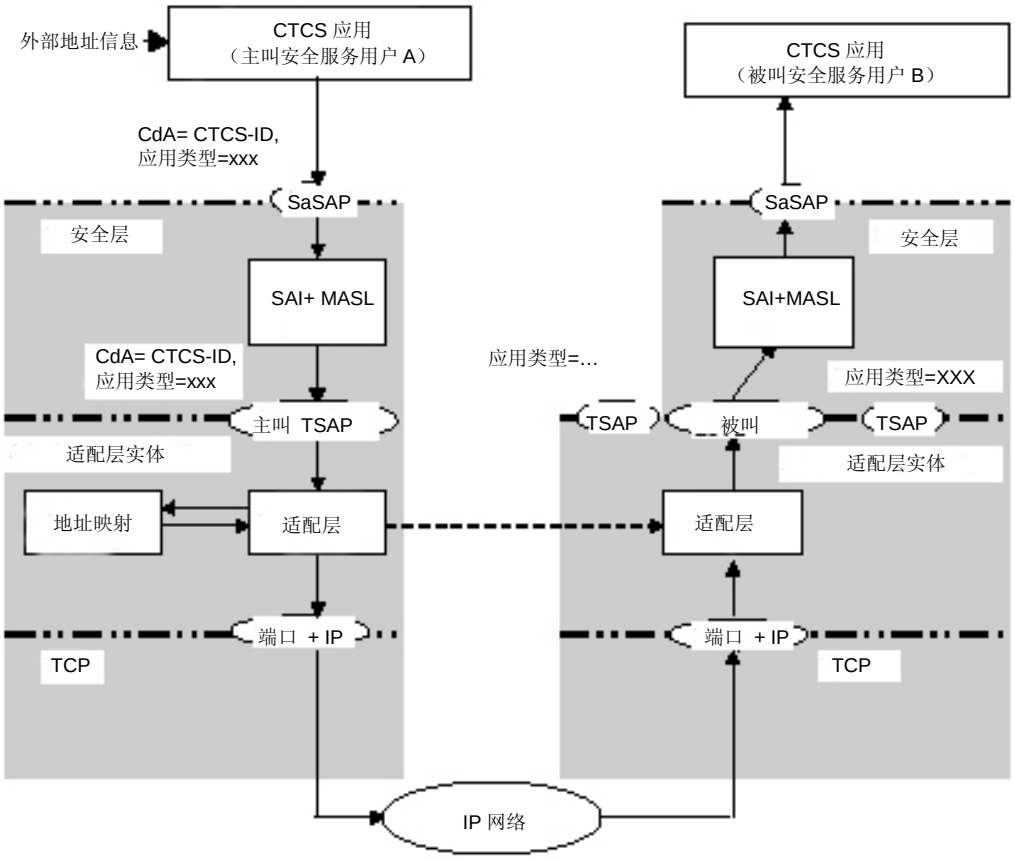


图 36：地址映射实例

5.2.1.10. 上图给出了在传输服务之间建立连接的过程中一个地址映射的实例。主叫 TS 用户实体（本例中即为安全层实体）从 CTCS 应用（CTCS-ID）中获取被叫地址。地址信息通过安全层传送给适配层的传输服务。适配层实体具有下列任务：

- 监听呼入连接，在每一个可能接受呼入连接的端口上执行 TCP 监听。根据正常的 TCP 惯例，接收的呼叫应被分配给相应的套接字，而监听行为则继续保持；
- 应从标识了被叫传输服务用户的寻址信息中得到被叫网络地址

（TCP/IP 地址），除非具体地址已经由安全层实体提供；

- 生成相应的 ALEPKT，参见本节后续部分的定义。

5.3. 数据链路层

5.3.1. 以太网

5.3.1.1. 逻辑链路控制

5.3.1.1.1. 用于将系统连接到网络的数据链路层应当具有适当的类型。

5.3.1.1.2. 对于以太网，应使用 LLC1 无确认无连接服务对 IP 网络层提供支持，此标准协议用于互连基于 CSMA/CD LAN 的 IP 设备。

5.3.2. 介质访问控制

5.3.2.1. 应当根据标准并适应相应的介质。

5.3.3. 广域连接

5.3.3.1. 广域连接可使用 X.25、HDLC、帧中继或 PPP、或根据使用的介质采用的其它数据链路协议。

5.4. 密钥管理

5.4.1. 范围

5.4.1.1. 为保证互操作，必须实现密钥管理（KM）功能。

5.4.1.2. 在铁路运输管理系统中，当一个安全信号设备试图同另一个安全信号设备通信时，应当能够确认通信是与一个已授权的安全信号设备建立的，反之亦然。因此，也需确认安全信号设备之间交互的所有信息的真实性和完整性。

5.4.1.3. 本指导涵盖了 Subset-037 中定义的密钥管理，描述了一般概念，原理和功能，用于管理 MASL 层的加密。

5.4.1.4. 一个 KM 域中对于轨旁实体的密钥管理流程和特性属于铁路运营实施的范畴。

5.4.2. 密钥管理概念和原理

5.4.2.1. 简介和背景

5.4.2.1.1. 基于一个标识和认证（I&A）对话，可确认通信双方实体的身份。为了确保防护的完整性，这一过程应在每次对等实体间启动的新的有效会话时进行。

5.4.2.1.2. 在每次成功的 I&A 对话之后，数据将通过消息验证码（MAC）保护。MAC 通过实际通信双方之间共享的密钥进行计算。

5.4.2.1.3. I&A 对话和 MAC 计算过程在 Subset-037 的安全功能模块中进行了详细描述。这些过程基于使用密钥的密码计算技术。但是，Subset-037 没有指定任何生成、分配或更新密钥的方法。

5.4.2.1.4. 此外，以上安全机制的完整有效性取决于密钥的保密性。这只有在根据具体实施和铁路运营场景明确了密钥管理功能和系统安全策略后，才能够得到保证。

5.4.3. KMS 的各个阶段和参与方

5.4.3.1. 应当考虑两个不同的阶段来确定 KMS 涉及的各参与方：

1. 在系统开发和试运行阶段（即：在授权投入运营阶段之前）。
2. 在系统运营阶段。

5.4.3.2. 在第 1 阶段，涉及到下列各方：

- CTCS 设备供应商；
- 签约实体。

5.4.3.3. 在第 2 阶段，涉及到下列各方：

- 密钥管理中心（KMC）；
- 基础设施管理方（即：轨旁 CTCS 设备操作员）。

例如，基础设施管理方能够实施 KMC 的功能，不过，其它公司（即使不属于“铁路领域”）也可以提供此项服务。

5.4.3.4. 在本指导中，参与方的概念仅用于识别一种角色和一系列类似的责任。没有对其它方面进行假设。换言之，两个或两个以上的不同参与方可能为同一组织，或者同一组织的不同部门，或者独立的单位等等。类似的，涉及 KMS 功能任何部分的组织可以通过本指导范围之外的协议和合同等手段来协调它们之间的关系以及相应的责任。

5.4.4. 一般原则

5.4.4.1. 下图概括说明了用于信号安全设备间安全通信的 KMS 的内容：

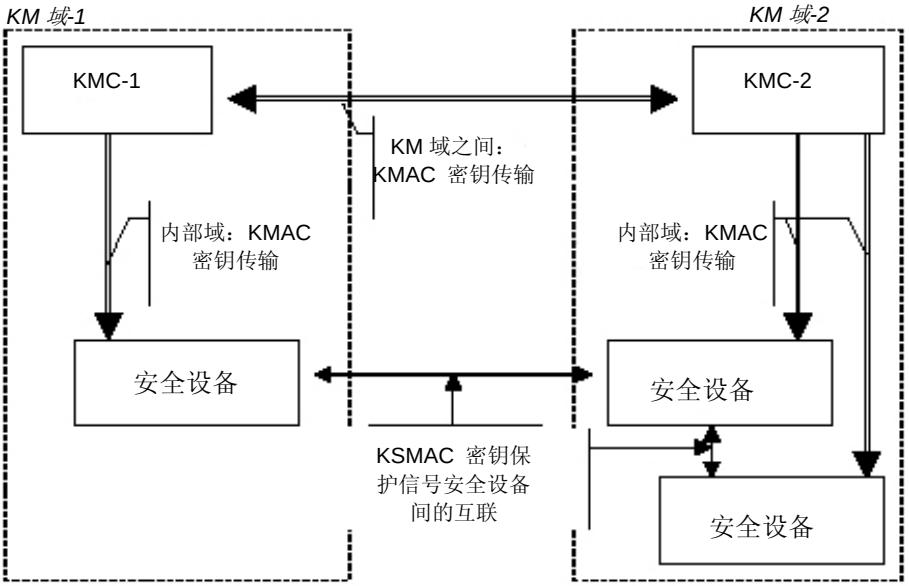


图 37：密钥管理内容图示

5.4.4.2. 图中所示 KMAC 的传输用于分配、更新和删除 KMAC。

5.4.4.3. 一个本地 KMC 负责对本区域中所有的信号安全设备生成、验证、分配、更新和撤销 KMAC。因此，每个信号安全设备应当仅通过本地 KMC 进行密钥管理。

5.4.4.4. 为了实现 KMC 互联，KMC-KMC 的接口应参照 Subset-038,其中描述了原理和过程，包括使用必需的附加密钥来交换 KMAC，和实现跨 KM 域的行车运营。

5.4.5. 密钥层级

5.4.5.1. 这一准则符合［Subset-037］中定义的密钥层级。

等级	用途
3：传输密钥 KTRANS	对 KMC 和通信系统之间的 KMS 通信进行保护。每个 KMC-信号安全设备实体的关系需要一个 KTRANS。
2：验证密钥 KMAC	在 MASL 安全连接建立期间，验证信号安全设备。每个信号安全设备间的通信连接需要一个 KMAC。
1：会话密钥 KSMAC	验证信号安全设备间通过安全连接传输的数据。该密钥来自于建立安全连接期间的 KMAC，并仅在本次安全连接期间有效。

表 12：密钥层级

5.4.5.2. KTRANS 密钥包括两种密钥：K-TRANS1 用于保护 KMC 与信号安全设备之间所交互消息的真实性和完整性。K-TRANS2 用于加密保护 KMC 与信号安全设备之间交换的 KMAC。

5.4.5.3. 下表总结了不同密钥的类型以及各自的用途：

涉及的实体	识别与验证所用的密钥	消息验证所用的密钥	加密所用的密钥	备注
安全设备-安全设备	KMAC	KSMAC	——	安全设备间互通
KMC-安全设备	——	KTRANS1	KTRANS2	KM 域内部 KMAC 传输

表 13：定义的密钥使用类别

5.4.6. 密钥分配

5.4.6.1. 信号安全设备应为每一个与其他安全设备的连接配备 KMAC。

5.4.6.2. KMAC 的分配应根据系统安全策略由 KM 域负责。

5.4.6.3. 如果信号安全设备间采用开放式网络进行通信，则每个信号安全设备间的连接关系都应分配不同的 KMAC。

5.4.6.4. 如果采用封闭式网络，则 KM 域中的所有的信号安全设备连接关系可使用同一个的 KMAC。

5.4.6.5. KMAC 可以安装于不同 KM 域中的信号安全设备中：在本地 KM 域和任何外部 KM 域。此时，KMAC 的生成和分配必须获得本地和外部 KM 域管理员的一致同意。KM 域之间的 KMAC 交换详见［Subset-038］。

5.4.7. 基本的 KM 功能

5.4.7.1. 功能列表

5.4.7.1.1. 为了在 KMC 与安全设备之间进行安全和一致的密钥交换，密钥管理系统应支持以下列功能：

- 生成和验证 KTRANS
- 分配 KTRANS
- 生成和验证 KMAC
- 分配 KMAC
- 更新 KMAC
- 删除 KMAC
- 密钥存档和 KM 实施

注意： 安全设备 KMAC 与另一个 KMC 之间的交换不在本指导原则的范围之内（详见 Subset-038）。

5.4.7.2. KTRANS 和 KMAC 的生成和验证

5.4.7.2.1. 只有在一个完善的组织和安全环境中的授权人员和程序才能为加密/解密或者认证产生密钥。这些密钥应当随机生成，即要具有不可预测性。

5.4.7.2.2. 生成的所有密钥都应当进行检查，以保证其不是弱密或半弱密。

5.4.7.2.3. 应当根据可互操作的 ERTMS 应用的安全需求由 KMC 来生成和验证 KTRANS 和 KMAC。具体的技术解决方案无需统一（只要确保安全即可）。

5.4.7.3. 分配 KTRANS

5.4.7.3.1. KM 域中密钥的分配应当在 KMC 和安全设备之间进行。密钥是否被盗或丢失应当进行检测，未授权人员不得参与修改密钥。

5.4.7.3.2. 发布 KMC 的管理员将指出 KTRANS 打算提供给哪一对安全信号设备联系使用。

5.4.7.3.3. 接收方设备会确认接收并采取必要的措施使得 KTRANS 生效。

5.4.7.3.4. 密钥应保密发送，通过一些人员的干预安装在安全设备中。

5.4.7.4. 分配 KMAC

5.4.7.4.1. 发布 KMC 的管理员将指出 KAMC 打算提供给哪一对安全信号设备联系使用。

5.4.7.4.2. 接收设备会确认接收并采取必要的措施使得 KMAC 生效。

5.4.7.4.3. 通过 KTRANS2 加密之后的密钥应保密发送。

5.4.7.4.4. KMC 应负责将密钥安装在设备实体中。安装应当以安全的方式进行并包括所有相关的密钥信息。

5.4.7.4.5. 密钥的存放应当保持其可靠性和机密性。

5.4.7.5. 更新 KMAC

5.4.7.5.1. KMC 管理员决定何时适合更新 KMAC。应当根据预先制定的密钥更新计划，或者在检测到危害情况时（丧失机密性时）做出密钥更新决定。

5.4.7.5.2. 在维护和正常运行期间都应当能够更新 KMAC。

5.4.7.5.3. 注 1：在维护期间更新旨在引入或者移除安全设备实体。

5.4.7.5.4. 注 2：在运行期间更新旨在不干扰正在运行的安全设备实体。

5.4.7.6. 删除 KMAC

5.4.7.6.1. KMC 负责删除密钥。删除应当以安全的方式进行，并包含所有相关的密钥信息。

5.4.7.6.2. 密钥材料所有可能的拷贝都应删除，包括设备中已安装的密钥，除了 KMC 责任范围内存档的密钥以外。

5.4.7.6.3. KMC 管理员应当能够启动密钥删除程序。

5.4.7.6.4. 安全设备应当向请求的发起者确认密钥删除已经完成。

5.4.7.7. 存档密钥和 KM 实施

5.4.7.7.1. 所有密钥、密钥相关材料和相关的密钥实施都应当由 KMC 以可靠而且保密的方式进行，包括：

- 分配密钥给实体；
- 密钥的状态（当前使用，删除，损害）。

5.4.8. 缩略语与定义

5.4.8.1. 本节包含一些与密钥管理相关的附加缩略语和定义。

缩略语	定义
KM	密钥管理
KMAC	验证密钥
KMC	密钥管理中心
KMS	密钥管理系统（实施密钥管理）
KSMAC	会话密钥
KTRANS	传输密钥

术语	定义
验证	在两个实体之间用来确定实体的身份和信息来源
机密性	用来防止消息被未授权实体读取。

域	域由一个 KMC 和所有使用那个 KMC 进行密钥管理的车载与轨旁实体来定义。
密钥档案	用于密钥的长期存储，必须可信并且保密。
密钥删除	删除密钥，包括：所有相关的消息和拷贝。
密钥生成	密钥相关材料的机密生成，用于加密/解密和密钥推导。
密钥安装	密钥在实体中的保密安装。
密钥管理	密钥材料的生成、存储、安全分配、解除、销毁和应用。
密钥管理中心	负责密钥管理职能的功能性实体。
密钥材料	建立和保持加密关系所需的资料（例如：密钥）。
KMC 管理员	KMC 管理员应当负责一个域中的所有密钥管理职能。
安全环境	安全的地方或者特定的开发设备。在正常情况下，未授权人员不能够从这些专门的地方或设备读取任何消息。
安全性	通过所有措施进行保护，包括：行政措施在内，以防止偶然或者恶意更改或者披露数据。为了进行密钥管理，保护通常是保证密钥的机密性、可靠性和完整性。
有效时间	密钥有效的时间。

5.5. 校验和结果实例

包长度	版本	应用类型	TSeqNum	N/R 标识	包类型	CRC
001E	01	1A	0000	01	01	1089
0021	01	1A	0000	01	02	F38E
0011	01	1A	0001	01	03	8D12
002A	01	1A	0002	01	03	C6E0
000B	01	1A	0003	01	04	57A0

表 14：校验和结果实例——所有数值均为十六进制数值