基于 Asterisk 的 VoIP 开发指南——(1)实现基本呼叫功能

2008/06/12

说明：

1. 本文档探讨基于 Asterisk 如何实现 VoIP 的一些基本功能，包括基本呼叫功能的方案选取、主叫号码透传、如何编写 Asterisk AGI 程序、Radius 认证计费模块等。

2. 本文档 VoIP 软终端使用 X-Lite，其它终端均可以接入测试。

3. 文章内容仅供参考,转载请注明出处。

1 VoIP 系统相关协议和标准

由于 I P 电话技术标准的开发涉及多个领域，因此，VOIP 系统要想实现这些 IP 电话之间的通信，则必须提供支持这些协议的实现。目前主要涉及的协议如图 1-1 所示，其中除了 HTTP 是与 WWW 相关的协议外，其它的都是 VOIP 相关协议。

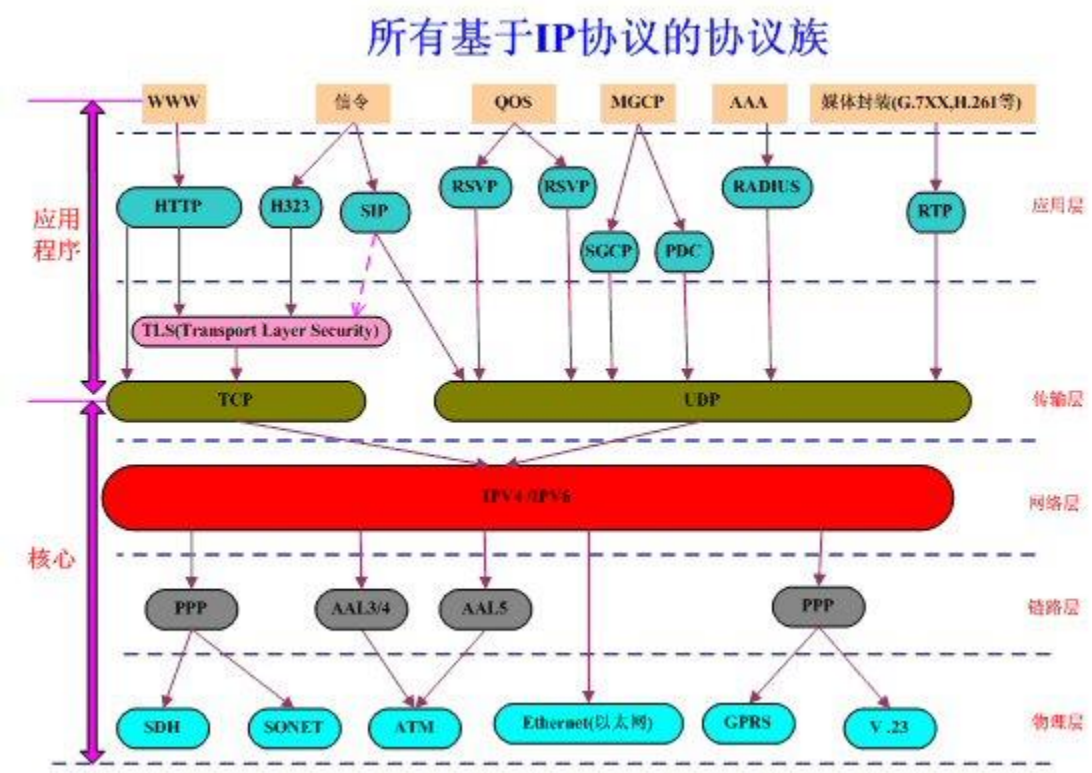


图 1-1 所有基于 IP 协议的协议族

由图 1 -1 可以看出，与 VOIP 相关的协议共分五层，每一层又由许多协议组成。目前有关 IP 电话制定的标准体现在应用层。而应用层又可分为信令控制协议、网关控制协议、媒体编码和传输协议和 QOS 协议等。

1. 信令控制协议，目前被广泛接受的 IP 电话控制信令体系主要有 ITU-T 的 H.323 系列和 IETF 的会话初始化协议(SIP)。

2. 网关控制协议，网关控制协议主要有媒体网关控制协议(MGCP)。该协议是为了解决目前 IP 电话负担过重，不能满足未来容量和业务扩展的要求而设计的。

3. 媒体编码，媒体编码主要有两类——视频编码和音频编码。视频编码主要有 H.261 和 H.263。音频编码主要有 G. 7xx 系列。

4. 实时传输协，实时传输协议有包括了实时传输协议(RTP)、实时传输控制协议(RTCP)、实时流协议(RTSP)和资源预留协议(RSVP)。相关的协议标准可以参考相应的网上资料，在这不细述。

2 Asterisk 简介

Asterisk 是一个开源的软件包，它可以运行 PBX 的所有功能，通常运行在 Linux 操作系统平台上。它不仅包含了 PBX 的功能，同时还有其它一些附加特性。Asterisk 可以用三种协议来实现 VoIP，同时可以与目前电话使用的标准硬件进行交互通信。

Asterisk 提供了附加的语音邮件服务、电话会议、交互语音应答、呼叫排队等基本电话服务。它还提供了多方呼叫、显示呼叫者 ID(显示主叫号码)等服务

Asterisk 在实现 VoIP 时，不需要任何附加硬件，DDD 软交换所采用的也是这种使用方式。但是，如果企业没有与 VoIP 语音网关运营商建立合作关系，想要自己构建这样的一个平台，那么要和数字电话设备与模拟电话设备进行交互通信，Asterisk 需要一个 PCI 硬件的支持，这个硬件生产商中最著名的是 Digium 平台提供的。

Asterisk 的结构基本上是十分简单，但是它不同于大多数的电话产品。基本上，

Asterisk 担任的是一个中间件的功能，它连接了底层的电话技术和上层的电话应用。Asterisk 为布局混合的电话环境提供了一致性。Asterisk 是开源 PBX (Private Branch eXchange)和 IVR (Interactive Voice Response)系统。使用兼容的 PCI 硬件，Asterisk 支持传统的电话线路，包括:TDM(Time Division Multiplexing), TI/El PRI/PRA&RBS (Robbed Bit Signal)模式、模拟电话线/模拟电话(POTS),ISDN(Integrated Services Digital Network)和 BRI(Basic Rate)与 PRI(Primary Rate)。

Asterisk 可以透明的桥接 VoIP 之间的一些协议，包括：会话初始协议(SIP-Session Initiation Protocol), H.323(国际电信工业会的一种标准)、IAX(Inter-Asterisk eXchange)媒体网关控制协(MGCP-Media Gateway Control Protocol)等其它一些协议。Asterisk 具有很大的柔韧性，特殊的 API 接口都围绕着 PBX 核心系统。这个核心处理着 PBX 内部之间的相互联系。每一部分都是清晰来自于协议、编码或内部电话使用的硬件接口的抽象。这些抽象的接口使 Asterisk 可以与任何的硬件和技术以及将来的硬件和软件技术完美的结合。从图 2-5 可以看出，Asterisk 由内部核心和外围动态可加载模块组成。内部核心由以下六个部分组成：PBX 交换核心模块(PBX Switching Core)、调度和 I/O 管理模块(Scheduler and I/O Manager)、应用调用模块(Application Launcher)、编解码转换模块(Codec Translator)、动态模块加载器模块(Dynamic Module Loader)和 CDR 生成模块(CDR Core)。

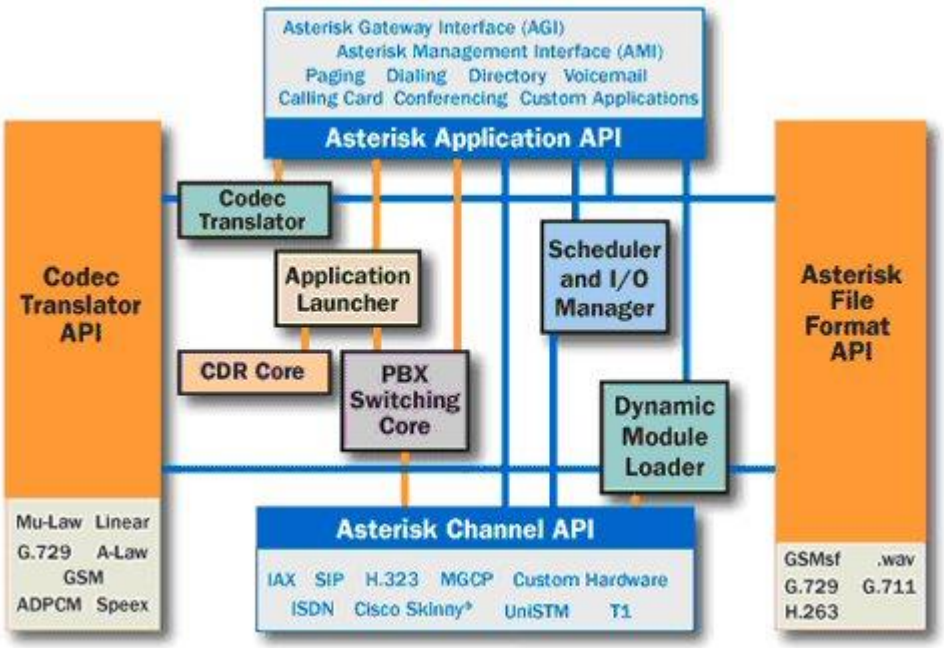


图 3-1 VOIP 通信系统功能模块图

Asterisk 提供了很多的基本拨号语法及应用的拨号函数，它共有 40 多个配置文件，通过 Asterisk 特有的语法修改特有的配置文件，才能实现通话的基本功能，同时可以实现针对不同的用户实现不同的通信功能。它的配置文件的源文件采用的是 C 语言编写。但是基于 Asterisk 的 Application API 编程接口，如 AGI，对外部的应

用程序可以使用 PHP，Python，Perl，Java 等语言编写。Asterisk 运行操作系统平台的 Linux 内核要求大于等于 2. 4. x 的版本。

3 VoIP 通信系统基本功能概述

如果是基于纯软件的实现方案，Asterisk 是构建 VoIP 项目的核心，系统中所有与用户呼叫有关的功能和管理都通过它来实现,包括 VoIP 各种协议的互通和配置，以及各种呼叫设备的配置文件。本文档所讨论的 VoIP 通信系统基本功能如图 3-2 所示，一般的 Asterisk 的任务包括了两方面，一是与呼叫有关的，包括基本呼叫处理、主叫号码透传、呼叫纪录和日志生成等，二是与呼叫控制有关，即终端用户的认证计费功能。

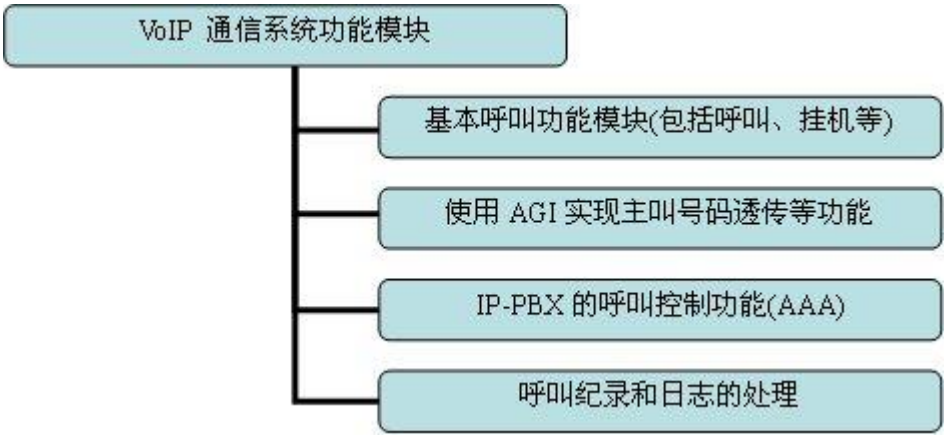


图 3-1 VOIP 通信系统功能模块图

4 基于 Asterisk 实现 VOIP 通信系统基本功能

PBX 是专用交换机，俗话叫集团电话。广泛地运用在企业办公机构中，极大地提高了企业的办事效率。但传统的 PBX 存在缺点有：

1. 它对新兴的 CTI（计算机与电话集成）和 VoIP 支持不够。

2. 传统的 PBX 都采用的是专用技术，缺乏开放性和标准性，并且价格昂贵。解决它的缺点的措施是 IP PBX 技术。

4.1 VoIP 通信系统方案选择

方案 1：模拟电话+语音网关+网守+PBX+模拟电话

语音网关型的应用是将 VoIP 语音网关的 FXO/FXS 接口同总部或分支机构的 PBX(小交换机或集团电话)直接相连，当需要打长途电话时，将话音转到 VoIP 网关上，通过因特网传输。用户在使用时只需在分机上先拨 IP 电话特服号(如可设为"8")，便可直接拨打 IP 电话。

在这个方案中，若要像普通电话那样的数字号码拨号，就得经过网守的路由管理，但对于中小企业这种设备太昂贵。网守处于高层，提供对端点的呼叫管理功能，是 IP 电话网络系统中的重要管理实体。网守的主要功能有：地址解析、接入控制、带宽管理、区域管理等四项基本功能;此外，还能提供呼叫控制信令、呼叫管理等其他功能。要构建一个稳定可靠的、实用的 VoIP 网，离不开 GK 的管理。

基于 VoIP 语音网关的复杂性与成本昂贵，本文档不使用这种方案。

方案 2：VoIP 电话/IP 电话+商业 IP-PBX 设备+PBX+模拟电话

IP-PBX 是一种基于 IP 的电话交换系统，它具有传统 PBX 交换机的所有功能，它的目标是取代企业内部原先的 PBX。这个系统可以完全将话音通信集成到公司的数据网络中，从而建立能够连接分布在全球各地办公地点和员工的统一话音数据网络。IP-PBX 最显著的特征是一个集成通信系统，因此，通过互联网，仅需要单一设备即可为用户提供语音、传真、数据和视频等多种通信方式，建立中、小型的呼叫中心。由于 VoIP 技术是将语音以数据包的形式在 IP 网络中进行传送，因此采用 VoIP 技术构建的通信平台，用户具有可移动的特性，形象的说就是同一个用户在 A 地用的是 011 的号码，到了 B 地还是 011 的号码，号码随着人走，VoIP 还支持语音信箱、多方会议、视频会议等传统 PBX 没有的功能。有助于移动办公和异地协同办公。

虽然说商业的 VoIP 设备或者软件，如华为 SoftCo 5816 IP 语音交换机、贝尔阿尔卡特 A5020，他们能够更容易、方便提供丰富的 IP-PBX 业务类型，也提供了数字中继接口与 PSTN 网络方便连接，并且只需要手动配置参数就可以投入到使用，不需要大量地编程，但是这种方案需要的成本跟方案 1 差不多，比较昂贵，并且灵活性不够，所以不使用。

方案 3：IP 电话/模拟电话+Linux PC 机+开源 IP-PBX+媒体网关+PBX+模拟电话

基于 PC 服务器+ Asterisk 呼叫管理软件的 IP-PBX 系统，Asterisk 作为 IP 电话网络的控制中心(PC 型 PBX)，该控制中心以软件方式工作，安装在一台服务器内。数字中继网关与原有传统 PBX 的 E1 中继接口相联(在这里媒体网关特指单独的 VoIP 落地网关运营商的语音网关设备，本文档让 Asterisk 与之对接实现 IP 与 PSTN 的完美转换)，VoIP 媒体网关提供的多路数字设置为中继模式，一端连接 PSTN 专网，一端对接 Asterisk 软交换 IP 侧。在控制中心的服务器上对 IP 电话号码进行分配。通过适当调整控制中心软件的参数以及添加、修改某些模块代码，即可完成本文档

最终完成的 IP 电话系统的建设。如果使用专用、商业的 IP-PBX 系统，可能会花费不菲，所以本文档使用方案 3。

本节的目标就是基于开源 IP-PBX Asterisk 设计 VoIP 电话系统的基本呼叫功能模块、认证计费功能模块、AGI 功能模块等。

4.2 VOIP 系统的基本组件

一般 VOIP 系统基本组件组成如图 4-1 所示。

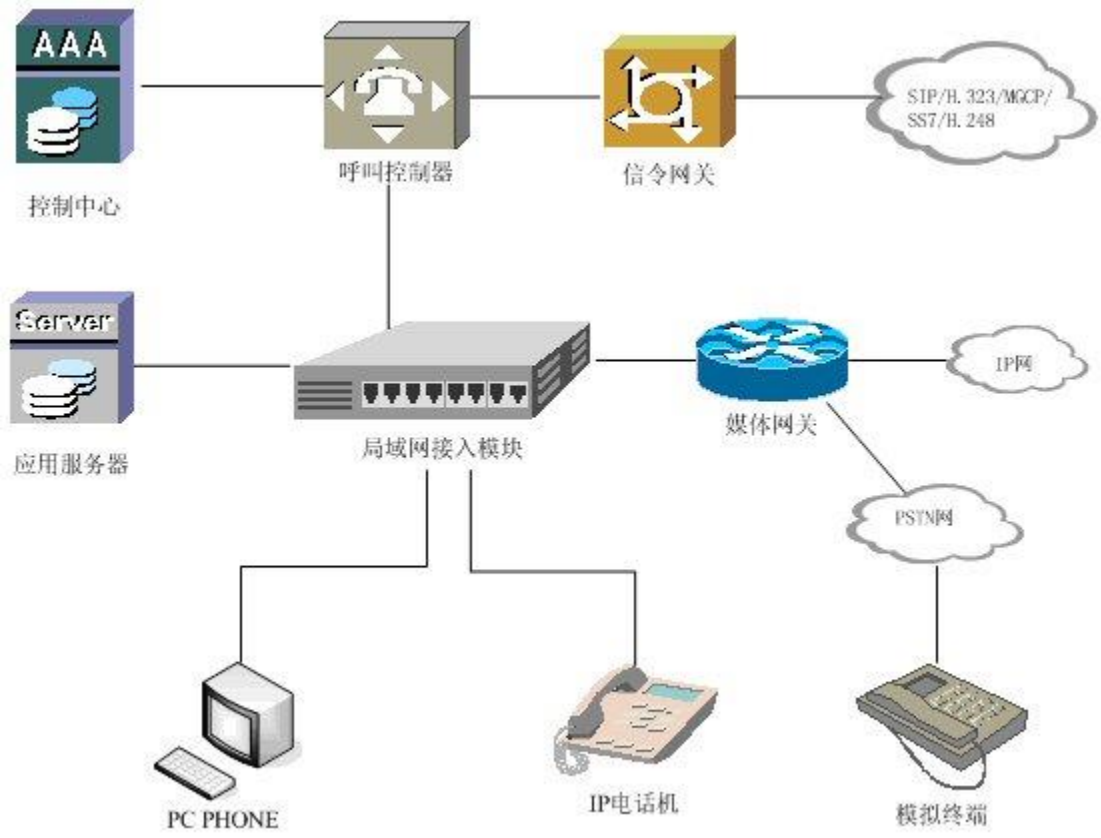


图 4-1 VOIP 系统的基本组成

IP- PBX 主要功能组件如图 4-1 所示，下面讨论这些基本功能组件。

1. 呼叫控制器

IP 电话系统的智能部分，它负责提供一切传统 PBX 系统的中心-PBX 交换机所能提供的服务。负责控制所有的呼叫建立和呼叫管理，能轻易提供大部分基本服务(如呼叫保持、呼叫转移、呼叫等待等)，以及配置电话的分机号码(Extension Number)、功能按钮、通话管理和路由决策功能。此外，它还控制所有的 PC 虚拟电话功能，如语音邮件、统一消息、自动话务员、交互式语音响应(IVR)和自动呼叫分配等。

2. 媒体网关

在 IP-PBX 系统中，媒体网关用来实现 IP 网络和传统电路交换网的通信，负责把呼叫转接到 PSTN 网，完成异种网络的电话呼入和呼出。它除了具有接通被叫的功能外，还可以把来自 PSTN 的呼叫连接到 IP 电话系统。媒体网关在整个 VoIP 系统中起着非常关键的作用。它不仅使 VoIP 系统能够连接 PSTN 用户，而且能够增加整个系统的可靠性，使系统具有处理紧急呼叫的能力。媒体网关还需要完成语音编码转换，通信协议转换以及 LAN/WAN-PSTN 之间的呼叫建立拆除等功能。

3. 局域网接入模块

提供 IP-PBX 系统各组件之间的连接。在局域网的环境中，该模块可以使用局域网交换机来代替。

4. 控制中心

包括资源管理系统、计费系统、网管系统、语音信箱等。

5. 信令网关

提供和其它信令网络的互通功能，可以支持 H.323，IAX，SIP，MGCP 等的全部或者部分功能。

6. 应用服务器

为 IP-PBX 系统提供增值应用。

7. IP 话机终端

包括终端部分软件 PC Phone，IP-Phone 等。

8. 模拟终端普通电话，传真机等传统模拟设备。

本文档 IP-PBX 系统的基本组件

1. VoIP 软终端 X-Lite，对应图 4-1 的 PC PHONE：PC 用户使用 SIP 软终端(目前只使 SIP 协议作为输入信令)访问 IP-PBX 服务器软交换后台，对普通座机或手机发起呼叫，实现 PC2Phone 的通讯。

2. IP-PBX(VoIP 软交换)，对应图 4-1 的呼叫控制器：基于开源软交换平台 Asterisk，接受 VoIP 软终端(或其它能发起 SIP/H323 请求的硬件终端)发起的呼叫信令、解析被叫号码、构建 VoIP 语音数据包发送到 IP 网络中。

3. 语音网关，对应图 4-1 的媒体网关、信令网关，因为实际使用中大部分的媒体网

关设备都集成了信令网关功能：提供模拟语音信号和 VoIP 信令的转换，即从 IP 网络进入的 VOIP 数据包被转换成模拟语音，通过与 PBX 相连的数字中继线路进入到 PBX(数字程控交换机)。

4. 数字程控交换机(PBX)：用于电话交换网的交换设备，它以计算机程序控制电话的接续，从语音网关的 E1 数字中继线路送出来的 7 号信令或 1 号信令或 PRI 信令以及模拟语音数据包进入到数字程控交换机。

4.3 VOIP 系统的软硬件平台

第 1 节已经简单介绍过 Asterisk，它是一个非常灵活的软件，可以轻松的安装在任何 Linux 平台上。Asterisk 的资源需求与其它的嵌入式、实时的应用系统很类似，都是通过优先级的方式来访问 CPU 和总线，并规定系统上的任何函数都不能直接调用比 Asterisk 优先的进程。对于非专业的系统而言，这也许不是很重要，如果目标是商用系统，这种优先级方式带来的性能上的缺陷会引起通话质量的问题。比如，经常出现回声、噪音等等。这种情况在手机超出服务区外的时候常常出现。由于对于 Linux 的内核代码和优化技巧不是很了解，选择一个高的配置，而不是重新对内核进行编程，是一个比较好的主意。

硬件平台

表 4-1 可以对系统的硬件配置有一个大概的认识

表 4-1 VOIP 系统的硬件配置

系统	并发通话数量	最小要求
非专业系统	<5	400M CPU 256M 内存
SOHO 系统	5-10	1G CPU 512M 内存
小型商用系统	10-15	3G CPU 1G 内存
中等商用系统	>15	双处理器，在分布式构架里采用多个服务器集群

对于要安装 Asterisk 的计算机，如果在预算有限的情况下，下面是一些建议：系统的稳定性及质量取决于所选择主板的结构设计，考虑使用服务器主板是一个很好的主意。比如服务器主板提供的 PCI 插槽有 3.5V 和 5.0V，服务器主板可以给主板提供更稳定的电压和电流。而且，语音卡常常会造成每秒 100 个以上的中断请求，所以对于主板来说，一定要仔细考察芯片组是否能供支持。

安装 PCI 显卡，而不是 AGP 显卡，因为 AGP 通道会造成内存的高占用率和 CPU 中断占用。如果采用工控机/服务器构架，根本没有安装显卡，而是使用 Console 来管理系统。对于 CPU 而言，由于 Asterisk 使用 CPU 进行信号的模数转换(也就是说

CPU 具有 DSP 的作用)，所以浮点运算能力是非常重要的，同时 CPU 的 L2Cache 也应该尽量的大。

1. 专有板卡的准备

如果准备连接 Asterisk 系统到任何电信设备上去，必须需要一个专有硬件的支持。板卡的主要功能是连接 PSTN 和 LAN/WAN。为了桥接电路交换的电信网络和包交换的数据网络，最流行和最经济的连接 PSTN 的方法是使用接口卡，接口卡有好几种，这里仅仅讨论常见的两种情况。

(a)模拟接口卡

PSTN 介入情况是普通的电话线或者模拟中继电话线的时候，就需要这种卡。

最流行的 Asterisk 模拟接口卡也许是 TDM400P(实际上这款卡和时分复用没有任何关系，仅仅是这么叫好听而已)，由 Digium 公司制造。TDM400P 是一个 4 口卡，可以插 4 块子卡，既可以提供 FXO 口，也可以提供 FXS 口。这个卡是贵的，当然最有名气。

(b)数字接口卡

如果需要多于 10 条电路或者需要数字连接的时候，就要购买或者寻找 T1 或 El 卡了。但是要注意，El 的接入的价格由信息产业部统一规定，在一些地区可以找到非常便宜的 PSTN 接入价格(落地价格)，有关这方面的内容在这不细述。

2. 硬件的需求

针对中小型公司，硬件的要求一般不是很高，普通的网络设备就可以满足要求。如果想采用很好的语音质量，可以采用专门的语音网关来处理语音信息。因为，采用通信的硬件设备目前大多是 PSTN 电话终端，因此要求附加一个硬件来将 PSTN 电话转化为“IP”电话。目前大多采用 ATA 设备来转换 PSTN 的电话终端。当然也可以使用 IP 电话，如 X-Lite 等。

对于本文档的 VoIP 开发环境说明如下：

1. 基于非专业系统的配置需求来进行硬件的配置，如 Ubuntu 7.04。

2. 不方便配备专有板卡，寻找某些地区的 PSTN 接入运营商，简称 VoIP 落地运营商或者 VoIP 落地网关，他们能够提供接口卡(数字接口卡、语音网关、媒体网关)。

软件平台

1.操作系统

对于中小型公司而言，Linux 可能是首选的操作系统。同时，Asterisk 源码便于在 Linux 中编译和运行，相对比较安全。在对于中小型软件研发而言，Linux 可能成为开发中的首选的操作系统。

2.Asterisk 软件包

Asterisk 的核心，主要由三个包组成：

1. Asterisk 主程序(Asterisk)

2. Zapate 电话驱动(zaptel)

3. PRI 库(libpri)

其余的还有一些其它的软件，如语音附加包等，都可以从开源的网站上下载。

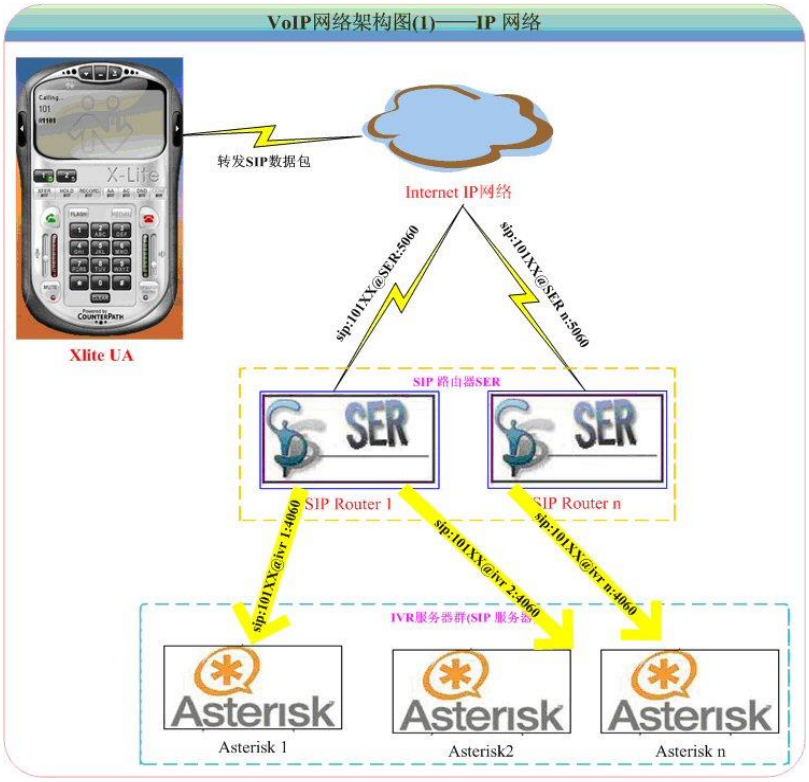
4.4 Asterisk 构建 VoIP 整合应用方案(基本呼叫功能的实现)

Asterisk 和 VoIP 的最初设计思想相同，其最终目的是减少长途通话的费用，实现通话的网络化，使 IP 网络成为一个可运载语音数据和其它数据的平台，实现语音网络和视频网络等完美地结合。

为了实现基本的呼叫功能，即摘机、挂机等功能，目前设计的拓扑结构为：VoIP 软终端--->SIP 代理服务器(SIP Proxy Server、SIP Redirect Server)-->转发到用户代理服务器(UAS，Asterisk)--->与 VoIP 语音网关通信(Cisco AS5300，华为 8010 等设备，一般有 2、4、n 个数字中继接口)---->通过 E1 中继线路对接数字程控交换机（华为 C&c08、贝尔 S1240 数字程控交换机）。

VoIP 系统第一部分——IP 侧的实现

VoIP 系统有两侧：IP 侧与 PSTN 侧，这部分主要是 IP 侧，发起呼叫请求的 VoIP 软终端是任何一个能够发起 SIP 请求的客户端软件或者硬件设备或者语音网关设备，网络拓扑结构图如下所示，图中每个结点在前几个小节均有描述。

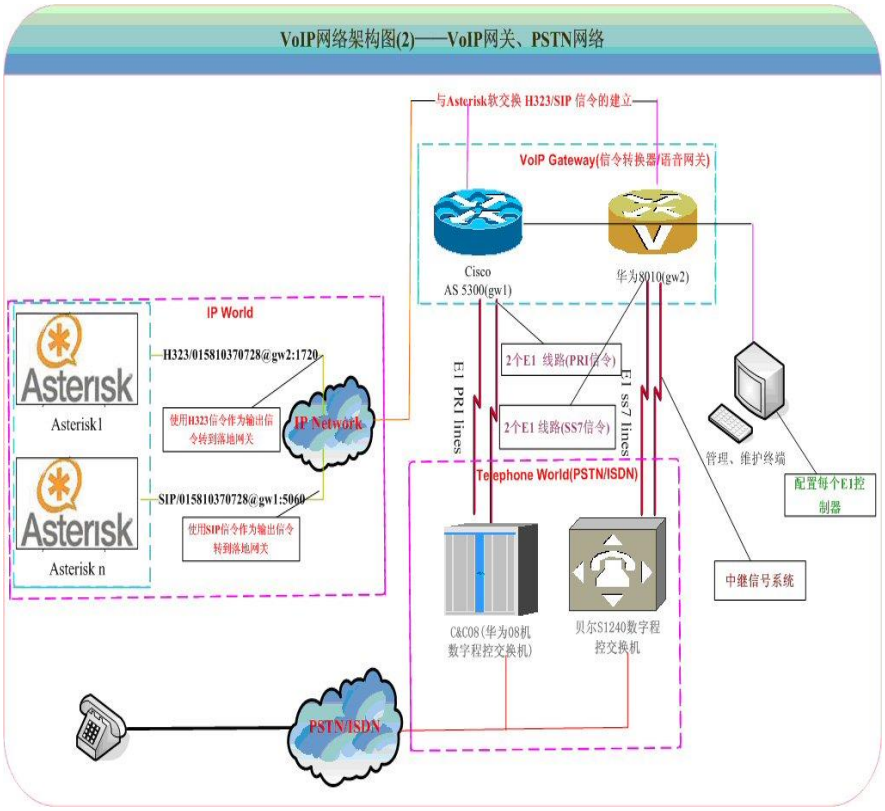


(点击看大图)

第二部分——VoIP 网关、PSTN 网络层

这部分是本文档所讨论的开源软交换平台(Asterisk)通信的 VoIP 语音网关，如图 4-3 所示，经过它出局的数字中继是一个 E1 接口（又称一个 PCM），是一对引自数字程控交换机的同轴电缆线，在电缆线上数据传输速率是 2.048 Mbps 可以同时容纳 32 时隙

*64Kbps 的语音数据。



(点击看大图)

结合图 4-2 与图 4-3，基本呼叫处理包含主叫摘机、拨号、通话、挂机、被叫挂机全过程。通常应用呼叫的方式可能是 PC 到 PC、PC 到电话(PSTN/IP)、电话(PSTN/IP)到电话(PSTN/IP)等方式。对于基本的呼叫流程是任何 PBX 都具有的，设计流程大多都一样。

1. 在 X-Lite 客户端输入完被叫号码后，点击呼叫按钮，用户听到 VoIP 会话应用程序播放的拨号音，然后开始拨号。

2. X-Lite 收集用户拨打的号码，并按照标准的 SIP 代理服务器到 VoIP 用户代理服务器 Asterisk。

3. Asterisk 动态的调用呼叫模块，进入到 Asterisk 内部的呼叫 Dialplan，并按照呼叫 Dialplan 中配置的被叫号码模板进行匹配。

4. 当成功匹配某个已配置的被叫号码模板后，号码将被映射至某语音网关（此语音网关直接连接目的电话或用户小交换机 PBX）。

5. IP-PBX Asterisk 的 IP 网络利用 H.323/SIP 协议向语音网关发起呼叫，并为每路呼叫建立通道，用以发送和接收语音数据。

6. 被叫语音网关接收 IP 侧的 H.323/SIP 呼叫，通过 PSTN 信令将呼叫传递到给 PBX 处理，直到接通目的电话。

7. 在呼叫连接过程中的 H.323/SIP 阶段，IP 侧与 PSTN 侧协商所使用的语音编解码方式，并使用 RTP 协议传递语音数据。

8. 呼叫中的任何一方挂机时，VoIP 会话应用程序 X-Lite 将结束会话。

4.5 基本呼叫功能环境搭建示例(SIP 与 H.323 互通)

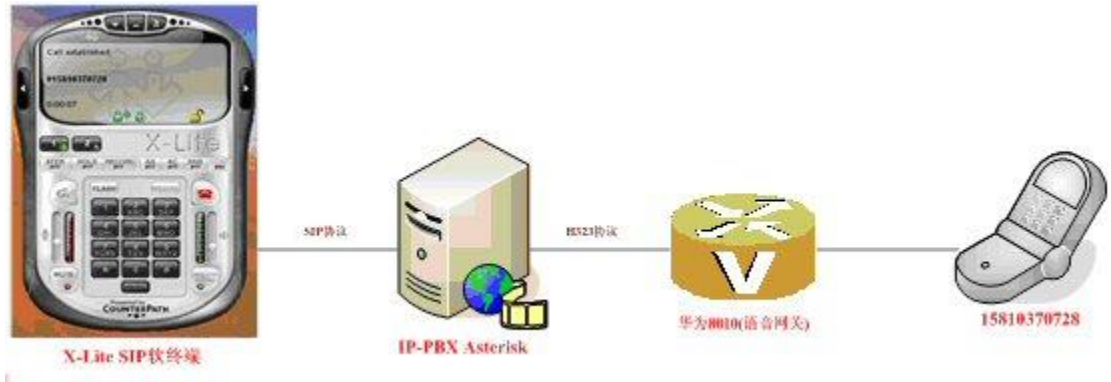
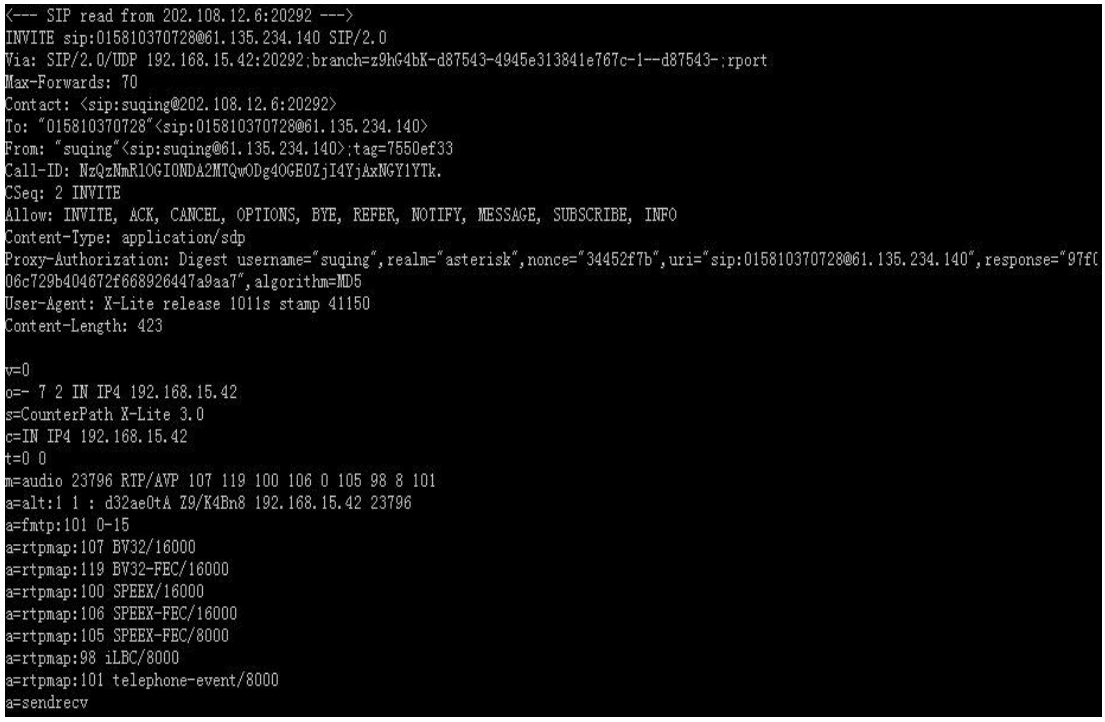


图 4-4 简单拓扑图

软终端 Xlite 注册到软交换 Asterisk 上

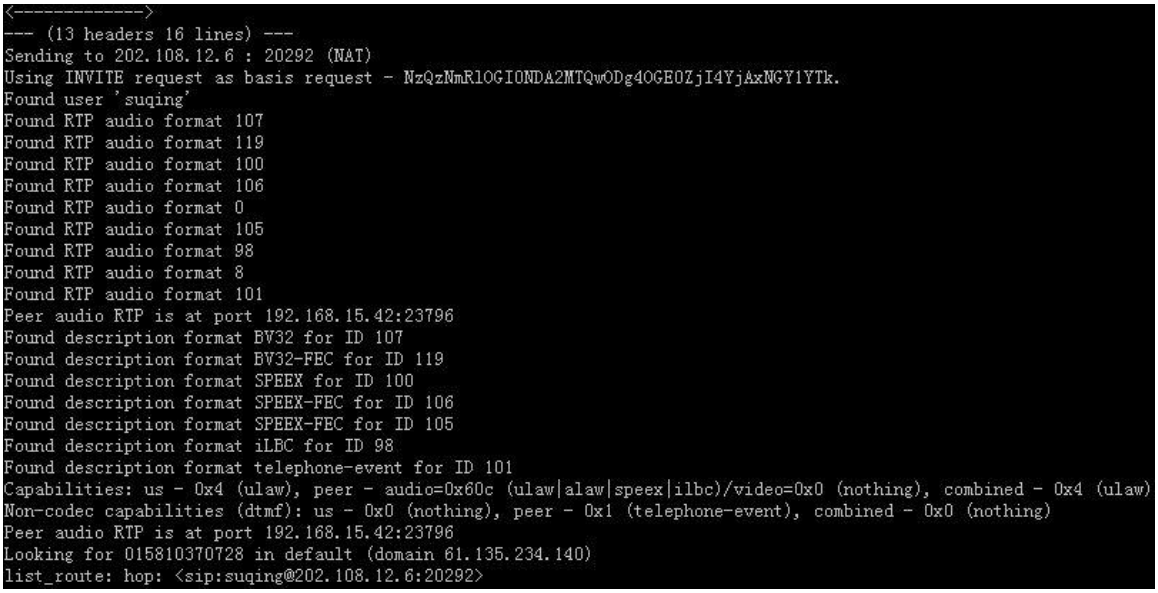
图 4-5 软终端设定

4.5.1 IP-PBX 服务器 Asterisk 抓包分析



(点击看大图)

图 4-6 SIP_REGISTER_1



(点击看大图)

图 4-7 SIP_REGISTER_2

X-Lite(UA) ------> asterisk PBX（读取 SIP INVITE 消息）

```
<--- SIP read from 202.108.12.6:20292 --->
INVITE sip:015810370728@61.135.234.140 SIP/2.0
Via: SIP/2.0/UDP 192.168.15.42:20292;branch=z9hG4bK-d87543-4945e313841e767c-1--d87543-;rport
Max-Forwards: 70
Contact: <sip:suqing@202.108.12.6:20292>
To: "015810370728"<sip:015810370728@61.135.234.140>
From: "suqing"<sip:suqing@61.135.234.140>;tag=7550ef33
Call-ID: NzQzNmRlOGIONDA2MTQwODg4OGE0ZjI4YjAxNGY1YTk.
CSeq: 2 INVITE
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
Content-Type: application/sdp
Proxy-Authorization: Digest username="suqing",realm="asterisk",nonce="34452f7b",uri="sip:015810370728@61.135.234.140",response="97f0
06c729b404672f668926447a9aa7",algorithm=MD5
User-Agent: X-Lite release 1011s stamp 41150
Content-Length: 423

v=0
o=- 7 2 IN IP4 192.168.15.42
s=CounterPath X-Lite 3.0
c=IN IP4 192.168.15.42
t=0 0
m=audio 23796 RTP/AVP 107 119 100 106 0 105 98 8 101
a=alt:1 1 : d32ae0tA Z9/K4Bn8 192.168.15.42 23796
a=fmtp:101 0-15
a=rtpmap:107 BV32/16000
a=rtpmap:119 BV32-FEC/16000
a=rtpmap:100 SPEEX/16000
a=rtpmap:106 SPEEX-FEC/16000
a=rtpmap:105 SPEEX-FEC/8000
a=rtpmap:98 iLBC/8000
a=rtpmap:101 telephone-event/8000
a=sendrecv
```

(点击看大图)

图 4-8 IP_INVITE_1

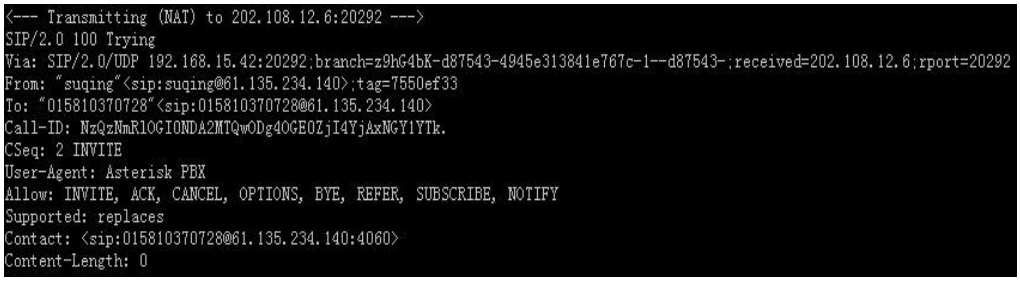
Asterisk PBX 针对上面的分析情况，响应 X-Lite 会话继续下去：

Transmitting to X-Lite（202.108.12.6）

```
<------------>
--- (13 headers 16 lines) ---
Sending to 202.108.12.6 : 20292 (NAT)
Using INVITE request as basis request - NzQzNmRlOGIONDA2MTQwODg4OGE0ZjI4YjAxNGY1YTk.
Found user 'suqing'
Found RTP audio format 107
Found RTP audio format 119
Found RTP audio format 100
Found RTP audio format 106
Found RTP audio format 0
Found RTP audio format 105
Found RTP audio format 98
Found RTP audio format 8
Found RTP audio format 101
Peer audio RTP is at port 192.168.15.42:23796
Found description format BV32 for ID 107
Found description format BV32-FEC for ID 119
Found description format SPEEX for ID 100
Found description format SPEEX-FEC for ID 106
Found description format SPEEX-FEC for ID 105
Found description format iLBC for ID 98
Found description format telephone-event for ID 101
Capabilities: us - 0x4 (ulaw), peer - audio=0x60c (ulaw|alaw|speex|ilbc)/video=0x0 (nothing), combined - 0x4 (ulaw)
Non-codec capabilities (dtmf): us - 0x0 (nothing), peer - 0x1 (telephone-event), combined - 0x0 (nothing)
Peer audio RTP is at port 192.168.15.42:23796
Looking for 015810370728 in default (domain 61.135.234.140)
list_route: hop: <sip:suqing@202.108.12.6:20292>
```

(点击看大图)

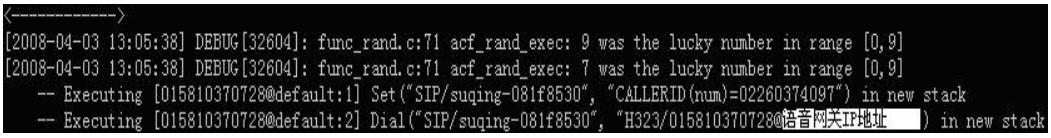
图 4-9 SIP_INVITE_2



(点击看大图)

图 4-10 SIP_TRYING

执行拨号方案



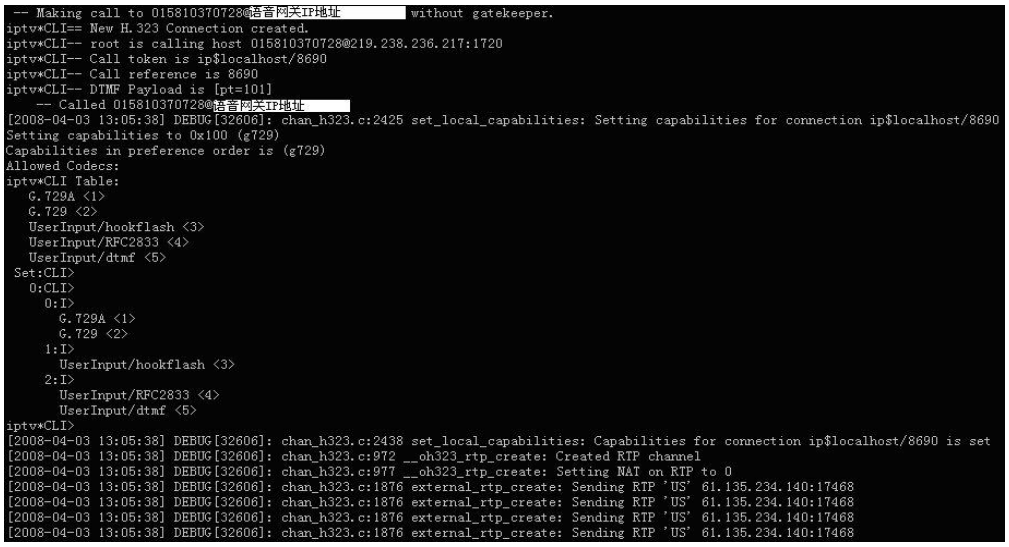
(点击看大图)

图 4-11 dialplan

Asterisk 然后开始与华为 8010 语音网关建立 H323 通信

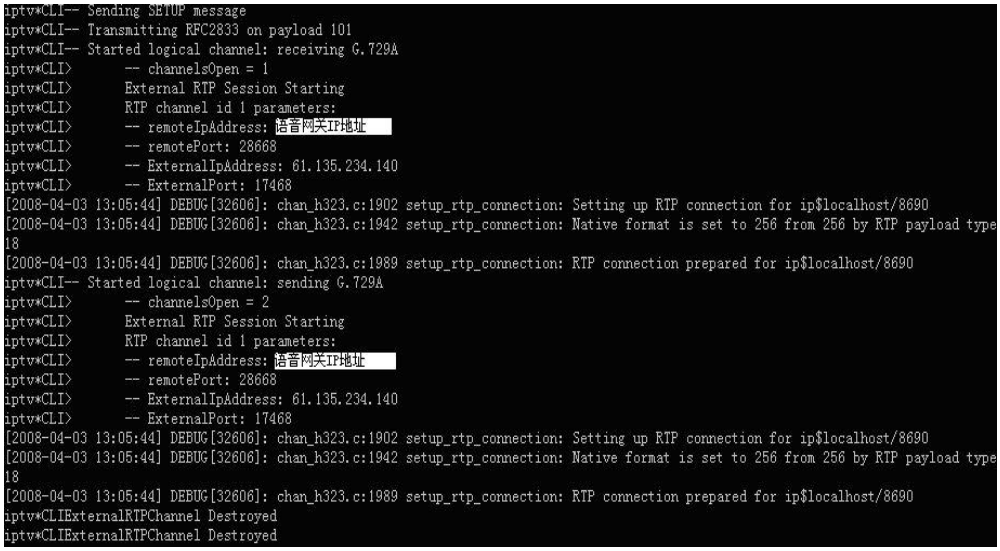
① Call set up.

H．225/Q.931 Call Setup



(点击看大图)

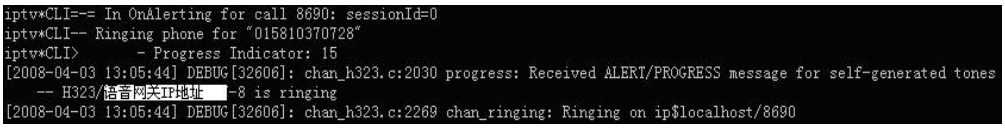
图 4-12 h323_SETUP_1



(点击看大图)

图 4-13 h323_SETUP_2

②ALERT/PROGRESS 表示被叫已经正在响铃…

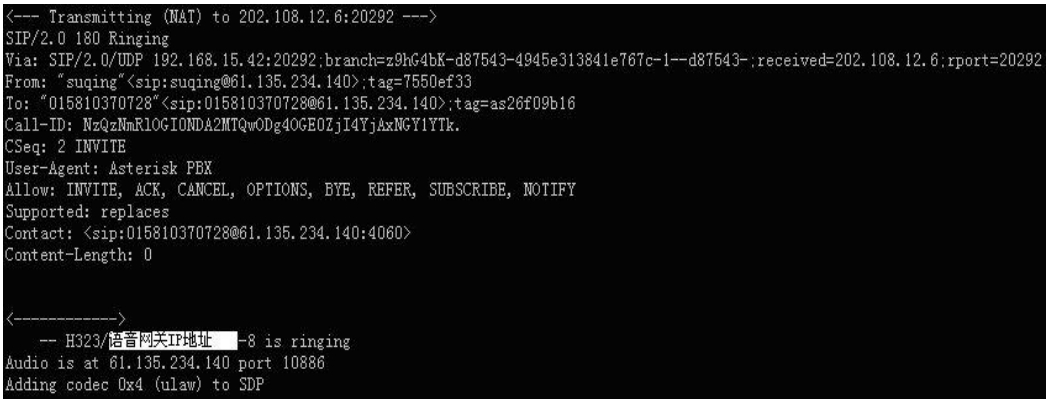


(点击看大图)

图 4-14 h323_3_ALERT

这时候，Asterisk PBX 将被叫手机正在响铃的信号以 SIP 消息 的形式发送到客户端 X-Lite，这是一种 sip_indicate 类型的 SIP 消

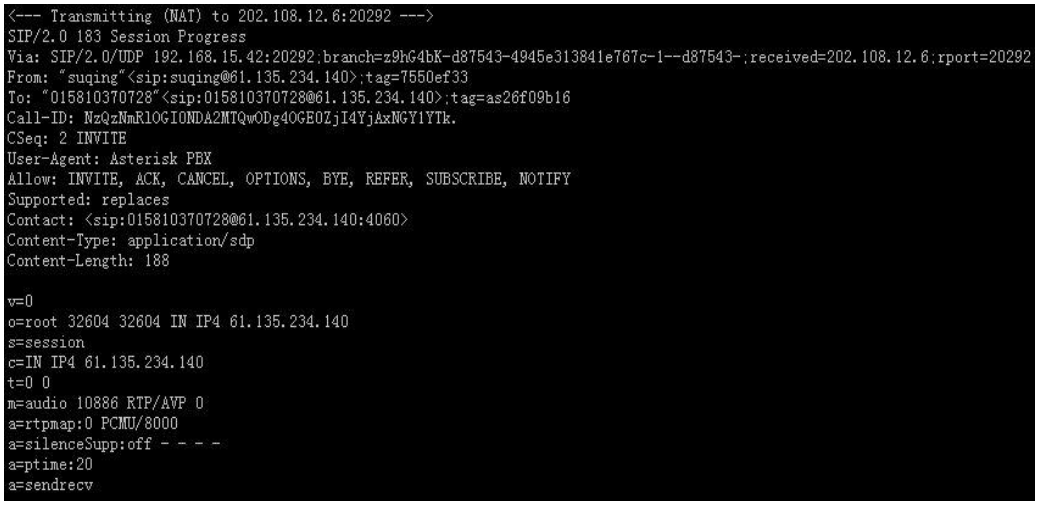
息。



(点击看大图)

图 4-15 SIP_Ringing

Asterisk 响应会话继续的 SIP 消息



(点击看大图)

图 4-16 SIP_183_SESSION_PROGRESS

Asterisk 与华为 8010 语音网关的 H323 连接已经成功建立

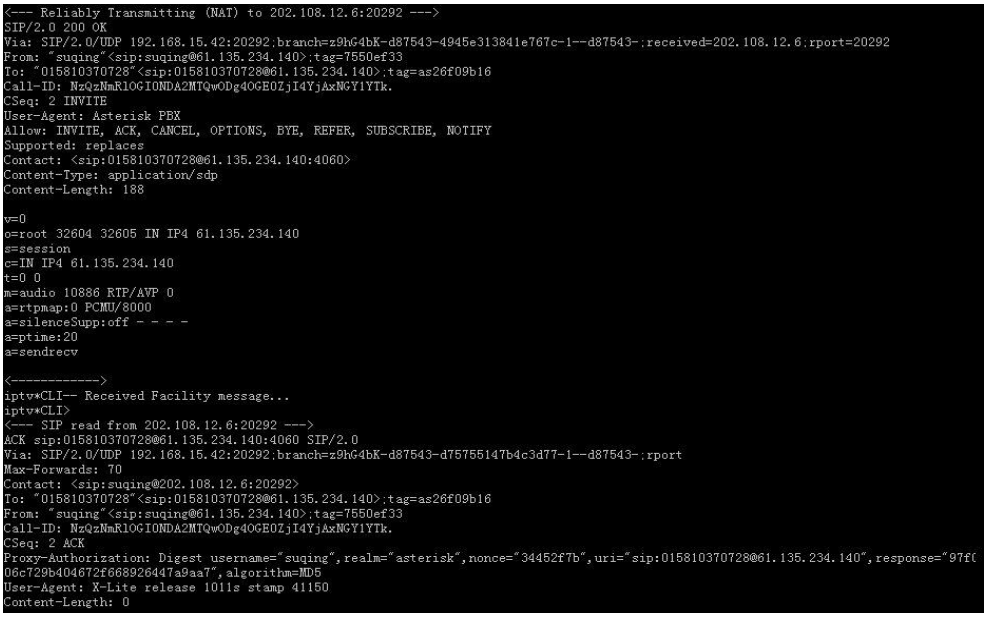
③Connect

表示软交换 Asterisk 与华为 8010 语音网关的 H323 连接已经成功建立

(点击看大图)

图 4-17 H323_ESTABLISHED

Asterisk 与 X-Lite 之间发送 OK 与 ACK 响应消息，表示 SIP 软终端已经跟 H323 语音网关建立了连接，并且被叫已经接听，开始成功通话



(点击看大图)

图 4-18 SIP_200ok_ack 被叫先挂断，Asterisk 与华为 8010 语音网关的 H323 连接在此释放

④Release Complete

表示软交换 Asterisk 与华为 8010 语音网关的 H323 连接在此释放

(点击看大图)

图 4-19 H323_RELEASE Asterisk 向软终端 X-Lite 发送 BYE 消息

图 4-20 SIP_BYE X-Lite 收到 BYE 消息后以 OK 消息响应，整个会话就此终止

图 4-21 SIP_BYE_OK

4.5.2 结论

通过观察上面这个流程图可以容易的看出,语音网关在收到 INVITE 消息后立即发送 SETUP 消息,反之亦然.

所以可以得到如下的 H.323 和 SIP 的消息对应关系:

H.323 消息	SIP 消息
Setup	Invite
Call Proceeding	100 Trying
Alerting	180 Ringing
Connect	200 OK
Release Complete	BYE

基于 Asterisk 的 VoIP 开发指南——(2)Asterisk AGI 程序编写指南

2008/06/12

1.概述

很多时候，我们需要在拨号方案中做某些业务逻辑的判断或者外部数据库的查询，根据具体地需要，有几种做法：

使用 Asterisk 的通道变量、Goto 函数、Gotoif 函数等实现某些简单跳转，通过几个这样的函数的组合，实现简单的业务。

对终端接入用户的呼叫请求中的某些属性，进行简单的数据库增删改查，在 Asterisk 官方发布的 asterisk-addons 开发包中安装 MYSQL 模块，具体地方法在这不细述。使用类似下面的方式：

- exten => _0[0-9].,1, MYSQL(Connect connid dhhost dbuser dbpass dbname)

- exten => _0[0-9].,2., MYSQL(Query resultid ${connid} query-string)

- exten => _0[0-9].,3，MYSQL(Fetch fetchid ${resultid} var1\ var2\ ...\ varN)

- exten => _0[0-9].,4，MYSQL(Disconnect ${connid})

如果碰到了结合了 1、2 的业务需求，这时候拨号方案配置文件中就会出现大量地与业务逻辑有关的复杂代码，造成技术人员阅读上不方便，并且代码也不好维护，如下面这段配置文件：

……

- exten => 888,1,MYSQL(Connect connid localhost ipcontact passwd ipcontact)

- exten => 888,n,GotoIf($["${connid}" = ""]?error,1)

- exten => 888,n,MYSQL(Query resultid ${connid} SELECT\ `number`\ FROM\ `phones`\ WHERE\ `channel`=\'${chan}\')

……

- exten => 888,n,GotoIf($["${foundRow}" = "1"]?done) ; leave loop if no row found

- exten => 888,n,NoOp(${number})

- exten => 888,n,Goto(fetchrow) ; continue loop if row found

- exten => 888,n(done),MYSQL(Clear ${resultid})

- exten => 888,n,MYSQL(Disconnect ${connid})

……

上面综合了通道变量、Goto、Gotoif、MYSQL 数据查询等操作，其实如果配置文件中只是几行这样的操作阅读起来没问题，但业务需求具有可变性，代码维护起来就比较麻烦。所以，我们就寻找另外一种方案，即在配置文件中不进行数据库查

询等重量级的操作，而是将它交到一个外部应用程序或者脚本中，即封装到 Asterisk 的 AGI 后台中，可以实现各种复杂的业务需求。使用类似 PHP、JAVA、C 等编程语言内置的判断语句，而不是 Asterisk 封装的类似 Goto、Gotoif 等函数，会更加快地实现业务需求，代码也更好维护。下面先看看如何在拨号方案中使用 AGI 程序。

2.使用 AGI 脚本

执行 AGI 脚本时，Application 应用就是'agi'，参数是脚本的文件名，同时脚本需要满足以下条件：

- 必须可执行，chomd 755

- 必须放置在指定目录，如标准目录：/var/lib/asterisk/agi-bin

- 必须指定完整的 extension 信息

比如说执行一个用 PHP 语言实现的 AGI 程序，有可能就是这样：

exten => 1,2,AGI(test.php, ${CALLERID(name)})

脚本执行时，可以从控制台上得到不同基本的详细信息，例如，文件不能被执行，或者找不到文件等等。

通过向 Asterisk 控制台信息输出，可以得到 AGI 脚本的执行信息，至少在开发初期中，控制台信息输出是个好办法。

通过 agi VERBOSE 命令，可以将信息发送到 asterisk 控制台上，并且而通过 verbosity 设置可以关闭开启这个功能。

如果直接，可以采用各种语言很方便的通过 AGI 接口编写实施脚本。脚本和 Asterisk 之间通过标准的输入输出进行交互。

3.AGI(Asterisk Gateway Interface)技术实现原理

- 传递参数到 AGI 脚本

在脚本名称后紧跟以英文半角状态下的逗号,分隔的字符串，就可以把需要的参数传入脚本：AGI(dial_agi.php,${EXTEN:11},${CALLERID(name)})，AGI 脚本总是接收 2 种参数，1 是脚本的完整路径，2 是拨号方案中'Exten'传递的参数，其中第 1 种参数，如果 AGI 程序放在了 Asterisk 默认路径，可以省略，只写 AGI 程序名。第 2 种参数是 AGI 程序需要拨号方案中'Exten'传递进来的参数，比如说本文中 ${EXTEN:11},${CALLERID(name)}，一个是被叫的电话号码，一个是主叫的账号 ID。

- 通过标准的输出，发送命令到 Asterisk

我们可以在 AGI 脚本程序中向 Asterisk 发送各种命令从而调用 Asterisk 的某些应用程序，如 Dial、Goto、Monitor 等，也可以直接发送命令获得或者设置某些 Asterisk 通道变量的值。对于基于 PHP 的 AGI 脚本程序，可以按照如下步骤：

(1). 打开 PHP 输出文件描述符：

$stdout = fopen('php://stdout', 'w');

(2). 向 Asterisk 发送命令：

fputs($stdout," SET CONTEXT media_gw1\n"); ");
fflush($stdout);

上述步骤是对于 SET 或者 GET Asterisk 的某些通道变量时的使用方法，如果需要调用 Asterisk 内置的应用，如执行跳转到某个 context 下的某个 priority 的 Goto 应用函数，可以调用 EXEC 命令后面紧跟 Asterisk，如：fputs($stdout," EXEC Goto media_gw1|s|2\n"); ");命令必须以换行符结束，AGI 命令返回文本字符串，如下格式：200 Result=，有时会在 number 数字后附加一些信息。如果向 Asterisk 发送了无效的命令，信息如下：510 Invalid or unknown command。对应上面的命令，如下所示：

- AGI Rx << SET CONTEXT media_gw1

- AGI Tx >> 200 result=0

- 通过标准的输入，从 Asterisk 接收信息

当 AGI 脚本执行时，Asterisk 会向脚本发送各种的信息，可以在做其他事情之前通过标准输入获取这些信息，每项数据都是一行，发送完毕 Asterisk 会发送一个空行，表示结束，如：

- AGI Tx >> agi_request: dial_agi.php
- AGI Tx >> agi_channel: SIP/25946-0821ea88
- AGI Tx >> agi_language: en
- AGI Tx >> agi_type: SIP
- AGI Tx >> agi_uniqueid: 1209093478.477
- AGI Tx >> agi_callerid: 0000123456
- AGI Tx >> agi_calleridname: beigaolin
- AGI Tx >> agi_dnid: 998866015810370728
- AGI Tx >> agi_context: default
- AGI Tx >> agi_extension: 998866015810370728
- AGI Tx >> agi_priority: 1

根据项目需求，如果需要这些数据，就先保存起来，否则不用处理它。保存步骤按如下过程。

1．打开 PHP 输出文件描述符：

```
$in = fopen("php://stdin","r");
```

2．分析从 Asterisk 传到 AGI 的头信息，如需要在 AGI 程序中获取终端用户的 ID，那么从“agi_calleridname: beigaolin”这个头信息可以获取，我们通过分析每一行这样以:分隔的字符串，取到需要后续处理的字符串

```
while (!feof($stdin)) {
$temp = fgets($stdin);
$temp = str_replace("\n","",$temp);
$s = explode(":",$temp);
$agivar[$s[0]] = trim($s[1]);
if (($temp == "") || ($temp == "\n")) {
break;
}
}
```

4.使用开源 PHP AGI 类函数 PHPAGI

像上一小节那样先是获取输入流，分析从输入头字符串中获取对应某个输入变量的值，或者获取输出流然后发送各种标准命令执行某些 Asterisk 内置应用，如果在 AGI 程序中实现很复杂的业务逻辑，这样的流程会显得有点累赘，所以需要提取某些常用的操作，我们使用的时候不用关心这些操作，直接以调用类似 Asterisk 内置应用那样的方式。PHPAGI 就是这样的一个开源 PHP 类函数。它封装了对应 Asterisk 内置应用的常用函数调用接口，比如说从 PHP 向 Asterisk 发送 Dial 命令的操作，可以直接调用 PHP AGI 类函数中的 exec_dial。使用 PHP AGI 能够很容易的操作 Asterisk AGI 常用接口。使用这个类函数也很简单：

- 下载准备 phpagi 函数文件
- cd /var/lib/asterisk/agi-bin/(也有可能在用户自定义的路径中)
- wget http://nchc.dl.sourceforge.net/sourceforge/phpagi/phpagi-2.14.tgz
- tar zxvf phpagi-2.14.tgz

- 在代码中使用
- include ("phpagi.php");//包含文件
- include ("phpagi-asmanager.php");
- $agi = new AGI;//引用 PHPAGI 类函数

5.使用 AGI 实现主叫号码透传功能

在这里以一个例子来说明 AGI 程序在 VoIP 开发中的作用以及开发思路。

假设说有个普通电话为 02412345678，手机号为 15810370728，而网络电话虚拟号码是 0000123456，如果想让拨打出去的电话号码在被叫方(手机或者带有来电显示功能的座机)的来电显示为 02412345678 或者 15810370728，那么他们回复电话的时候就可以直接打到这个普通电话上，方便与主叫的业务联系。这个需求就叫主叫号码透传，能不能进行主叫号码的透传，取决于 VoIP 落地网关运营商，语音网关可以设置 IP 侧送过来的主叫号码是否透传。在保证号码规范的前提下，透传什么样的主叫号码，则取决于 IP-PBX 系统，即 Asterisk 的设计了。

- 设计思路

1.增加一个针对终端用户账户 ID 的绑定管理系统，如图用户在第二项中输入自己的账户 ID，然后再输入想要作为来显示的主叫号码完成绑定操作，后台 php 程序向数据库中插入一条新记录(X-Lite ID 对应电话号码或者手机号码)。

图 1AGI 后台管理系统页面

2.使用绑定了主叫号码的 X-Lite 呼叫某个被叫(手机或者座机)

Asterisk 的后台 PHP AGI 程序的详细设计主叫号码透传流程设计如图 2 所示。

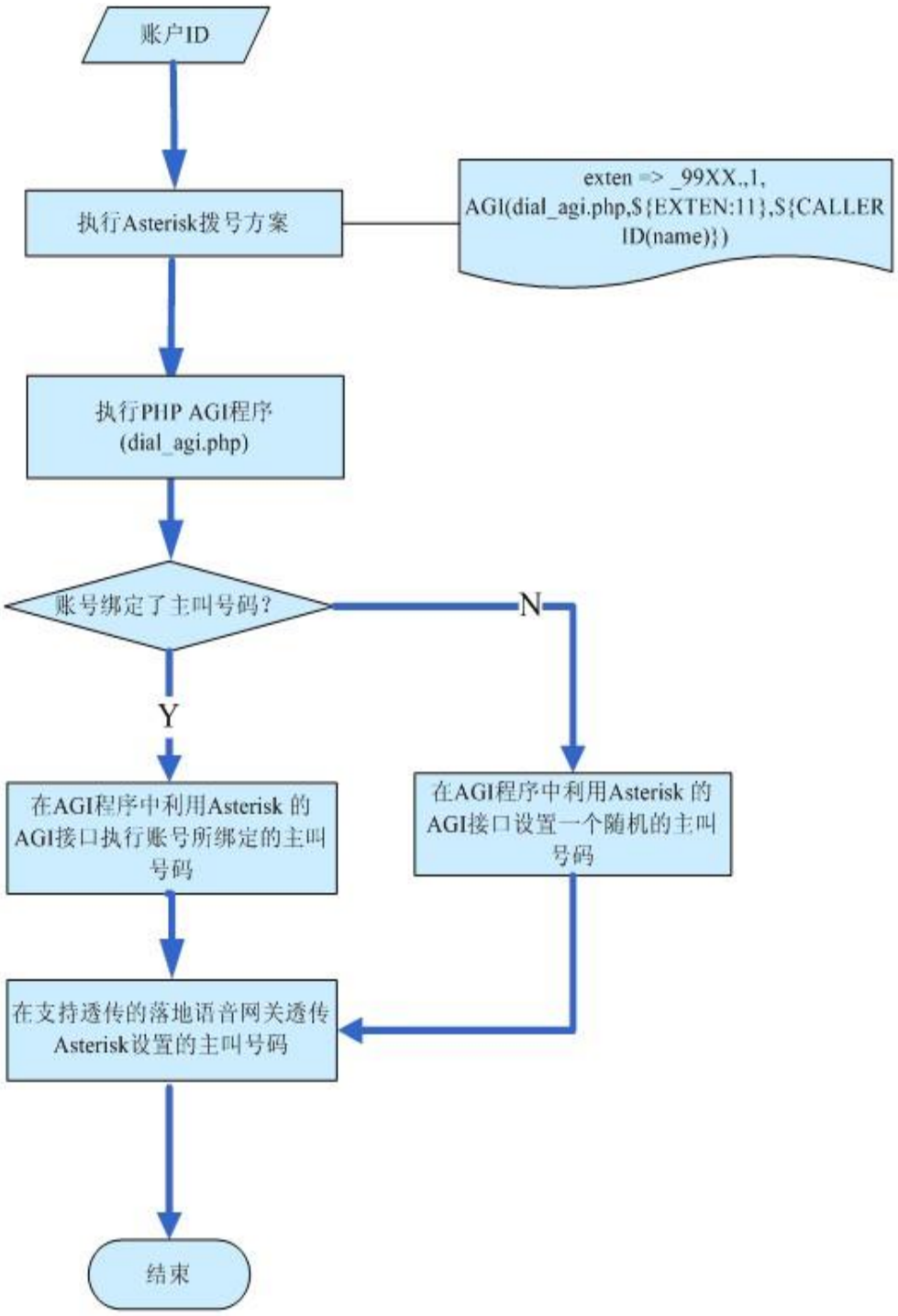


图 2Asterisk 主叫号码透传的后台 PHP AGI 流程图

- 代码实现

以下代码片断展示的是 PHP AGI 中部分代码，并且作了简化。

```
#!/usr/local/php.5.2.5/bin/php –q
include_once("phpagi.php");//开源 PHP 类函数
```

```
.....
//判断当前这个 id 是否做了主叫号码来电显示的绑定操作
$query_string = "select * from xliteid where xliteid = '{$caller_name}'";
$query_result = mysql_query($query_string, $db_connection);

//如果当前这个 id 做了绑定操作，调用 PHPAGI 类函数，设置 Asterisk 主叫号
码

if($query_result && mysql_num_rows($query_result) > 0)
{
caller_phone_display_agi ();
}
//没有做绑定，设置一个随机的号码
else
{
caller_name = $argv[2];
$rand_num1 = rand(0,9);
$rand_num2 = rand(0,9);
$rand_num3 = rand(0,9);
$caller_phone=
"024{$rand_num1}{$rand_num2}650{$rand_num3}{$rand_num4}";
land_media_gw1($caller_phone);
exit();
}
/**
*@caller_phone_display_agi 主叫号码特殊显示
*/
function caller_phone_display_agi()
{
global $db_connection, $callee_phone, $caller_name;
$query_string = "select caller_phone from caller_phone_display _xliteid where
skype_id = '{$caller_name}'";
$query_result = mysql_query($query_string, $db_connection);
{
$row = mysql_fetch_array($query_result);
$caller_phone = $row[0];
$callerid_cli = "\"{$caller_name}\"<{$caller_phone}>";
land_media_gw1($callerid_cli);
exit();
}
```

```
}
/**
*@ land_media_gw1  VoIP 语音网关 media_gw1
*/
function  land_media_gw1($callerid_num)
{
global  $agi, $callee_phone_withpre;
$agi->set_context("media_gw1");
$agi->set_extension($callee_phone_withpre);
$agi->set_priority(1);
//调用 phpagi 封装的 set_callerid 方法，向 Asterisk 传递设置主叫号码的指令
$agi->set_callerid($callerid_num);
}
```

对 X-Lite 账户 gaolinb 作了主叫号码绑定，使用 X-Lite 软终端呼叫普通的手机，在 Asterisk 中设置了 agi debug，从 Asterisk 后台我们可以清晰地看到：

1．AGI Tx >> *CLI>上面部分，全是从 Asterisk 输入到当前 AGI 的环境变量信息，它包含了当前这个呼叫的详细信息，如 Channel 的类型，是 SIP 还是 H.323，calleridname，即终端用户是 gaolinb 等重要信息。

2．AGI Tx >> *CLI>下面部分，全是在上面调用 PHPAGI 类函数后将命令传给了 AGI 程序执行，对于主叫号码来电显示的命令是：

SET CALLERID ‘gaolinb’<15810370728>，Asterisk 将 15810370728 传到能够支持主叫号码透传的 VoIP 运营商，从而被叫用户在接听电话前能够显示一个有意义的电话号码。

图 3 Asterisk 服务器上 AGI 的输入输出信息

基于 Asterisk 的 VoIP 开发指南——Asterisk 模块编写指南

2008/06/12

1.开源项目概述

Asterisk 是一个开源的软件包，通常运行在 Linux 操作系统平台上。Asterisk 可以用三种协议来实现 VoIP，同时可以与目前电话使用的标准硬件进行交互通信，Asterisk 在实现 VoIP 时，不需要任何附加硬件，本文所采用的也是这种使用方式。但是，如果企业没有与 VoIP 语音网关运营商建立合作关系，想要自己构建这样的一个平台，那么要和数字电话设备与模拟电话设备进行交互通信，Asterisk 需要一个 PCI 硬件的支持，这个硬件生产商中最著名的是 Digium 平台提供的。

Asterisk 的结构基本上是十分简单，但是它不同于大多数的电话产品。基本上，Asterisk 担任的是一个中间件的功能，它连接了底层的电话技术和上层的电话应用。所以，Asterisk 具有很大的柔韧性，特殊的 API 接口都围绕着 PBX 核心系统。这个核心处理着 PBX 内部之间的相互联系。每一部分都是清晰来自于协议、编码或内部电话使用的硬件接口的抽象。这些抽象的接口使 Asterisk 可以与任何的硬件和技术以及将来的硬件和软件技术完美的结合。从图 2.5 可以看出，Asterisk 由内部核心和

外围动态可加载模块组成。内部核心由以下六个部分组成：PBX 交换核心模块(PBX Switching Core)、调度和 I/O 管理模块(Scheduler and I/O Manager)、应用调用模块(Application Launcher)、编解码转换模块(Codec Translator)、动态模块加载器模块(Dynamic Module Loader)和 CDR 生成模块(CDR Core) 。

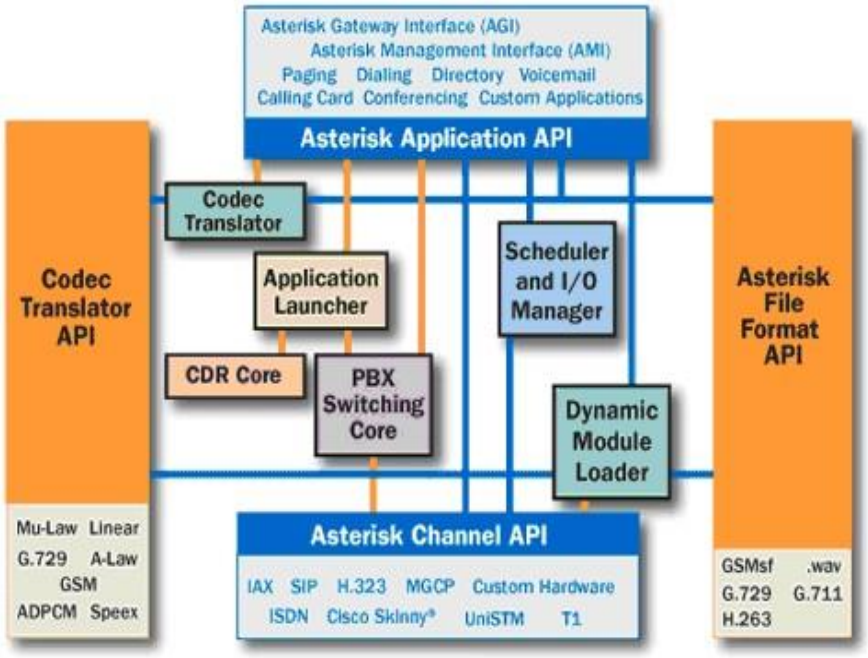


图 1 Asterisk 结构图

2.Asterisk 二次开发概述

Asterisk 是一个开源的 PBX 架构；但它并不是一个成品。通常情况下，由于企业应用的多样性，很难有一个成型的 PBX 产品可以满足企业的各种需求。传统的 PBX 成品，要么功能和灵活性不足，要么配置和维护复杂；而且都具有一个致命的缺点，那就是开放性、可扩展性。

Asterisk 具有传统 PBX 无法比拟的优点，那就是其灵活性，可扩展能力；Asterisk 的扩展能力是通过开放相应的架构和接口来实现的。这就意味着 Asterisk 是一个组件而不是一个成型的产品，Asterisk 的核心提供了一个基本的可运行环境，而外围相应的能力则可以通过加载和配置相关的插件和模块来实现。

Asterisk 是一个开源的 PBX 架构；但它并不是一个成品。Asterisk 的扩展能力是通过开放相应的架构和接口来实现的。这就意味着 Asterisk 是一个组件而不是一个成型的产品，Asterisk 的核心提供了一个基本的可运行环境，而外围相应的能力则可以通过加载和配置相关的插件和模块来实现。

因此，使用 Asterisk，一定会面临二次开发问题，这些二次开发主要围绕以下几个方面：

（1）内部核心模块

- 开发扩展编解码能力模块

- 开发扩展相应的通道模块

（2）外围动态可加载模块

- 开发应用部分

- 开发外围管理部分

一般来说，Asterisk 使用者很少需要去开发编解码能力模块和通道模块等内部核心模块；而需要开发最多的情况则是外围动态可加载模块，即外围管理部分和应用开发，本文也是指这些方面的开发。

3.Asterisk 通道模型与呼叫流程

3.1 什么是 asterisk 通道？

Asterisk 通道是指通过 asterisk 建立起来的一路通话。这类通话都包含一个 incoming 连接和一个 outbound 连接。每个电话都是通过一种通道驱动程序建立起来的，比如 SIP,ZAP,IAX2 等等。每一类的通道驱动，都拥有自己私有的通道数据结构，这些私有的结构从属于一个通用的 Asterisk 通道数据结构中，具体定义在 channel.h 和 channel.c 中。

3.2 基本的呼叫流程

Asterisk PBX 呼叫流程如图 2 所示。

- 通过 Asterisk 的一个电话呼叫在一个通道驱动接口上到达，如 SIP Socket。

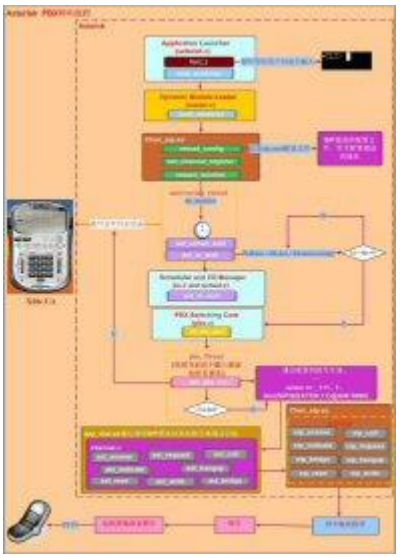
- 通道驱动在该通道上创建一个 PBX 通道并启动一个 pbx 线程

- 拨号方案被执行，拨号方案在一些地方通过 dial 应用(查看 app_dial.c)

强制 Asterisk 创建一个呼出呼叫，一旦呼出，Asterisk 会有以下两个动作将发生。

- Dial 创建一个呼出的 PBX 通道并请求一种通道驱动创建一个呼叫。

- 当呼叫被应答时，Asterisk 桥接媒体流，于是在第一个通道上的主叫可以和在第二个通道也就是呼出通道上的被叫通话。



(点击放大）

图 2 Asterisk PBX 呼叫流程

4.RADIUS 协议的概述

（1）Radius 协议在协议栈中的位置

Radius 是一种流行的 AAA 协议，同时其采用的是 UDP 协议传输模式，AAA 协议在协议栈中位置如图 3 所示。

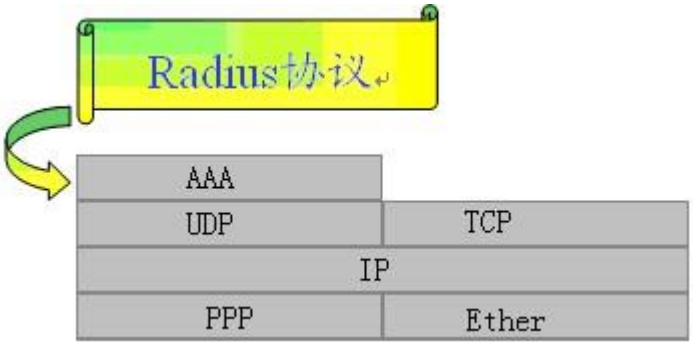


图 3 Radius 协议在协议栈中的位置

（2）Radius 协议选择 UDP 作为传输层协议

- NAS 和 Radius 服务器之间传递的是几十上百个字节长度的数据，且 Radius 要求特别的定时器管理机制，用户可以容忍几十秒的验证等待时间。

- 当处理大量用户,服务器端采用多线程，UDP 简化了服务器端的实现过程。

- TCP 是必须成功建立连接后才能进行数据传输的，这种方式在有大量用户使用的情况下实时性不好。Radius 要有重传机制和备用服务器机制，它所采用的定时，TCP 不能很好的满足。由于数据包可能会在网络上丢失，如果客户没有收到响应，那么可以重新发送该请求包。多次发送之后如果仍然收不到响应，RADIUS 客户可以向备用的 RADIUS 服务器发送请求包。

- Radius 依靠自身协议保证报文重传和服务器备份机制以确保计费可靠性。

5.认证计费功能概述

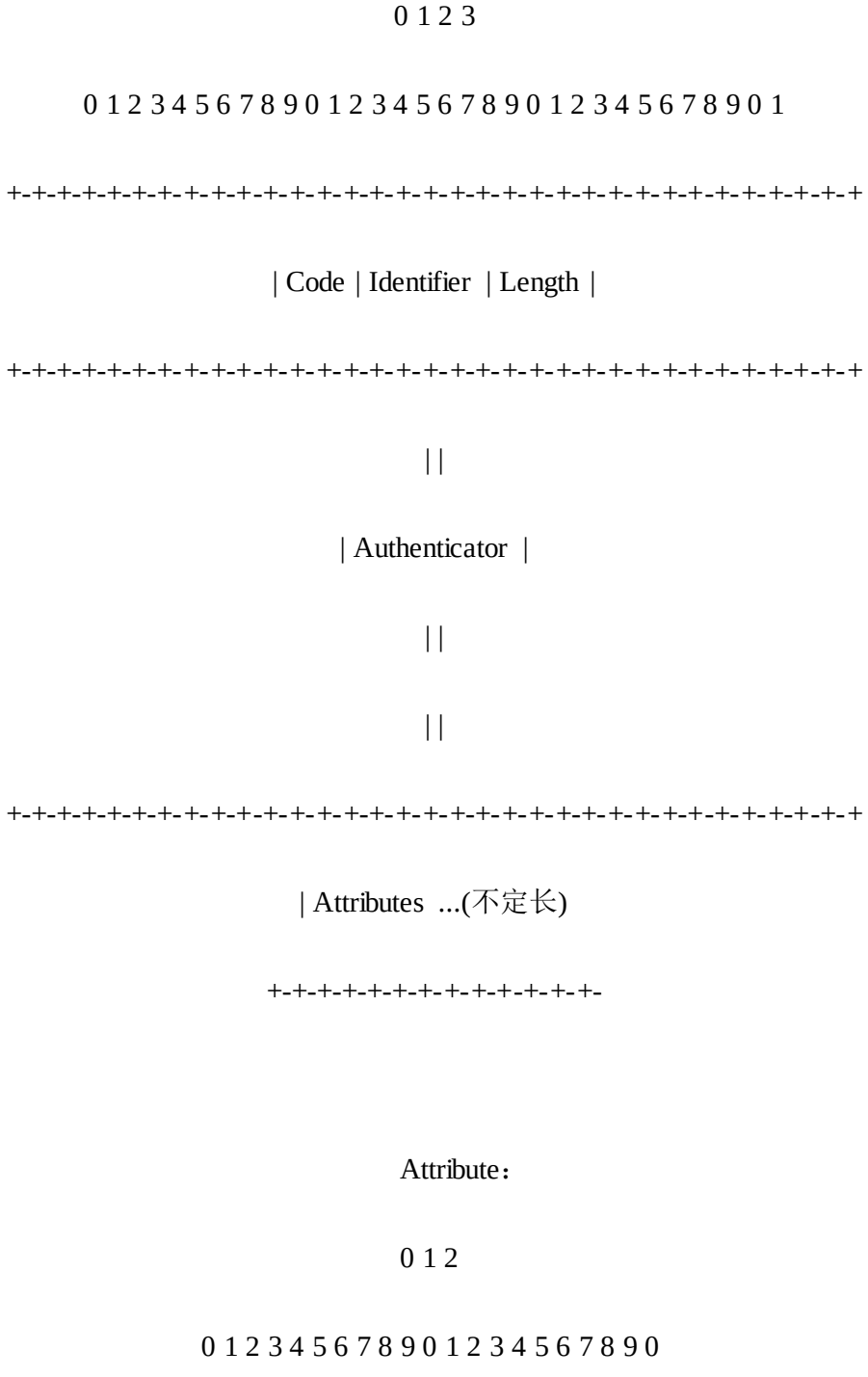
IP-PBX 呼叫控制功能，主要是 VoIP 终端用户的认证计费控制过程，是 VoIP 系统商业化运营的核心模块。

Radius Client 端，也叫 NAS，主要的任务就是根据 VoIP 终端的呼叫请求携带的各种属性，包括账户 ID、被叫号码、通话时间等，封装成标准的 Radius 数据包发送到 Radius Server 端，达到账户信息实时更新的效果。整个 NAS 端程序主要由两个模块构成：认证模块和计费模块，并把这两个模块整合到开源 IP-PBX 项目 Asterisk 中。

5.1 标准 RADIUS 协议分析

（1）Radius Packet

RADIUS 数据包被包装在 UDP 数据报的数据块（Data field)）中，其中的目的端口为 1812，RADIUS 协议包结构如图 4 所示。



+-+

| Type | Length | Value …(不定长)

+-+

图 4 RADIUS 协议包结构图

（2）对 Radius Packet 格式各个域解释

Code：包类型，一个字节长，指示 RADIUS 包的类型，包含不合法的 Code 的 Radius 包将被直接丢弃，code 域主要包含了以下值类型。

1）code=1 Access-Request——认证请求数据包

本文 AAA 功能就是构建 code＝1 的认证请求数据包。

2）code=2 Access-Accept——认证响应数据包

3) code=3 Access-Reject——认证拒绝数据包

4）code=4 Accounting-Request——计费请求数据包

本文 Asterisk 的 AAA 功能另外一个重点任务就是构建 code=4 的计费请求数据包，Accounting-Request 数据包中的两种状态类型（Acct-Status-Type）的计费请求数据包：Start（Value=1）：Client 开始对指定用户提供服务，计费开始;Stop（Value=2）：Client 停止对指定用户提供服务，计费结束。

5）code=5 Accounting-Response——计费响应数据包

因为是要更新账户信息，所以目前本文不需要处理计费响应数据包。

Identifier：包标识符，一个字节长，用于匹配请求包和响应包，同一组请求包和响应包的 Identifier 应相同。协议规定：

1) 在任何时间，发给同一个 RADIUS 服务器的不同包的 Identifier 域不能相同，如果出现相同的情况，RADIUS 将认为后一个包是前一个包的拷贝而不对其进行处理。

2) Radius 针对某个请求包的响应包应与该请求包在 Identifier 上相匹配（相同）。

Length：包长度，两个字节长，说明数据包的长度，是 code、identifier、length、authenticator attribute fields 的长度总和，有效范围是 20~4096，超出范围的数据将被视为附加数据（Padding）或直接被忽略。

Authenticator：验证字，16 字节长，用于验证消息的负载,对包进行签名，该验证字分为两种。

1) 请求验证字---Request Authenticator,用在请求报文中,必须为全局唯一的随机值。

2) 响应验证字---Response Authenticator,用在响应报文中，用于鉴别响应报文的合法性。响应验证字＝MD5(Code+ID+Length+请求验证字+Attributes+Key)。

Attributes: Type 指示了 Atribute 的类型，通用的有几十种，在系统中使用到的，如表 4.1 所示。Asterisk AAA 模块的构建主要是构建表 1 列出的这些属性值的 RADIUS 数据包。

表 1 Atribute 的属性列表

属性值	属性名称	属性意义
1	User-Name	用户账户 ID
2	User-Password	用户密码
4	Nas-IP-Address	Nas 的 ip 地址
5	Nas-Port	用户接入端口号
6	Service-Type	服务类型
7	Framed-Protocol	协议类型
8	Framed-IP-Address	为用户提供的 IP 地址
11	Filter-Id	过滤表的名称
27	Session-Timeout	通知 NAS 该用户可用的会话时长（时长预付费）
32	NAS-Identifier	标识 NAS 的字符串
40	Acct-Status-Type	计费请求报文的类型
41	Acct-Delay-Time	Radius 客户端发送计费报文耗费的时间
44	Acct-Session-Id	计费会话标识

45	Acct-Authentic	在计费包中标识用户认证通过的方式
46	Acct-Session-Time	通话时长(用户在线时长）
49	Acct-Terminate-Case	用户下线原因

5.2 选择一个合适的 Radius Client API

上个小节介绍的 RADIUS 数据包格式，是构建应用协议层数据包的封装所关注的，在 Asterisk 中如果需要亲自把标准 RADIUS 数据包的发送、接收等过程从零开始写起，那本文就把重点放在了 RADIUS UDP 数据包与服务器通信过程的编写中了，实际本文关注的是在 Asterisk 中根据 VoIP 通信中的业务需求，构建 RADIUS 认证计费模块，重点是业务应用层的开发，即如何组织认证包、计费包的数据结构等，而 RADIUS 数据包传输层直接调用现成的开源 API，目前主要有两种这样的开源项目。

（1）pam_radius

一个 PAM 模块提供了 RADIUS 客户端的功能。它是从开源项目 Freeradius 中提取出来的，如果要使用需要对代码做大量的修改、打补丁后才能使用。

（2）radiusclient-ng

相对比 PAM 的 pam_radius 模块而言，radiusclient-ng 的动态库代码不用修改就可以拿过来使用，只需安装 radiusclient-ng 的动态库，然后根据配置文件、开放的 API 接口修改 Asterisk 代码就可以完成 Asterisk AAA 模块的构建。

所以在本文使用 radiusclient-ng 开源软件包。