基于嵌入式的 Internet 远程监控系统①

Internet Remote Monitor Based on Embedded System

张华林　黄以平　(桂林电子科技大学 机电工程学院 广西 桂林 541004)

摘　要： 本文研究借鉴 IT 领域中常用的远程监控实现方式，从实际应用角度出发提出一种实时、可靠的嵌入式 Internet 远程监控方案。此方案首先利用 OpenVPN 建立 PC 和嵌入式端的隧道链接，然后在嵌入式平台启用 telnetd 和 sshd 服务，最后在嵌入式平台搭建 WebServer 服务器。PC 端通过 VPN 隧道，可以用 telnet 或 ssh 对嵌入式平台远程登陆，或用浏览器对嵌入式端进行监控。该方案主要从软件实现思路方面考虑，和硬件平台基本无关，经测试稳定可靠。

关键词： 嵌入式 Internet 远程监控 OpenVPN WebServer

1　引言

随着现代工业设备智能化的不断提高和网络技术的不断发展，企业希望将分布式生产设备或科研测试仪器与网络相连，使工作人员能在办公室内对不同生产现场的设备仪器等进行远程监测，并进而实现生产现场数据和管理系统数据集成共享[1]。远程网络监控可以为企业减少工作流程中的环节，节省人力物力资源，是工业控制领域发展的必然趋势。

网络监控根据距离的不同大致分为局域网监控和 Internet 监控。相对而言局域网具有较高得稳定性，局域网范围的监控技术已经非常成熟和完善，现已广泛应用于实际生产。而实际的 Internet 远程监控，则受到很多外部因素的制约：网络接入方式的不统一，企业内部路由限制，ISP 限制，网络延迟过大、掉包过多、不稳定等。正是因为这些因素，使得很多可以运行于局域网的远程控制方式不能稳定应用于 Internet 平台。针对上述情况，本文设计一种可用于 Internet 的远程控制监控方案：应用 VPN 技术，在嵌入式和远端 PC 之间建立稳定加密的隧道通信；在嵌入式端启用 telnetd 服务和 sshd 服务，使远端开发人员可以远程登陆，完全控制嵌入式系统；在嵌入式上使用 B/S 瘦客户端方式的 WebServer 技术，使远端使用人员可以很好地监控远端嵌入式系统。这个远程监控平台实现原理如图 1 所示。

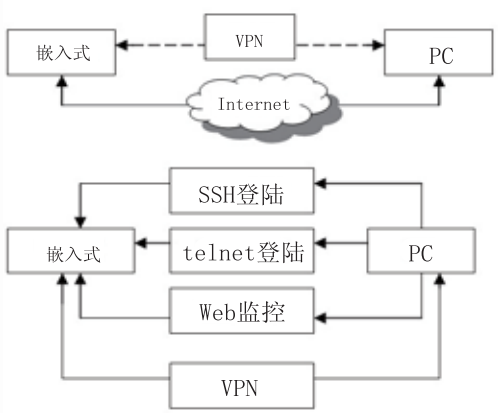

图 1　平台实现原理

2　嵌入式远程监控系统的构建

2.1 VPN 技术在嵌入式平台上的应用

无论任何平台的远程监控，首先要解决的问题是保证监控两端要有稳定互通的连接机制。当监控两端为 Internet 上的两个节点，若要建立连接就必须考虑它们之间的网络结构，这样在不同地点部署监控就要对其网络独立设计或配置，比较的繁琐。本文针对此问题，应用 VPN 技术，使嵌入式平台通过 VPN 连接到控制平台，建立直接通信机制，解决监控两端之间的网路连接问题，为后续工作提供必要前提。

虚拟专用网 VPN(virtual private network)技术

① 收稿时间:2008-12-14

是指在公共网络上建立专用网络的技术[2]。网络的任意两个结点之间的连接并没有传统专用网所需的点到点的物理链路，而是架构在公用网络服务商 ISP 提供的网络平台之上的逻辑网络。用户的数据通过 ISP 在公共网络中建立逻辑隧道(即点到点的虚拟专线)进行传输,并通过相应的加密和认证技术来保证用户内部网络数据在公网上的安全传输，从而真正实现网络数据传输的专有性,并使安全性得以保证[3]。

OpenVPN 是一种具备完全特征的 SSLVPN 解决方案，是一种实现在用户空间，但依然是工作在网络二三层的 VPN 技术，因此它能够支持任意一种 IP 应用。OpenVPN 通过使用工业标准 SSL/TLS 协议实现了 OSI 二层及三层安全网络扩展，支持灵活的基于证书、智能卡的客户端认证方法。Openvpn 使用 TUN/TAP 驱动程序将二三层的数据包传送到用户空间，然后使用应用层 SSK/TLS 技术加密传输，从而实现隧道功能。TUN/TAP 驱动程序包含两个部分，一部分是字符设备驱动，另一部分是网卡驱动。网卡驱动可以用来接收来自 TCP/IP 协议栈的网络包并发送出去，或者将收到的网络包传输给协议栈处理，而字符驱动可以实现网络包在内核与用户空间之间的传送，模拟物理链路的数据接受和发送。TUN/TAP 原理如图 2 所示。

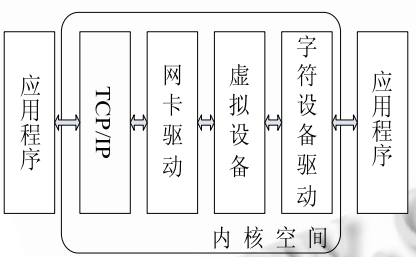

图 2　TUN/TAP 实现原理

OpenVPN 为一种加密 vpn 技术,加密和解密双方必须预先知道加密的 Key, OpenVPN 使用 TLS 加密是通过使用公开密钥(非对称密钥，加密解密使用不同的 key，一个称为 Public key，另一个是 Private key)对数据进行加密的，对于 OpenVPN 使用 TLS mode，首先 Server 和 Client 要有相同 CA 签发的证书，双方通过交换证书验证双方的合法性以决定是否建立 VPN 连接，然后使用对方 CA 把自己目前使用的数据加密方法(类似于密钥)加密后发送给对方，由于使用对方 CA 加密的 key，所以只有对方 CA 对应的 Private key 才能解密该字串，保证了此密钥的安全性。

在嵌入式平台上移植 OpenVPN，需要 2.6 内核的 TUN/TAP 设备支持,同时需要 OpenSSL 库的支持。移植步骤如下:

(1) 内核选项 Universal TUN/TAP device driver support 必须选择以支持 TUN/TAP 设备。

(2) 交叉编译 OpenSSL,OpenSSL 整个软件包大概可以分成三个主要的功能部分:密码算法库、SSL 协议库以及应用程序，编译 OpenSSL 主要是为了下一步编译 OpenVPN 提供密码算法库。

(3) 交叉编译 OpenVPN,需要包含上面编译出来的 OpenSSL 库。然后配置 OpenVPN 脚本 Client.conf，填入证书路径以及服务器信息。把 OpenVPN 服务器端生成的证书拷贝到嵌入式平台中。

(4) 启动 OpenVPN 客户端进程，嵌入式平台就可以和远端 PC 建立虚拟网络连接。

2.2 TELNET 和 SSH 在嵌入式平台上的应用

当监控和被监控端连接建立以后，开发人员需要完全控制被控端系统，执行特定软件或者修改系统状态。对于 linux 操作系统，telnet 和 ssh 是两种最常见的远程登陆的实现方式，telnet 为不加密连接，ssh 为加密连接。

(1) TELNET 的在嵌入式平台上的应用

远程登录(RemoteLogin)是 Internet 上最广泛的应用之一。用户可以先登录到一台主机然后再通过网络远程登录到任何其他一台网络主机上去，不需要为每一台主机连接一个硬件终端。在 TCP/IP 网络上，telnet 是标准的远程登录应用，几乎每个 TCP/IP 的实现都提供这个功能。它能够运行在不同操作系统的主机之间。telnet 通过客户进程和服务器进程之间的选项协商机制，从而确定通信双方是否进行通信。一般嵌入式平台都提供 telnetd 服务以方便调试。

Telnet 实现原理如图 3 所示，首先客户端发送请求，服务器端 telnetd 接收到远程登录请求后，将其作为仿真终端(伪终端),派生出子进程 Pseudo 与客户端 telnet 进程交互，用户输入用户名和口令，进行远程登录。如果登录成功，用户在键盘上输入的每一个字符都传到远程主机服务器上，用户输入主机终端命令，Pseudo 进程接收命令，将用户输入的命令传给

操作系统进行处理，并将处理结果传给用户进程 telnet，用户进程将结果显示在屏幕上。

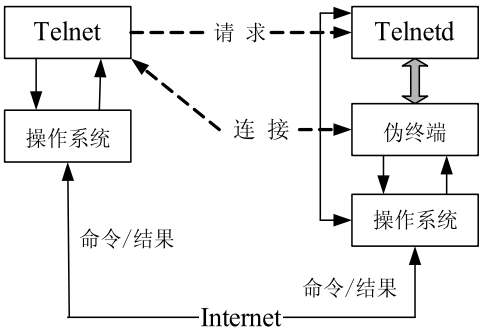

图 3 TELNET 实现原理

要在嵌入式平台上移植 telnetd，首先在嵌入式平台上建立/dev/ptmx 设备文件，mknod -m 666 dev/ptmxc52；其次嵌入式平台的/etc/inted.conf 文件里面必须告诉操作系统 telnetd 的位置，权限和工作模式，然后 devpts 必须和/dev/pts 关联起来，/dev/pts 为程序运行以后伪终端位置，mount-t devpts-o gid=5,mode=620 /dev/pts /dev/pts；最后启动 telnetd 服务，这样其他 PC 就可以通过 telnet 连接到嵌入式平台。

(2) OpenSSH 的加密连接在嵌入式平台上的应用

SSH(secure shell)是一种通用，功能强大的网络安全解决方案，计算机每次向网络发送数据时，SSH 都会自动对其进行加密。数据到达目的地时，SSH 自动对加密数据进行解密，整个过程都是透明的。

SSH 协议框架中最主要的部分是三个协议：传输层协议、用户认证协议和连接协议。同时 SSH 协议框架中还为许多高层的网络安全应用协议提供扩展的支持。它们之间的层次关系如图 4 表示。

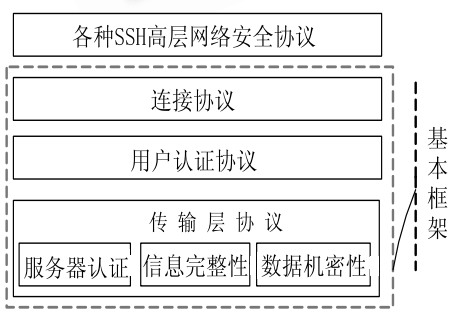

图 4 SSH 协议框架

在 SSH 的协议框架中，传输层协议(The Transport Layer Protocol)提供服务器认证，数据机密性，信息完整性等的支持；用户认证协议(The User Authentication Protocol)则为服务器提供客户端的身份鉴别；连接协议(The Connection Protocol)将加密的信息隧道复用成若干个逻辑通道，提供给更高层的应用协议使用；各种高层应用协议可以相对地独立于 SSH 基本体系之外，并依靠这个基本框架，通过连接协议使用 SSH 的安全机制。使用 SSH 的主机都必须有自己的主机密钥，每一对主机密钥对包括公开密钥和私有密钥。主机将自己的公用密钥分发给相关的客户机，客户机在访问主机时则使用该主机的公开密钥来加密数据，主机则使用自己的私有密钥来解密数据，从而实现主机密钥认证，确定客户机的可靠身份。

在嵌入式平台上移植 OpenSSH 需要 OpenSSL 加密库的支持，首先要交叉编译 OpenSSL；再交叉编译 OpenSSH，然后把所有编译后的文件拷贝到嵌入式平台，在平台上生成密匙；接着配置好 OpenSSH 的配置文件 sshd_config,填入服务器地址、运行模式、加密压缩选项等；最后启动 sshd 服务进程。这样其他 PC 就可以通过 ssh 连接到嵌入式平台。

2.3 基于 WebServer 的远程监控平台

B/S(Browser/Server 客户机和服务器结构)和 C/S (Client/Server 浏览器和服务器结构)是当前嵌入式监控中常用的两种软件系统体系构架。B/S 结构软件设计简单，维护方便，升级容易，成本低，无平台局限，是以后的发展方向；C/S 结构软件专业性强，实时性好，但设备维护复杂，升级不便。对于嵌入式监控领域，两种平台各有优势，对于简单使用，可以考虑 B/S 构架，若要实时要求高，如故障诊断，可采用 C/S 构架。本文主要实现的是基于 B/S 构架的 boa 服务器的使用。

嵌入式设备中使用 HTTPD 服务，可以向 Internet 或者局域网提供基于 Web 的图形化管理接口。使用 CGI(通用网关接口)技术使得浏览器和服务器之间具有交互性[4]，WebServer 和 CGI 的结合使用，是当前嵌入式远程监控中最常用和最成熟的手段。嵌入式 Web 服务器是实现嵌入式系统通过 Web 方式与 Internet 互联的关键组成部分，而 CGI 是用户与 Web 服务器交互的一个重要途径[5]。其工作原理如图 5 所示。

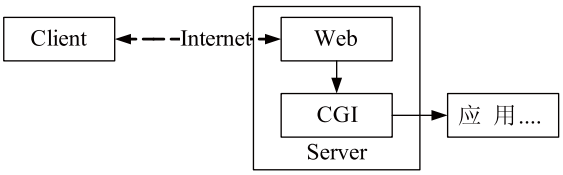

图 5　WebServer 原理

客户端通过 Internet 链接到嵌入式的 HTTPD 服务器上，HTTPD 首先分析 web 浏览器的 http 请求，然后 fork()一个子进程去处理这个请求，在刚 fork()出来的子进程中，根据 URL 去调用相应的 CGI 程序。CGI 程序可以用多种语言进行编写，这样执行 CGI 程序后就能实现所需要的功能。

本文使用 boa 作为 WebServer 服务器。移植 boa 首先要交叉编译 boa 应用程序；配置 boa 的配置选项 boa.conf，启用 CGI 支持，并且设定好网页路径和 BOA 启动的一些相关参数；然后设计制作好网页文件；最后启动 boa 服务进程。远端 PC 即可通过浏览器监控嵌入式平台。

3　平台测试

测试硬件平台为优龙公司的 FS2410 嵌入式开发板，它通过校园网接入 Internet，另一端为另一城市通过 adsl 上网的 PC 服务器，这是一个较常见的内外网连接模型。首先嵌入式平台通过 OpenVPN 和远端 PC 建立虚拟专用网连接。PC 端通过 telnet 和 ssh 均可以非常稳定登陆到嵌入式平台；PC 通过浏览器可以远程执行预定义的控制指令，也可在浏览器中流畅看到远端摄像头拍摄到的画面。在普通的 Internet 网络环境下，平台测试基本正常。为了模拟更多的工作环境，使用 vmware 并结合 Linux 的 iptables 技术，搭建一个恶劣的网络环境(50Kb 带宽，500ms 的时延，5％掉包),在此环境中继续进行测试，结果 telnet 和 ssh 的远程登陆影响不大，偶尔会断线，基于 web 的远程监控使用不是很正常，网页打开缓慢，监控指令反馈也偶有中断，摄像头传过来图象已经定格，从测试效果来看平台基本工作正常。

4　结束语

本文针对嵌入式 Internet 远程监控要求，从应用层面考虑，通过软件搭建了一个嵌入式的远程监控平台。其设计做法思路基本和平台无关，可应用于各种嵌入式平台。平台通过测试表明其设计思路正确，功能扩展方便，但实时、交互、稳定性还有待加强。此平台的搭建，为建立稳定专一的 C/S 构建的监控提供了必要的前提和基础。

参考文献

1 陈蓉,吴慧中.基于嵌入式的制造系统的远程监控系统.计算机工程, 2005,(6):179－180.

2 Intel Corporation. Embedded Intel architecture in virtual private network design, 2006.

3 Carlton R.周永彬译. IPSec:VPN 的安全实施.北京:清华大学出版社.

4 李善平.Linux 与嵌入式系统.北京:清华大学出版社, 2003.

5 谢仕义,徐兵.嵌入式 Web 服务器的设计及其 CGI 实现.计算机工程与设计,2007,4.