基于 RFB 协议的 Linux 远程桌面程序的分析与研究

李 晖，钟生海，王清理

（中国航天科工集团第二研究院 七〇六所，北京 100854）

摘 要：为基于国产平台研发安全可靠的远程桌面程序，对目前已有的各种远程桌面程序进行分析与研究，其中重点研究 Linux 平台上基于 RFB 协议的 VNC 程序。针对基于 RFB 协议的 VNC 程序，对其基本框架及工作流程进行分析与研究，重点研究图像消息的传输、屏幕变化的检测以及图像的压缩编码方式，针对目前国产化发展应用需求以及现状，为进一步提升 VNC 程序的性能，以满足国产信息系统的新需求，提出未来需改进的方向。

关键词：远程帧缓存协议；Linux 系统；远程桌面；虚拟网络计算；远程控制；屏幕共享

中图法分类号：TP311.5 **文献标识号**：A **文章编号**：1000-7024（2022）02-0346-06

doi：10.16208/j.issn1000-7024.2022.02.007

Research and analysis of Linux remote desktop program based on RFB protocol

LI Hui, ZHONG Sheng-hai, WANG Qing-li

(Institute 706, Second Academy of China Aerospace Science and Industry Corporation, Beijing 100854, China)

Abstract: To develop secure and reliable remote desktop programs based on domestic platform, various existing remote desktop programs were analyzed and studied, especially VNC programs based on RFB protocol on Linux platform. Aiming at the VNC program based on RFB protocol, the basic frame and workflow were analyzed and studied, the transmission of image message, the detection of screen change and the compression and coding method of image were studied emphatically. In view of the current domestic development and application requirements as well as the current situation, to further improve the performance of VNC program and to meet the new requirements of domestic information system, the future direction of improvement was put forward.

Key words: remote frame buffer (RFB) protocol; Linux operating system; remote desktop; virtual network computing (VNC); remote control; screen sharing

0 引 言

远程桌面技术是利用有线或无线网络由一个终端远程控制另一个或多个终端[1]，并且远程被控端的桌面界面图像可以显示于本地的控制端。使用本地控制端的设备对远程被控端设备进行操控时，如同直接操控远程设备，屏幕上显示的是被控端的桌面图像，并且可以操控鼠标打开被控端的应用程序，浏览被控端设备中的文件资料等。

在不同的平台之上有着各自独立的远程桌面系统，并且采用了不同的技术手段来实现，例如微软 Windows 系统常用的 RDP（remote desktop protocol）远程桌面协议，苹果 Mac 系统采用的 ARD（apple remote desktop）苹果远程桌面，而在 Linux 系统之上，各类 VNC 程序例如 Tight VNC、Real VNC、Ultra VNC 等较为常用。各类 VNC 程序主要框架基本一致，都包括了服务器端 VNC Server、客户端 VNC Client 以及用来连接服务器端与客户端的 RFB 协议，本文研究了主流 VNC 远程桌面软件的工作流程及原理，分析 VNC 远程桌面软件存在的主要问题，并提出了对

收稿日期：2021-04-13；**修订日期**：2021-09-27

作者简介：李晖（1984-），男，湖南邵阳人，硕士，高级工程师，研究方向为操作系统、国产信息系统集成等；钟生海（1983-），男，黑龙江鹤岗人，硕士，研究员，研究方向为国产计算机、信息系统集成等；王清理（1969-），男，河南漯河人，硕士，研究员，研究方向为嵌入式计算机、测试、云计算等。E-mail：lihui205214@163.com

该程序进行优化改进的建议。

远程桌面软件的流程分为初始化握手阶段以及正常交互阶段两部分。在正常交互阶段中，远程桌面软件主要实现两个功能，一个是远端系统桌面图像的显示，另一个是本地端对远端系统的控制。其中远端系统桌面图像的显示又涉及到图像消息的传输、屏幕变化的检测与图像消息的压缩编码这 3 个重要组成部分，对远端系统的控制涉及到对键盘及鼠标事件的模拟输入。

1　远程桌面软件工作流程

基于 RFB 协议的远程桌面软件工作流程分为两个阶段：初始化握手阶段和正常交互阶段。初始化握手阶段主要进行客户端连接服务器时的认证以及双方协商服务器向客户端发送图像画面消息时所采用的像素格式以及编码方式。在正常交互阶段中，服务器将根据初始化握手阶段所协商的像素格式以及编码方式，将图像发送给客户端，实现远程画面的显示；同时客户端将鼠标、键盘事件发送给服务器，实现对服务器的远程控制。

1.1　初始化握手阶段

初始化握手阶段主要分为协商版本号、协商认证方法、双方交换初始化消息、协商像素格式和编码方式这 4 个步骤。

首先，服务器与客户端双方进行握手，服务器将其所能支持的最高 RFB 协议版本号发送给客户端，客户端回复其所要使用的版本号，其所使用的版本必须小于或者等于服务器的版本号，目前主流的协议版本主要有 3.3、3.7 和 3.8。

接着双方协商所要进行安全认证的方式，主要分为两种：NONE 与 VNC 认证，采用 NONE 方式则不需要验证，采用 VNC 认证方式时，则使用挑战/应答机制身份认证方法，服务器向客户端发送一个 16 字节的随机数，客户端使用 DES 加密算法对其进行加密处理，秘钥采用用户的密码，将加密后的结果发送给服务器，服务器使用自己保存的密码以同样的方法对随机数进行处理，若处理得到的结果与客户端发送过来的结果一致，则说明客户端输入了正确的密码，此时服务器将认证成功的结果发送给客户端，否则服务器则返回给客户端认证失败的结果。

双方认证成功后，客户端向服务器发送一个字节的客户端初始化消息，该消息包含是否允许服务器共享屏幕的标志位，如果该标志位为真，则允许其它客户端同时连入该服务器，否则只允许当前客户端连接服务器。之后服务器向客户端发送服务器初始化消息，其中含有服务器的各种配置信息，包括：帧缓存的宽度和高度、像素格式以及桌面相关的名称。其中像素格式包含了每个像素需要的位数、像素值中有用的位数和是否为真彩色等相关参数，该像素格式为服务器原本的像素格式，并且在之后的发送图像画面消息中会沿用此格式，除非客户端发送了设置像素格式消息，请求了另外一种像素格式。

客户端如果要使用自己的像素格式，则向服务器发送设置像素格式消息，之后发送设置编码格式的消息，该消息包括了客户端所支持的所有编码类型，先后次序按所期望使用的优先级进行排列，首位具有最高优先级，而后服务器对所使用的编码类型进行选择。在此之后，服务器与客户端双方便进入了正常的协议交互阶段。

1.2　正常交互阶段

在服务器与客户端进行的正常交互过程中，客户端不断发送鼠标、键盘以及请求屏幕画面更新的消息，服务器则不断发送屏幕画面更新的数据，从而实现远程显示画面以及远程操控的功能。

2　图像传输方式

在 VNC 中，屏幕更新推送策略主要有两种分别是客户端主动拉取更新策略（Poll 机制）和服务端主动推送的更新策略（Push 机制）[2]。其中 Push 方式为服务器端每隔 10 ms 检测是否有图像的更新，若有则主动把更新的画面推送到客户端。而 Poll 方式则需要客户端主动发送帧缓存更新请求消息来对服务器画面的更新进行一个请求，服务器端收到该消息后，检测屏幕画面更新情况，若有更新，则向客户端发送画面的更新。VNC 默认情况下采用惰性更新的 Poll 方式[3]，因此其具有一定的自调节可适应性，根据客户端的处理速度以及当前的网络带宽情况进行相应传输速率的调整，保证了远程桌面不会因网络的延迟等因素造成画面更新错误的发生。

2.1　图像消息的传输

当程序进入正常交互阶段后，客户端需要不断发送帧缓存更新请求消息以获取服务器的画面更新。该消息包含消息的类型（默认为 3）、是否为增量更新、所请求需要更新画面左上角的 (x,y) 坐标以及画面的宽高。在通常情况下客户端所请求的为服务器端的全屏信息[3]，因此该消息中所请求需要更新画面左上角的 (x,y) 坐标默认为 $(0,0)$，宽高默认为服务器画面的宽高。增量更新代表着是否需要服务器发送客户端所请求画面的全部内容还是只需要发送变化的部分，因为通常情况下，客户端都保留着服务器帧缓存的副本，服务器只需要每次发送变化部分的更新即可保持客户端与服务器的同步。但是某些情况下客户端会丢失某一区域的内容，那么它便需要服务器将该区域的全部内容发送过来，此时发送帧缓存更新请求消息时，需要将其中的增量更新标志位设为 false。而在通常情况下，客户端不会丢失任何内容，因此每次发送帧缓存更新请求消息时，都将其中的增量更新标志位设为 true，服务器收到该消息后，会将其变化的内容发送给客户端。

当服务器收到客户端发送过来的帧缓存更新请求消息

时，会记录这一消息，同时调用 PollingManager 类的对象的函数进行屏幕变化的检测，若检测到屏幕自上一次发送更新之后，又产生了新的变化，则向客户端发送帧缓存图像更新消息，若画面没有任何变化则不发送。同时服务器有一个计时器，每隔 30 ms 也会对屏幕的变化进行一次检测，若检测到屏幕的变化，同时还有客户端的更新请求未处理时，则会向客户端发送最新的屏幕变化消息。该流程如图 1 所示。

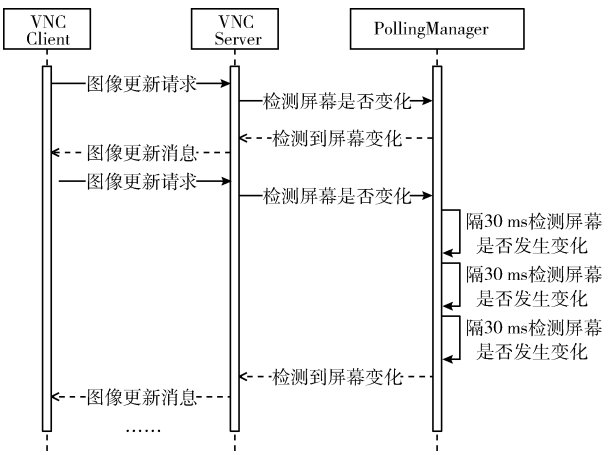

图 1 图像更新流程

帧缓存更新图像更新消息包含一系列像素数据矩形，客户端接收这些消息并解码，而后显示在自己的帧缓存中。其格式如下：首先为屏幕图像更新消息的头部，其中包含了消息的类型（默认为 0）、一字节的填充与所要发送的矩形块的个数。接下来是所要发送的各个矩形块的像素信息，包含了矩形块像素信息的头部和数据部分。其中头部包含了矩形块的左上角坐标、宽、高以及所使用的编码方式。数据部分为所使用指定编码方式进行编码的图像数据。

当客户端收到服务器发送过来的帧缓存更新图像更新消息之后，会将其解码，得到需要进行更新的矩形块的原始像素信息，根据像素信息填充到相应的坐标位置，便完成了一次屏幕图像的更新。同时客户端完成一次画面更新后，会再次向服务器发送帧缓存更新请求消息对服务器进行画面更新的请求，循环往复，这样便实现了客户端与服务器的同步显示。

2.2 屏幕变化的检测

在 Windows 操作系统中，屏幕的变化通常可以通过系统钩子（Hook）来获取[4]，因为在 windows 系统中的应用程序是基于消息驱动的，所有的应用程序通过对各种消息做出响应进而实现各种功能，而桌面的改变同样也会产生相应的消息，因此便可以通过系统钩子获取到屏幕的变化。

而在 Linux 中，RFB 协议则基于帧缓存的扫描来实现屏幕变化的检测，即通过新旧帧缓存的对比，找出屏幕变化的部分，具体实现方法如下：

首先对屏幕进行分块，将屏幕分为一个个宽高均为 32pixel 的小块，用 *m_changeFlags* [] 数组记录改变标志位，即记录哪一个小块发生了变化，另外用一个数据结构保存旧的屏幕图像。接下来便开始按照从上到下从左到右的顺序，对每一个小块分别进行扫描。对小块的新旧图像进行对比时，并不是对小块的每一个像素点都进行比较，而是只选取了其中的某一个横排（即 32 个像素点）作为代表，只扫描这一横排，对比判定是否一致，若一致，则未发生变化，若不一致，则说明该小块的图像发生了改变，之后将相应的标志位置为 true 记录这一改变，这样便减少了比较的次数，极大地提高了程序的效率。而具体取哪一行作为小块的代表，RFB 协议的作者经过测试，得出了以下的顺序：*m_pollingOrder*[32] = {0,16,8,24,4,20,12,28,10,26,18,2,22,6,30,14,1,17,9,25,7,23,15,31,19,3,27,11,29,13,5,21}，即当第一次扫描时，每一次只比较每个小块第 0 行的数据，第二次比较时，则比较每个小块第 16 行的数据，以此类推，如图 2 所示。通过对每个小块的扫描，便得到了屏幕变化的块，服务器将这些变化的块进行编码发送，客户端便得到了最新的画面。

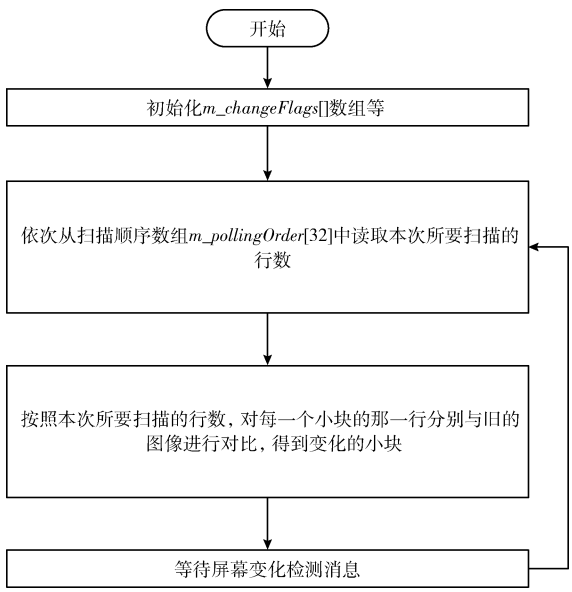

图 2 屏幕变化的检测流程

2.3 图像的压缩编码

当获取到屏幕变化的图像数据后，需要采取一定的压缩编码方法进行处理后传输；在客户端展现图像时，需用相应的算法对其进行译码便得到了服务器的图像。而图像的压缩编码方法是 RFB 协议的核心[5]，关系着远程桌面传输所占的带宽以及画面的质量。图像的压缩方法大体上分为如下两类：①有损压缩，即经过相应方法压缩处理之后，无论采用何种算法进行译码处理，所获得的图像与原始的

图像相比总会产生一部分失真，因为其允许一些数据的流失，因此此种压缩方法通常有着比较高的压缩比率；②无损压缩，经过压缩处理后，采用一定的算法进行译码处理，得到的图像与原始的图像数据完全一致，不产生任何失真，还原度高，但由于其保留着所有的图像数据，通常这类压缩方法的压缩比率会比较低[6]。目前 RFB 协议支持许多图像压缩处理方法，包含 Raw 编码、CopyRect 编码、RRE 编码、CoRRE 编码、Hextile 编码、ZRLE 编码以及伪编码等，下文将对其中几种具有代表性的编码进行简单介绍。

2.3.1　Raw 编码

Raw 编码，即原始编码，该编码不进行任何的数据压缩，保留原始的图像数据。此方法在同等条件下传输所占用的带宽最大，传输效率最低，但是因其不进行任何的编码处理，所以对 CPU 处理的负荷比较小。所有 RFB 服务器都必须支持 Raw 编码方式，通常情况下，如果客户端不特别请求使用别的编码方式，则服务器则默认使用 Raw 编码方式进行处理。

Raw 编码的处理方法如下：对于要进行编码处理的矩形图像块，从其左上角开始，从左到右，逐行对每一个像素点进行扫描，并按照其像素数据格式依次放入内存的编码区域。假设像素的格式为 32 位真彩色，则一个像素占 4 个字节，若所要发送的矩形区域高为 $height$、宽为 $width$，则这个矩形区域采用 Raw 编码所占总大小为 $4 * height * width$ 个字节。Raw 编码是最为简单的一种编码方式，在内存中编码后的数据与矩形内的像素点一一对应，此编码方法为其它编码方法的设计基础。

2.3.2　CopyRect 编码

CopyRect 编码，对于在客户端缓存中已有某些相同的图像数据区域，但是其位置发生改变的情况下十分有效，该编码仅包含待更新图像数据区域的坐标，客户端将已有的图像数据直接复制到指定坐标即完成了一次图像的更新。也就是说，算法只需要在第一次将矩形的所有像素数据发送，而后在服务器端该矩形的位置可能发生变化，但其矩形内的像素内容并没有发生改变，因此接下来只需要发送该矩形当前左上角的坐标即可完成客户端画面的更新。而对于矩形的第一次发送时所采用的编码方式并没有加以限定，所以该编码方法可以和其它编码方式结合使用，以达到更好的效果。该算法可以应用于许多种情形，其中最明显的就是当用户在屏幕上移动程序窗口的时候，采用该编码方法会极大地减少带宽的使用，提高传输效率。

2.3.3　RRE 编码

RRE 编码（rise-and-run-length encoding），正如它的名字所示，它的本质是 RLE 编码（run-length encoding），为其在二维空间上的实现。该编码的主要思想是将要传送的图像区域划分为一些子块，其中每个子块的像素点颜色都一致，将这些子块作为整体进行传输，可以极大程度上减少数据的重复表示。该编码主要利用了图像空间关联的特性，也就是空间中相邻像素的值通常一致或近似，因此该编码方法在一些图像较为简单的情况下十分有效，在复杂的桌面环境下效果差强人意。

该编码的具体方法如下：首先在所要发送的像素矩形中找到一个出现频率最高的像素值 Vb，将该像素值作为该矩形的背景色进行传送。然后将矩形分为一个个颜色一致的子块，子块个数记为 N，将子块的个数发送给客户端。接下来便要发送所有小子块的像素信息，格式为 $< v,x,y,w,h >$，其中 v 代表子块的像素值，并且 $v \neq Vb$，也就是和背景色像素值不相同。(x,y) 为子块左上角的坐标，(w,h) 分别为子块的宽度和高度。在进行分块处理时，该算法会分别对行与列都进行分块，然后判断这两种分块方法的压缩率，选择压缩率高的分块方法进行分块并发送。在接收到服务器发送过来的该编码消息后，客户端首先将背景色填充到指定的区域，接下来再分别把每个子块绘制到相应的位置即可完成图像的传递工作。

2.3.4　Hextile 编码

Hextile 编码[7]为 RRE 编码的变种，该编码会将所要进行编码处理的图像细分为 16 * 16 的小块，这样可以压缩控制参数所占的空间，可以仅仅用 4 bits 分别表示块的长度或宽度，也就是说块的左上角坐标 (x,y) 可以用一个字节来表示，块的宽度及高度 (w,h) 也可以只用一个字节来表示。排列顺序按照从左到右、从上到下依次进行编码，若图像的宽度非 16 的整数倍，则每行最右侧的块的宽度会进行相应地减少，同样地如果高度非 16 的整数倍，每列最下侧的块的高度也会相应减少。

对于每一个小块的图像数据的编码方式，可以采用 Raw 编码，也可以采用 RRE 编码的变种，该变种同样要选择出现频率最高的像素颜色作为背景色，之后要将小块再分成一系列的小矩形，其中小矩形中的每个像素都含有相同的颜色，然后便是将背景色与这些小矩形的数据发送即可。而与 RRE 编码不同的是，其中的背景色与前景色可以不用明确地写出，当不写明时，便会沿用上一个小矩形所使用的数据信息，节省了数据空间，提高传输效率。

2.3.5　几种编码的测试

通过在 Linux 系统下使用 VNC 软件，对几种主流编码方式所占用的带宽进行测试后发现：当进行一些简单操作并且图像变化不大时，使用各种算法效果均令人满意。但是当服务器的桌面分辨率较高，并且图像快速变化时，则会出现画面残影现象，并且流畅度不够高。使用 Raw 原始编码虽然可以提高流畅度，但会显著提高带宽的消耗，经测试，使用 Raw 编码在正常使用中平均占用带宽 10 Mb/s，若让屏幕图像快速变化，最高占用带宽 60 Mb/s。使用 RRE 等压缩编码虽然可以显著降低带宽，在实际使用中大约占用带宽 2 Mb/s，但是由于需要利用 CPU 对图像编码

数据进行额外的处理，因此图像的快速变化又会导致电脑的流畅度降低。

3　鼠标、键盘操作处理

3.1　鼠标消息的传递

客户端不断检测鼠标的移动、按键的点击与释放等事件，当此类事件发生时，便会向服务器发送相应鼠标事件的消息，从而对服务器进行控制。鼠标消息的格式见表 1。

表 1　鼠标消息格式

字节数	类型	说明
1	U8	消息类型
1	U8	按键掩码
2	U16	x 坐标
2	U16	y 坐标

鼠标消息的第一项为消息类型，这里默认值为 5。接着是按键掩码，由 1 字节表示，也就是 8 bit，鼠标的每个按钮分别对应其中的每一位，例如 0 号位为 true 代表着没有任何按键被按下，1、2 和 3 号位分别对应着鼠标的左中右键被按下，4 号位表示鼠标滚轮往上滑，5 号位表示鼠标滚轮往下滑。(x,y) 坐标代表了当前鼠标所在的位置。

3.2　键盘消息的传递

同样地，客户端也要对键盘的事件进行监测，当监测到键盘按键的按下与释放时，都要发送键盘的消息给服务器，其中键盘消息格式见表 2。

表 2　键盘消息格式

字节数	类型	说明
1	U8	消息类型
1	U8	按下标志位
2		填充
4	U32	键号

其中消息类型的值这里默认为 4。当键盘上某一按键被按下时，按下标志位为 true，释放时为 false。然后发送 2 字节的填充，之后发送键号。在 X Window 中按键被赋值为 keysym（键号），对于大多数按键来说，比如 abcd 等等，它们的键号与 ASCII 码是相对应的，而其它按键例如 Delete、Insert 等功能键则有另外的键号表示，具体参考 Linux 系统下的＜X11/keysymdef. h＞文件。

3.3　鼠标、键盘消息的处理

Linux 输入子系统是 Linux 内核用来管理各种输入设备的，一般情况下，鼠标和键盘设备分别对应着/dev/input/event0 和/dev/input/event1[8]。通过 Input 子系统可以实现对鼠标以及键盘操作的模拟，例如假设当前键盘设备为/dev/input/event1，可以向/dev/input/event1 设备文件写入一个字符“A”，这样便相当于通过键盘按下了 A 键，模拟了键盘的操作。

基于此种方式，客户端向服务器发送鼠标与键盘事件的消息后，服务器解析消息，并向相应的鼠标或键盘设备节点写入数据，模拟服务器的鼠标或键盘操作，进而实现了客户端对服务器的远程操控。

4　展　望

基于 RFB 协议的远程桌面程序工作流程分为两个阶段：初始化握手阶段和正常交互阶段。初始化握手阶段分为协商版本号、协商认证方法、双方交换初始化消息、协商像素格式和编码方式这 4 个步骤。正常交互阶段中，服务器通过将自己帧缓存中的内容发送给客户端实现了画面的同步显示，同时采用了只传输变化的部分以及运用了多种编码方式降低了画面传输过程中的带宽占用率。并且采用了 Poll 方式，屏幕的画面刷新完全由客户机驱动，通过客户端不断发送帧缓存请求消息来请求画面的更新，收到该请求后服务器处理该消息并向客户端发送画面的更新，这样使得 VNC 程序具有了相当的自适应性，其画面更新的快慢取决于当前客户端的处理能力以及当前的网络质量，保证了传输的稳定性。同时客户端将其鼠标与键盘的操作通过消息发送给服务器，服务器将其解析，并按照一定的格式向自己的鼠标或键盘设备节点写入相应的数据，模拟服务器端鼠标或键盘的操作，实现了客户端对服务器端的操控。

使用 RFB 协议的 VNC 程序，实现了远程桌面同步显示与控制的功能，针对目前国产化发展应用需求以及现状，现有的 VNC 技术已经不能满足应用需求，特别是在当前国产处理器性能较低的情况下，为了进一步提升 VNC 程序的性能，以满足国产信息系统的新需求，笔者认为其下一步的改进优化重点在于：

（1）图像压缩编码的优化：通过使用几种主流的编码方式进行图像传输测试后发现：当进行一些简单操作并且图像变化不大时，使用各种算法效果均令人满意。但是当服务器的桌面分辨率较高，并且图像快速变化时，则会出现画面残影现象，并且流畅度不够高。虽然随着计算机技术的不断进步，网速以及 CPU 处理速度会不断提高，二者均会使远程桌面软件有着更好的体验，但是依赖于硬件革新的同时，也需要对软件进行不断地优化，研究如何保证画面质量的前提下进一步减少带宽的占用，同时在图像快速变化时保证画面的流畅度，另外现阶段国产平台 CPU 处理速度与国外主流 CPU 相比还有着些许差距，因此对于画面的传输，需要对其图像的压缩编码方式进行进一步的研究优化，以达到更好的使用效果。

（2）安全性的提高：在 VNC 的连接认证中，采用了基

于挑战/应答机制身份认证的方法[9]，比起传统的静态口令身份认证，增加了随机数，通过随机数的唯一性和多变性，增加了安全性，可以有效抵抗多种攻击，但是其仍然存在着一些不足：例如服务器用于验证身份的密码保存在服务器中，虽然其采用了 Des 进行加密后存储，但是通过对几个 VNC 程序的源码进行研究后发现，其所使用的 Des 加密秘钥都为 {23,82,107,6,35,78,88,7}，因此黑客可以入侵服务器获取密码的密文，并通过此秘钥解密获得密码的明文，从而可以连接到服务器对服务器进行控制。另外在挑战/应答机制中的挑战随机数在网络中是以明文的形式发送的，无任何保护，易被黑客截取，而后可能采取离线密码猜测攻击获得秘钥的值[10]。同时 RFB 协议中服务器与客户端正常交互的过程中，服务器画面图像的传送以及鼠标键盘消息的传送都为明文传送，同样易被不法分子截获，还原图像从而使服务器上的隐私一览无余。因此为了提升国产信息系统安全性，助力国家安全，基于国产平台的远程桌面软件需要使用更安全的认证手段，比如智能卡、数字证书、USB KEY、生物认证等多技术结合的多因素认证方法，同时对 RFB 协议的正常交互阶段均采用密文形式发送。

5 结束语

针对基于 RFB 协议的 Linux 远程桌面 VNC 程序，本文研究了其基本框架及流程，重点研究了图像消息的传输、屏幕变化的检测以及图像的压缩编码方式，并对几种编码方式的效率进行了测试。最后针对目前国产化发展应用需求以及现状，为了满足全面国产信息系统的新需求，基于目前 VNC 所存在的问题，提出了更进一步的改进优化重点，包括图像压缩编码的优化，以及安全性的提高，以实现基于国产平台的安全远程桌面软件。

参考文献：

[1] LIU Jianing. Research on remote control technologies for Android systems [D]. Dalian: Dalian University of Technology, 2017: 7-8 (in Chinese). [刘嘉宁. Android 系统远程控制技术研究 [D]. 大连：大连理工大学，2017：7-8.]

[2] LIAO Zhibing. Design and implementation of real-time interactive system based on VNC [D]. Sichuan: University of Electronic Science and Technonogy of China, 2020: 33-34 (in Chinese). [廖志冰. 基于 VNC 的实时交互系统的设计与实现 [D]. 四川：电子科技大学，2020：33-34.]

[3] ZHU Yongqiang, TANG Xiong. Research and analysis of remote desktop transfer protocol based on VNC [J]. Computer Systems & Applications, 2016, 25 (11): 284-287 (in Chinese). [朱永强，汤雄. 基于 VNC 的远程桌面传输协议分析与研究 [J]. 计算机系统应用，2016，25 (11)：284-287.]

[4] ZHAO Zhiheng, YU Xiushan, HUANG Song, et al. Capture method based on Windows Hook's GUI testing [J]. Computer Engineering and Design, 2016, 37 (3): 660-664 (in Chinese). [赵志桓，于秀山，黄松，等. 基于 Windows Hook 的 GUI 测试操作捕获方法 [J]. 计算机工程与设计，2016，37 (3)：660-664.]

[5] WANG Rui, CHEN Liquan, SHA Jing, et al. RFB remote secure digital forensic scheme based on mutual authentication [J]. Journal of Nanjing University of Posts and Telecommunications (Natural Science Edition), 2017, 37 (3): 106-112 (in Chinese). [王睿，陈立全，沙晶，等. 基于双向认证的 RFB 远程安全数字取证方案 [J]. 南京邮电大学学报（自然科学版），2017，37 (3)：106-112.]

[6] ZHANG Xia. Classification and evaluation of image compression methods [J]. Journal of Taishan University, 2018, 40 (3): 81-83 (in Chinese). [张霞. 图像压缩方法分类及其评价 [J]. 泰山学院学报，2018，40 (3)：81-83.]

[7] QIAN Hao. The research on the key technology of VNC image transmission [D]. Chongqing: Chongqing University of Posts and Telecommunications, 2017: 1-11 (in Chinese). [钱浩. VNC 图像传输关键技术的研究 [D]. 重庆：重庆邮电大学，2017：1-11.]

[8] LI Zeyin, GONG Jun, WU Changhao. Research and implementation of matrix keyboard driver of Linux on Loongson2H [J]. Electronic Design Engineering, 2016, 24 (19): 180-183 (in Chinese). [李泽银，龚俊，吴昌昊. 基于龙芯 2H 的 Linux 矩阵键盘驱动的研究与实现 [J]. 电子设计工程，2016，24 (19)：180-183.]

[9] WANG Shuting, GUO Heng. Analysis of current status and trends of patent technology of password authentication [J]. China Invention & Patent, 2017, 14 (S1): 74-79 (in Chinese). [王淑婷，郭珩. 密码认证专利技术现状与发展趋势分析 [J]. 中国发明与专利，2017，14 (S1)：74-79.]

[10] CHEN Xian. The research and implementation of base station management system based on one-time password [D]. Sichuan: Southwest University of Science and Technology, 2017: 9-10 (in Chinese). [陈县. 基于动态口令的基站管理系统的研究与实现 [D]. 四川：西南科技大学，2017：9-10.]