

Remote Protocol in Home Virtualization System

Joonyoung Jung*, Jeonwoo Lee*, Daeyoung Kim**

* Green Computing Platform Research Team, Electronics and Telecommunications Research Institute, Daejeon, Korea

** Information Communication, Chungnam National University, Daejeon, Korea

jyjung21@etri.re.kr

Abstract— We have designed the home virtualization system with a remote protocol. In this system, several users enjoy their computing environment with home consumer electronics and just one virtualization server at the same time. Home consumer electronics, such as a TV and a speaker, should be connected with the home virtualization server to be used as I/O devices. In this system, multiple VM(virtual machine)'s are run in the virtualization server and multiple home consumer electronics are used as I/O(Input/Output) devices. So, home consumer electronics should be connected with the virtualization server automatically and dynamically and remote image data should be sent from the home virtualization server to home consumer electronics. We propose the remote protocol to do it.

Keywords— virtualization, VM, protocol, dynamic, connection, remote

I. INTRODUCTION

The virtualization system integrates and manages various ubiquitous computing devices as virtual device resources. It offers the best computing environment to user by combining virtual device resources nearby. The user can enjoy the computing environment anywhere in the virtualization system. The virtualization system is that a lot of users connect to one server and use applications independently [1]. There has been a technology for a desktop or a server virtualization. XenDesktop developed by Citrix System uses para-virtualization technology to improve I/O performance [2]. xVM developed by Sun Microsystems includes desktop virtualization, server virtualization and data center automation technology. It is operated in Solaris environment and supports Microsoft window OS, linux and Solaris as guest OS [3]. PnP-X developed by Microsoft extends the PnP(Plug & Play) function, that is, the network device is managed by PnP[5]. VMware View developed by VMware manages all devices at center and support virtual desktop to user [4]. These technology uses network protocol. For example, Microsoft, N-computing and Citrix make a network protocol to connect virtual server and client such as RDP(remote desktop protocol), UXP(user eXension protocol) and ICA(independent computing architecture) [5-8]. However these technologies don't include the automatic and dynamic connection mechanism between virtualization server and client. In this paper, we propose automatic and dynamic connection protocol between the virtualization server and virtualization clients (I/O devices) in the virtualization system.

We have designed and developed the home virtualization system. This system consists of one virtualization server, several VM images and several virtualization clients (home consumer electronics). In this system, multiple VM's are run in the virtualization server and multiple home consumer electronics are used as I/O devices. We propose the automatic and dynamic connection protocol between the virtualization server and multiple virtualization clients.

The rest of the paper is organized as follows. In section II, we show the home virtualization system. In section III, we propose the remote protocol. Conclusions are presented in Section IV.

II. HOME VIRTUALIZATION SYSTEM

The home virtualization system can make a multiple virtual PC environment by using multiple VM. Each VM configures a computing environment with multiple I/O devices.

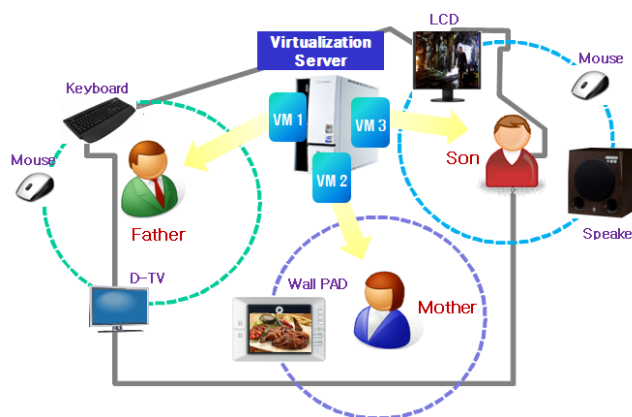


Figure 1. Example of Home virtualization system

Figure 1 shows the home virtualization system. Three VM's are running in the virtualization server simultaneously. Father checks his e-mail in the living room with a keyboard, a mouse, and a D-TV(Digital TV). Mother finds a recipe from internet in the kitchen with a touch screen wall pad. Son watches a movie in his room with a LCD monitor, a mouse, and a speaker. This is a just an example at home. The computing environment in the home virtualization system can be configured with various consumer electronics nearby.

III. REMOTE PROTOCOL

A. Dynamic Connection

We use Xen desktop as a virtualization server. In the virtualization server, there are several modules to connect with virtualization clients (consumer electronics) as seen figure 2. The SCM (server connection module) connects with virtualization clients automatically and dynamically. The SDM (server device manager) manages virtualization clients. The VIF (virtual interface) Agent manages the VIF-Driver. The VIF-Driver can send and receive an I/O data. In the virtualization client, there are some modules to connect with the virtualization server. The CCM (client connection module) connect with the virtualization server. The CDM(client device manager) manages a VIF. The VIF can send and receive an I/O data.

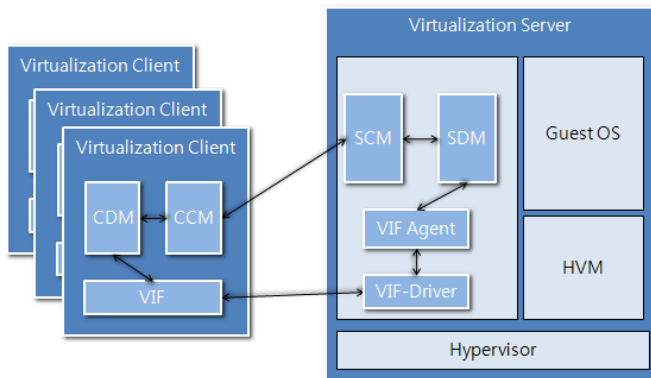


Figure 2. System structure

In the ubiquitous environment, there will be a lot of consumer electronics in home virtualization system. Therefore, consumer electronics (virtualization clients) should connect with the virtualization server automatically and dynamically.

In SCM, There are several modules for automatic and dynamic connection. The advertisement message module makes an advertisement message to announce the virtualization server information and broadcasts it periodically in the home virtualization system. The solicitation message module receives the solicitation message from virtualization clients and parses it. If the message is correct, the virtualization server sends the advertisement message to the virtualization client immediately. The connection managing module manages the connections with multiple virtualization clients. This module listen the connection request from virtualization client and saves the accepted connection information. Each virtualization client sends or receives the control data through this connection. The connection monitor module checks periodically if virtualization clients are disconnected or not.

In SDM, the device profile module receives a client profile from virtualization clients and save it client DB (data base). In SDM, the VM configure module chooses which virtualization client is used and then sends control message to the virtualization client through SCM. The I/O configure

module sends the virtualization client information to the VIF Agent. In VIF Agent, the creation module creates a VIF-Driver. The VM connection module connects between VIF-Driver and Guest OS (operating system). In CDM, the control message module receives a choose message through CCM. In VIF, the I/O data module connects with VIF-Driver to communicate I/O data with TCP/UDP connection.

In CCM, the IP address module searches the network interface of the virtualization client and finds IP address from network interface and then saves it. The solicitation message module makes a solicitation message and broadcasts it in the local network to announce that new virtualization client comes to the virtualization system and requests the advertisement message immediately. The advertisement message module receives the advertisement message from the virtualization server and parses it. If the message is correct, the virtualization client saves the information from the advertisement message to connect with the virtualization server. The keep alive module sends alive message periodically to the virtualization server.

In CDM, the client managing module manages the virtualization client. When this module receives client connection information from a lower layer, it makes the endpoint of the virtualization client and save it in the binding table. When this module receives client release information from a lower layer, it removes the saved information in the binding table. The client profile module makes a client profile and sends it to the virtualization server through CCM.

B. Remote Image Data Communication

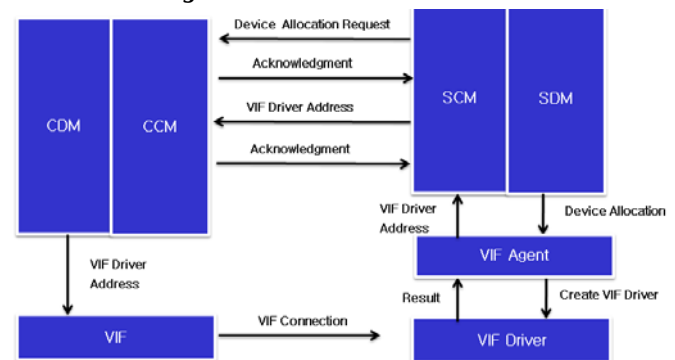


Figure 3. Remote Image Protocol

As seen in figure 3, if a virtualization server requests device allocation from the virtualization client, the virtualization client sends an acknowledgement message to the server. After receiving the acknowledge message, SDM informs a VIF Agent of device allocation. VIF Agent that receives device allocation makes a VIF Driver and then receives the result from VIF Driver. VIF Agent sends VIF Driver address to the virtualization client through the SCM. The virtualization client sends an acknowledgement message to the virtualization server. The CDM sends the VIF Driver address to VIF. The VIF connects with VIF Driver to communicate image data. The VIF Driver and VIF communicate remote image data after completing the connection.

We propose the messages for remote image data communication.

1) Messages from the virtualization client(VIF) to the virtualization server(VIF Driver)

The virtualization client sends client initialization message (Client Init Msg) to the virtualization server as shown in Table 1. Version means the remote image protocol version. Length means the length of message. Type means the message type. FrameBufferFormat means the frame buffer format as seen in Table 2. PixelFormat means the pixel format as seen in Table 3. EncodingFormat means the encoding format as seen in Table 4. Client_name means the virtualization client name

Table 1. Client Init Msg

Field Name	Data type	Description	Value
Version	uint8	Protocol version	
Length	uint16	Msg length	
Type	uint8	Msg type	
FrameBufferFormat	FrameBuffer_Format	default frame buffer format	see Table 2
PixelFormat	Pixel_Format	default pixel format	see Table 3
EncodingFormat	Encoding_Format	default encoding format	see Table 4
Client_name	char[25]	Virtualization Client name	

FrameBufferFormat means the frame buffer format. FrameBufferInfoNum means the frame buffer format information number and is used to classify a frame buffer format. FrameBuffer-width and FrameBuffer-height means the width and height of frame buffer.

Table 2. Client Frame Buffer Format

Field Name	Data type	Description
FrameBufferInfoNum	uint8	Frame buffer format information number
FrameBuffer-width	uint16	frame buffer width
FrameBuffer-height	uint16	frame buffer height

PixelFormat means the pixel format. PixelFormatInfoNum means the pixel format information number and is used to classify a pixel format. BitsPerPixel means the number of bits used for each pixel value. BigEndianFlag is non-zero (true) if multi-byte pixels are interpreted as big endian. If TrueColorFlag is non-zero (true) then the pixel value has red, green and blue value. If TrueColorFlag is zero (false) then this indicates that a “colour map” is to be used. BitsForRed means the number of bits used for red value. Shift-Red is the number of shifts needed to get the red value in a pixel to the least significant bit. BitsForGreen, BitsForBlue, Shift-Green and Shift-Blue are similar for green and blue.

Table 3. Client Pixel Format

Field Name	Data type	Description
PixelFormatInfoNum	uint8	Pixel format information number

BitsPerPixel	uint8	bits per Pixel
BigEndianFlag	uint8	big endian flag
TrueColorFlag	uint8	true color flag
BitsForRed	uint8	bits for red
BitsForGreen	uint8	bits for green
BitsForBlue	uint8	bits for blue
Shift-Red	uint8	the number of shifts for red
Shift-Green	uint8	the number of shifts for green
Shift-Blue	uint8	the number of shifts for blue

EncodingFormat means the encoding format. EncodingFormatInfoNum means the encoding format information number and is used to classify an encoding format. GEncodingNum means the number of general encodings. GEncodingTypeList is the list of general encoding type. MEncodingNum means the number of moving picture encodings. MEncodingTypeList is the list of moving picture encoding type.

Table 4. Client Encoding Format

Field Name	Data type	Description
EncodingFormatInfoNum	uint8	encoding format information number
GEncodingNum	uint8	number of general encoding
GEncodingTypeList	int32[GEncodingNum]	general encoding type list
MEncodingNum	uint8	number of moving encoding
MEncodingTypeList	int32[MEncodingNum]	moving encoding type list

The virtualization client sends the client frame buffer format message to the virtualization server to inform the frame buffer format of the virtualization client as seen Table 5. Version means the remote image protocol version. Length means the length of message. Type means the message type. FrameBufferFormat means the frame buffer format as seen in Table 2.

Table 5. Client Frame Buffer Format Msg

Field Name	Data type	Description	Value
Version	uint8	Protocol version	
Length	uint16	Msg length	
Type	uint8	Msg type	
FrameBufferFormat	FrameBuffer_Format	frame buffer format	see Table 2

The virtualization client sends the client pixel format message to the virtualization server to inform the pixel format of the virtualization client as seen Table 6. Version means the remote image protocol version. Length means the length of

message. Type means the message type. PixelFormat means the pixel format as seen in Table 3.

Table 6. Client Pixel Format Msg

Field Name	Data type	Description	Value
Version	uint8	Protocol version	
Length	uint16	Msg length	
Type	uint8	Msg type	
PixelFormat	Pixel_Format	pixel format	see Table 3

The virtualization client sends the client encoding format message to the virtualization server to inform the encoding format of the virtualization client as seen Table 7. Version means the remote image protocol version. Length means the length of message. Type means the message type. EncodingFormat means the encoding format as seen in Table 4.

Table 7. Client Encoding Format Msg

Field Name	Data type	Description	Value
Version	uint8	Protocol version	
Length	uint16	Msg length	
Type	uint8	Msg type	
EncodingFormat	Encoding_Format	encoding format	see Table 4

2) Messages from the virtualization client(VIF) to the virtualization server(VIF Driver)

After receiving the Client Init Msg from the virtualization client, the virtualization server sends server initialization message (Server Init Msg) to the virtualization client as shown in Table 8. Version means the remote image protocol version. Length means the length of message. Type means the message type. FrameBufferFormat means the frame buffer format as seen in Table 9. PixelFormat means the pixel format as seen in Table 10. EncodingFormat means the encoding format as seen in Table 11. Server_name means the virtualization server name.

Table 8. Server Init Msg

Field Name	Data type	Description	Value
Version	uint8	Protocol version	
Length	uint16	Msg length	
Type	uint8	Msg type	
FrameBufferFormat	FrameBuffer_Format	frame buffer format	see Table 9
PixelFormat	Pixel_Format	pixel format	see Table 10
EncodingFormat	Encoding_Format	encoding format	see Table 11
Server_name	char[25]	Virtualization server name	

FrameBufferFormat means the frame buffer format. FrameBufferInfoNum means the frame buffer format information number and is used to classify a frame buffer format. FrameBuffer-width and FrameBuffer-height means the width and height of frame buffer. MFrameBufferFlag is zero(fault) if there is not a moving picture region and is non-zero(true) if there is a moving picture region. The moving picture region is shown with MFrameBuffer_X, MFrameBuffer_Y, MFrameBuffer-width and MFrameBuffer-height value.

Table 9. Server Frame Buffer Format

Field Name	Data type	Description
FrameBufferInfoNum	uint8	Frame buffer format information number
FrameBuffer-width	uint16	frame buffer width
FrameBuffer-height	uint16	frame buffer height
MFrameBufferFlag	uint8	moving frame buffer Flag
MFrameBuffer_X	uint16	moving frame buffer X position
MFrameBuffer_Y	uint16	moving frame buffer Y position
MFrameBuffer-width	uint16	moving frame buffer width
MFrameBuffer-height	uint16	moving frame buffer height

PixelFormat means the pixel format. PixelFormatInfoNum means the pixel format information number and is used to classify a pixel format. BitsPerPixel means the number of bits used for each pixel value. BigEndianFlag is non-zero (true) if multi-byte pixels are interpreted as big endian. If TrueColorFlag is non-zero (true) then the pixel value has red, green and blue value. If TrueColorFlag is zero (false) then this indicates that a “colour map” is to be used. BitsForRed means the number of bits used for red value. Shift-Red is the number of shifts needed to get the red value in a pixel to the least significant bit. BitsForGreen, BitsForBlue, Shift-Green and Shift-Blue are similar for green and blue.

Table 10. Server Pixel Format

Field Name	Data type	Description
PixelFormatInfoNum	uint8	Pixel format information number
BitsPerPixel	uint8	bits per Pixel
BigEndianFlag	uint8	big endian flag
TrueColorFlag	uint8	true color flag
BitsForRed	uint8	bits for red
BitsForGreen	uint8	bits for green
BitsForBlue	uint8	bits for blue
Shift-Red	uint8	the number of shifts for red
Shift-Green	uint8	the number of shifts for green
Shift-Blue	uint8	the number of shifts for blue

EncodingFormat means the encoding format. EncodingFormatInfoNum means the encoding format

information number and is used to classify an encoding format. GEncodingType means the general encoding type. MEncodingType means the moving picture encoding type.

Table 11. Server Encoding Format

Field Name	Data type	Description
EncodingFormatInfoNum	uint8	encoding format information number
GEncodingType	int32	general encoding type
MEncodingType	int32	moving encoding type

The virtualization server sends the server frame buffer format message to the virtualization client to inform the frame buffer format of the virtualization server as seen Table 12. Version means the remote image protocol version. Length means the length of message. Type means the message type. FrameBufferFormat means the frame buffer format as seen in Table 9.

Table 12. Server Frame Buffer Format Msg

Field Name	Data type	Description	Value
Version	uint8	Protocol version	
Length	uint16	Msg length	
Type	uint8	Msg type	
FrameBufferFormat	FrameBuffer_Format	frame buffer format	see Table 9

The virtualization server sends the server pixel format message to the virtualization server to inform the pixel format of the virtualization server as seen Table 13. Version means the remote image protocol version. Length means the length of message. Type means the message type. PixelFormat means the pixel format as seen in Table 10.

Table 13. Server Pixel Format Msg

Field Name	Data type	Description	Value
Version	uint8	Protocol version	
Length	uint16	Msg length	
Type	uint8	Msg type	
PixelFormat	Pixel_Format	pixel format	see Table 10

The virtualization server sends the server encoding format message to the virtualization client to inform the encoding format of the virtualization server as seen Table 14. Version means the remote image protocol version. Length means the length of message. Type means the message type. EncodingFormat means the encoding format as seen in Table 11.

Table 14. Server Encoding Format Msg

Field Name	Data type	Description	Value
Version	uint8	Protocol version	
Length	uint16	Msg length	
Type	uint8	Msg type	
EncodingFormat	Encoding_Format	encoding format	see Table 11

IV. CONCLUSIONS

We propose the remote protocol in the virtualization system. This protocol is one of essential technologies because the virtualization server is connected with consumer electronics dynamically and image data is sent from the virtualization server to consumer electronics by this protocol.

ACKNOWLEDGMENT

This work was supported by the IT R&D program of MKE/KEIT, [2008-S-034-02, Development of Collaborative Virtual Machine Technology for SoD(System on-Demand) Service].

REFERENCES

- [1] Jason Nieh, S. Jae Yang, Naomi Novik, "A Comparison of Thin-Client Computing Architectures". Technical Report CUCS-022-00 November 2000.
- [2] Citrix White paper, "Desktop Virtualization with Citrix XenDesktop" January 06, 2009
- [3] Sun White paper, "SUN xVM VIRTUALIZATION PORTFOLIO: VIRTUALIZING THE DYNAMIC DATACENTER", August 2008
- [4] VMware White paper, "Solving the Desktop Dilemma : With User-Centric Desktop Virtualization for the Enterprise"
- [5] Microsoft White paper, "PnP-X: Plug and Play Extensions for Windows", January 11, 2007
- [6] Ncomputing White paper, "Technology White Paper", February 2002.
- [7] Microsoft White paper "Remote Desktop Protocol (RDP) Features and Performance", December 2002.
- [8] Tristan Prichardson, "The RFB Protocol Version 3.8", March 2009.