Virtualization as a Strategy for Maintaining Future Access to Multimedia Content

Knut Holmqvist, Till Halbach, and Thor Kristoffersen
Norwegian Computing Center
Oslo, Norway
{Knut.Holmqvist,Till.Halbach,Thor.Kristoffersen}@nr.no

Abstract—**Virtualization and emulation techniques are getting better and better, and it feels thus natural to apply them to address the problem of data preservation for future content access. In this paper, we virtualized a number of existing servers and clients and ran the virtual machines under VMware's hypervisor. Not all of the investigated operating systems could be virtualized successfully, and even with successful conversions often settings had to be reconfigurated, and support for particular devices had to be added manually. It is concluded that many operating systems are not flexible enough to allow for a problem-free conversion to a virtual machine.**

Keywords—Data preservation, virtualization, emulation, hypervisor, client server, service, multimedia, MARIAGE

I. INTRODUCTION

We are all witnesses of more and more digital content being produced and consumed. Preserving this content for the future has been and still is a hot topic among researchers, not only for "passive" data like text, music, and video, but also when it comes to executables/applications [1].

In this paper we discuss some of the challenges encountered when addressing the problem of making future access to multimedia content possible and simple. The paper is part of a research project at the Norwegian Computing Center with the name MARIAGE – Making Rich Media Accessible for Generations [2].

Two principal methods for preserving content exist, data migration and system emulation. Migration adapts or transcodes content so it can be played on new devices [3]. E.g, a VHS tape is ripped, transcoded into MPEG-2 and put on a DVD, played on a DVD player, and showed on a LCD TV-set. Emulation, on the other hand, recreates the old environment on the new hardware and uses the content as is [4]. For instance, a Commodore 64 emulator is downloaded to a PC, and the content of concern (here: games) is copied from the Commodore tape recorder, and then Commodore games can be played on the PC. While transcoding has been the main strategy for long-term use of content, emulation may become a viable alternative, in particular in the light of current research [3] and development of virtual machines as detailed below.

The paper is organized as follows. Before we focus on technical details in Section III, we give background information about virtualization and emulation in Section II. The experiments are discussed in detail in Section IV, and we have devoted Section V to challenges faced during virtualization. The article closes with a summary and conclusions.

II. RATIONALE FOR VIRTUALIZATION AND EMULATION

Virtualization of software is coming of age. Many computers nowadays run several virtual machines and run combinations of operating systems. The idea behind goes decades back. In the 1970s, IBM's 370 architecture allowed to run multiple operating systems at the same time, thereby making time sharing on a single processing unit more secure, more efficient, and easier to administer [5]. Batch and online processes were running simultaneously. These systems were often called Virtual Machine Monitors.

With mini computers and PCs the demand for virtualization diminished. Mini computers were often specialized for one purpose, and several physical computers were used instead of one physical computer with multiple logical computers [6].

Today the demand for virtualization arrives primarily from the server market [7]. With increasing computing power and multi-processor architecture, computers have the capacity to run several tasks in parallel. Also, many businesses require their products to be deployable on several operating systems and hence have a higher demand for virtualization during development and testing than previously. Other reasons include the interest in out-of-production systems like Atari and Amiga, an improved load factor of the available hardware (the costs of which are higher than those inherent to virtualization), an increasing number of operating systems to support, and environmental considerations. Moreover, emulation may be cheaper than data migration [4].

On portables and desktop computers, more and more people want/need to run several operating systems. All systems have different strengths and weaknesses. To be able to choose some applications from one operating environment and others from another is often an advantage. Also, with the market dominance of Microsoft, people are typically bound to use applications which are often only offered for the Windows operating system. Virtualization software helps these groups in choosing "the best of all worlds".

Projects in our research institute often result in prototypes and proof-of-concept multimedia applications. These applications usually utilize particular aspects of the operating system and hardware it runs on and are not upgraded to new versions of the operating system. We thus have a large collection of old computers running just a single application. In the process of building an emulation framework for old software programs and systems, we have virtualized several of these computers. But before we discuss the experiments conducted we introduce basic concepts within virtualization and emulation.

III. Virtualization details

There are many issues related to environment emulation. Subsequently, we concentrate on the technical aspects of the problem, ignoring licensing policies of different vendors and rights management issues.

The overall system architecture is as follows. A physical machine with I/O devices and peripherals runs an operating system (OS), the so-called host. The host runs specialized software called hypervisor [8] which provides emulators for one or several virtual machines with their own OS, also referred to as guests. The role of the hypervisor is to make itself transparent to the guest such that the guest 'believes' it runs on a physical machine.

In this way, several guests have to share the physical devices of the host. Internal devices are

- processor and
- memory.

I/O devices include serial and parallel devices, sometimes also referred to as ports, as well as bus systems like USB. The list of external devices includes

- harddisk, floppy disk, optical and magnetic drives, and other storage devices,
- network adapter,
- keyboard, mouse, and joystick,
- screen,
- speaker,
- printer,
- scanner, and
- still-image or video camera.

Sharing of physical devices means their time-multiplexed use. An OS usually deploys a job control mechanism that regulates which process may use the processor and for how long. Device drivers often manage control of internal and external devices. To enable device sharing, the control software and drivers of the host must support time-sharing.

Next, the hypervisor for a particular guest must provide virtual devices for all devices required by the guest to access specific content. It also must take care of emulating no-longer-existing I/O hardware as virtual devices for peripherals like the Commodore 64 joy-stick, and mapping the virtual devices to physical devices. This process depends greatly on the host, and there must be explicit support of the hypervisor for a certain host system. Examples for various guest systems are early computers like Commodore, Amiga, and Atari, as well as Macintosh and early MS-DOS and Windows systems, video game consoles like PlayStation, Wii, and Xbox, and any kind of mobile phone device or smartphone.

As stated above, virtual machines are nowadays often used as servers. Here, the need for shared peripherals is typically less critical than with desktop and laptop computers, as there are usually only a few shared peripherals, such as harddisk and network adapter. For a desktop computer, often many more peripherals are shared; see the listing of peripherals above for examples.

There is one practical problem with virtualization (which, as a side note, also applies to data migration): A functioning reading device is always needed at the beginning of the virtualization process to capture the original content in order to copy it.

In our field trials, the reading device is a harddisk. Depending on the OS of the computer to virtualize, at least the partition containing the system must be copied byte-wise into a so-called image file. Other partitions containing file systems and other data might have to be copied as well. The copying can e.g. be achieved by requiring a CDROM drive, in which a CD is put with the Linux distribution Knoppix on it, which in turn contains the software capable of byte copying. The raw image file generated by this procedure must then be converted into an image file with a format that the emulation software supports. The generation of some metadata about the system and its configuration might be required additionally.

IV. Field trials

We have virtualized several machines, i.e., converted them from the state of being a physical system to the state of being a virtual machine, to give a proof of concept of service virtualization. The hosts virtualized are demonstrator applications of previous projects. In our setup the physical machine is based on Intel Core 2 Duo E6600 processors with 4 GB RAM and a 500 GB disk. It runs the operating system Debian 5.0, also known as Lenny.

We have used the virtualization vendor VMware for our preliminary work in this area. In addition we have tested Microsoft's Hyperware. As of today, VMware has a richer functionality and supports more versions of Windows and Linux, as well as NetWare, Solaris, and a couple of other OSs like MS-DOS, than its competitor Hyperware [9]. We therefore only report our experiences with VMware in this paper. Although VMware has limitations with regard to the requirements put forward in MARIAGE, where we would also like to handle e.g. Macintosh computers and TV games based on PlayStation, Xbox, and other consoles, we found VMware to be the best solution on the market as of today.

A. *A talking-head demonstrator*

This experiment demonstrates the virtualization of a client server system and cross-guest communication over a virtual port.

The talking-head demonstrator was developed for Ericsson for the 2000 CeBIT exhibition. It demonstrates the use of a virtual news anchor service on a GSM mobile phone. The demonstrator was made before the advent of EDGE and other higher-bandwidth mobile capabilities [10].

The client animates an avatar with lip synchronization while playing audio and showing accompanying images and text. The animation runs inside a mobile phone emulator, emulating a Symbian OS. The emulator in turn runs on top of Windows NT. The server, running on Red Hat Linux 7.3, sends phonemes over the GSM network to its clients.

The virtualization process involves the machine on which the server runs, as well as the machine with the client emulator. As both become guests running on the same physical machine, they are connected through a logical serial cable, a so-called virtual port, obsoleting any now inefficient external physical channel such as the transmission over a cellular network or a physical serial port. This logical connection is implemented in software as a so-called pipe, a simple file. It is crucial for the proper functionality of the pipe to have defined reading and writing policies by client and server, respectively, and the type of pipe change notifications to the guest's OS, such as polling or interrupt mode. The server and client network devices are connected to their respective serial ports through the Point-To-Point Protocol.

Concluding, client and server services based on a serial connection can be virtualized successfully. While inter-guest communication can be established over a serial connection effortlessly, extensive use of the aforementioned port polling method by servers and clients likewise may make the host and other guests run sluggishly, as this method consumes a disproportionate share of (guest and host) processor time.

B. A radio streamer production environment

This experiment demonstrates the virtualization of a server system and the sharing of a single physical device (a sound card and a network card) across several machines.

For several years, our research institution has been streaming the radio frequency 99.3 MHz in the Oslo area. This frequency has been shared by the student's Radio Nova, the women’s radio radiOrakel, and Tellus Radio which produces content for immigrants [11].

The basic components of the production environment are an external FM radio and a PC with a sound card. The radio is tuned in to the desired frequency and outputs an analog signal to the PC´s audio card. The PC's OS is Linux 2.4.x (Debian 3). Moreover, the system runs the streaming server IceCast to encode the audio stream in MP3 format with three different qualities (i.e., simulcast) and sends it over the network. In addition to streaming, the encoded audio is stored as files. These files allow for easy access to previous radio programs in the form of downloading or streaming. Tellus Radio, as an example, makes links to “Today’s Tamil transmission”, “Today’s Urdu transmission”, and so on. The communication with the Internet is based on the web server Apache.

During the virtualization, the sharing of the network card is achieved by the host which provides a virtual network device /dev/vmnet0, to/from which the network activity of all guests is directed. The sampled and encoded audio is sent over this shared network device. The guest machine has its own IP address.

Considering the sharing of the host's sound card, it is crucial to map the guest's audio device (/dev/dsp), which previously was the access point to the computer's sound card, to the host's audio device and hereby the host's physical audio device. The driver of the host's sound card must have sound sharing enabled to allow several virtual machines to access the single physical sound card simultaneously, otherwise one guest's use of the audio device will block its use of other virtual machines or of other host applications.

The virtualization of the final system is completely transparent for the end user because the virtual machine has the same behavior as the original system. That is, access to the streaming server and the web server is as before, provided that either the guest's IP address is maintained or that the appropriate name-to-IP mapping is stored in the DNS server.

Concluding, server services relying on functionality of audio and network card can be virtualized without problems, provided that the host's driver software for the audio card allows device sharing.

C. Other virtualizations

We have further virtualized computers with various OSs and hardware settings, e.g. Red Hat Linux 7.3, 8.0, Microsoft (MS) Windows Server 2003, MS Windows NT Workstation 4.0, Debian Linux 3.0, MS Windows 2000 Professional, MS-DOS, Suse Enterprise Linux 10.0, Ubuntu Linux 7.04, and Windows XP.

Almost all Linux computers could be virtualized with success, whereas all involved Windows systems except for NT Workstations have turned out to be problematic, i.e., their virtualization failed. There are multiple reasons for that. Windows OSs are very sensible to changes in the hardware configuration. The virtualization of a physical system means a change in the hardware configuration from that system's point of view. In the best case, the OS will automatically reconfigure itself to run with the new hardware, and in the worst case it will not run at all.

Linux systems are in general resilient to changes in very basic hardware, like mainboard and disks. In our experiments, some Debian systems that use LILO as a boot manager failed to boot, caused by flaws in the boot manager. In contrast to that, Red Hat booted, detected the changes in hardware, and started the hardware configuration manager. The user was offered the possibility to change the configuration and only needed to accept the defaults offered. However, the configuration of the X Window System, a client server display protocol, had to be carried out by hand. Suse booted but complained about missing hardware, as it did not detect the changes automatically. The system's configuration had to be changed by running Suse's configuration manager yast2. Ubuntu had the MAC address of the physical network card buried in some configuration files, which gave a device name conflict during start-up after the virtualization. As a result, networking was unavailable and had to be configured manually.

Windows systems are in general very particular about the hardware they are running on, including very basic hardware. It is not unusual that a Windows system fails to boot when it is moved to a different mainboard. In our experiments, the NT Workstation booted but complained about changed devices. All other mentioned systems got stuck during the boot process. Considering Windows XP, we also tried a so-called "repair install" of the installation. This produced a working virtual machine, but it ran very slowly and was more or less unusable.

Concluding, the virtualization of physical machines with current technology requires manual updating of the configuration for those machines, and possibly the installation of different device drivers, too. It appears that the technology is not ripe yet to avoid such hassles.

V. Virtualization Challenges

The demand for other device drivers and altering the system configuration were already mentioned in the previous section. This applies not only to the virtualization process itself but also to situations where a guest is moved to another host. Other than that, we encountered the following challenges in our field trials.

To be able to display its visual output, the virtual machine has to make use of the memory of a graphic card. With VMware, resources on the graphic card of the system on which a console accesses the guest are reserved for this purpose. Consequently, the guest's display can be forwarded to arbitrary machines, hereby reducing the resource consumption of the host, especially when it comes to graphical displays. We have successfully tested such a system; problems can, however, arise e.g. in case of network congestion, which limits the amount of graphical information sent to the display terminal, which for one virtual machine lead to a reduced displayed color depth.

The forwarding of audio from multiple guests to other sound cards has proven to be possible but subject to restrictions of the network connection's bandwidth as well. With VMware, the guest's audio is sent to the sound card of the system on which a terminal accesses the guest. This allows, as with the display data, forwarding to arbitrary terminal machines, but may lead to sound jitter or bad quality with poor network conditions.

The VMware hypervisor used in our field trials had flaws concerning the sharing of a single CDROM/DVD drive on the host among several guests. This might be problematic when for instance the same CDROM has to be accessed by two different guests. However, since several physical drives on the host is a possibility, where each drive is accessed by a different guest, CDROMs can be copied and put into each of those drives as a work-around.

Concluding, we have found that the sharing of crucial resources like processor, memory, and hard disk, as well as a network device can be accomplished without problems. However, the capacity of the network connection is the most important factor limiting the fidelity of other resources like of course guest network, and display and sound forwarding.

VI. Summary and Conclusions

After giving some background information on virtualization, we have discussed the details of the involved technical solutions. We then conducted a number of experiments in which existing computers were virtualized. The article gave the technical details in depth and reported also on the problems and challenges encountered.

We have shown that virtualization and emulation are, with the current state of the art, only limited solutions to the problem of maintaining future access to digital information in terms of multimedia data and applications and services.

The virtualization of existing systems is a complex and involved process. Not all of the systems investigated could successfully be transformed to virtual machines. Linux has been shown to be quite robust during this process, even though additional configurations and installations typically had to be applied manually. Windows systems, with the exception of Windows NT, showed only a little degree of re-configuration capability, and they also lacked the possibility to carry out the necessary configurations and installations manually. It appears, though, that the problems encountered are not due to limitations with the hypervisor but are rather manifested by an insufficient flexibility of the operating system to be virtualized. Summing up, virtualization and emulation cannot be recommended in their current technological state as efficient solutions to the problems posed.

Concerning the hypervisors of different vendors, we have found that VMware currently provides the best solutions available on the market today.

Acknowledgments

The authors would like to thank Wolfgang Leister for support during some of the virtualization processes, and for proof reading.

References

[1] Raymond A. Lorie. *Long term preservation of digital information*. International Conference on Digital Libraries. Roanoke (Virginia, USA), 2001, pp. 346-352

[2] Norwegian Computing Center. *MARIAGE — Making Rich Media Accessable for Generations*. http://www.nr.no/pages/dart/project_flyer_mariage, accessed 2009-02-25

[3] Jeffrey van der Hoeven, Bram Lohman, and Remco Verdegem. *Emulation for Digital Preservation in Practice: The Results.* The International Journal of Digital Curation 2.2 (2007): 123-132

[4] Gregory Muira. *Pushing the Boundaries of Traditional Heritage Policy: maintaining long-term access to multimedia content*. IFLA Journal 33 (2007): 323-326

[5] Pugh et al. *IBM's 360 and Early 370 Systems*. MIT (1991). ISBN 0-262-16123-0: 160-161

[6] Anthony Ralston and Edwin Reilly. *Workstation*. Encyclopedia of Computer Science (third edition 1993). Van Nostrand Reinhold, New York. ISBN 0442276796

[7] Stephen Shankland. *Virtualization: A feature of the hardware, not the OS?* http://news.cnet.com/Virtualization-A-feature-of-the-hardware,-not-the-OS/2100-7339_3-6206867.html, accessed 2008-12-02

[8] IBM Systems. *Virtualization – description of basic concepts.* IBM Corporation, version 2 release 1 (2005)

[9] VMware. *Whitepaper – Why choose VMware?* http://www.vmware.com/technology/whyvmware/, accessed *2009-02-25*

[10] L. Aarhus, H. Hegna, T. Kristoffersen, W. Leister, A. Moen, and B. Østvold. *Streamed multimedia presentation for low-bandwidth mobile terminals: A virtual machine approach.* Proceedings 3Gwireless, San Francisco (CA, USA), May 2002

[11] Knut Holmqvist og Eirik Maus. *En oppsummering av delprosjekter Radiostrømmer i ChannelS* (Norwegian only), Norwegian Computing Center, report number: 10-04, January 2004