Chapter 1 : M. Tim Jones Publications

AI Application Programming by M Tim Jones (second edition) focuses on classical AI problems, game playing, and AI for the web. I am interested in so called Artificial Intelligence algorithms due to my past work in robotics but also due to my current engineering work.

In a nutshell, virtio is an abstraction layer over devices in a paravirtualized hypervisor. This article begins with an introduction to paravirtualization and emulated devices, and then explores the details of virtio. The focus is on the virtio framework from the 2. Linux is the hypervisor playground. As my article on Linux as a hypervisor showed, Linux offers a variety of hypervisor solutions with different attributes and advantages. Having these different hypervisor solutions on Linux can tax the operating system based on their independent needs. One of the taxes is virtualization of devices. Rather than have a variety of device emulation mechanisms for network, block, and other drivers , virtio provides a common front end for these device emulations to standardize the interface and increase the reuse of code across the platforms. In full virtualization, the guest operating system runs on top of a hypervisor that sits on the bare metal. The guest is unaware that it is being virtualized and requires no changes to work in this configuration. Conversely, in paravirtualization, the guest operating system is not only aware that it is running on a hypervisor but includes code to make guest-to-hypervisor transitions more efficient see Figure 1. In the full virtualization scheme, the hypervisor must emulate device hardware, which is emulating at the lowest level of the conversation for example, to a network driver. In the paravirtualization scheme, the guest and the hypervisor can work cooperatively to make this emulation efficient. Device emulation in full virtualization and paravirtualization environments View image at full size Hardware continues to change with virtualization. New processors incorporate advanced instructions to make guest operating systems and hypervisor transitions more efficient. Virtio alternatives virtio is not entirely alone in this space. Xen provides paravirtualized device drivers, and VMware provides what are called Guest Tools. But in traditional full virtualization environments, the hypervisor must trap these requests, and then emulate the behaviors of real hardware. Although doing so provides the greatest flexibility namely, running an unmodified operating system , it does introduce inefficiency see the left side of Figure 1. The right side of Figure 1 shows the paravirtualization case. The hypervisor implements the back-end drivers for the particular device emulation. These front-end and back-end drivers are where virtio comes in, providing a standardized interface for the development of emulated device access to propagate code reuse and increase efficiency. An abstraction for Linux guests From the previous section, you can see that virtio is an abstraction for a set of common emulated devices in a paravirtualized hypervisor. This design allows the hypervisor to export a common set of emulated devices and make them available through a common application programming interface API. Figure 2 illustrates why this is important. With paravirtualized hypervisors, the guests implement a common set of interfaces, with the particular device emulation behind a set of back-end drivers. The back-end drivers need not be common as long as they implement the required behaviors of the front end. QEMU is a system emulator that, in addition to providing a guest operating system virtualization platform, provides emulation of an entire system PCI host controller, disk, network, video hardware, USB controller, and other hardware elements. The virtio API relies on a simple buffer abstraction to encapsulate the command and data needs of the guest. Virtio architecture In addition to the front-end drivers implemented in the guest operating system and the back-end drivers implemented in the hypervisor , virtio defines two layers to support guest-to-hypervisor communication. At the top level called virtio is the virtual queue interface that conceptually attaches front-end drivers to back-end drivers. Drivers can use zero or more queues, depending on their need. For example, the virtio network driver uses two virtual queues one for receive and one for transmit , where the virtio block driver uses only one. Virtual queues, being virtual, are actually implemented as rings to traverse the guest-to-hypervisor transition. But this could be implemented any way, as long as both the guest and hypervisor implement it in the same way. High-level architecture of the virtio framework View image at full size As shown in Figure 3 , five front-end drivers are listed for block devices such as disks , network devices, PCI emulation, a balloon driver for dynamically managing guest memory usage , and a console driver. Each front-end driver has a corresponding back-end driver in the hypervisor. Concept hierarchy From the perspective of the guest, an object hierarchy is defined as shown in Figure 4. This object is cached with the management data for the device in a driver-dependent way. Virtio buffers Guest front-end drivers communicate with hypervisor back-end drivers through buffers. For example, you could provide three buffers, with the first representing a Read request and the subsequent two buffers representing the response data. Internally, this configuration is represented as a scatter-gather list with each entry in the list representing an address and a length. The virtqueue supports its own API consisting of five functions. This request is in the form of the scatter-gather list discussed previously. For best performance, the guest should load as many buffers as possible onto the virtqueue before notifying through kick. The guest can poll simply by calling this function or wait for notification through the provided virtqueue callback function. The format, order, and contents of the buffers are meaningful only to the front-end and back-end drivers. The internal transport rings in the current implementation move only buffers and have no knowledge of their internal representation. Example virtio drivers You can find the source to the various front-end drivers within the. The virtio network driver can be found in. You can learn more about this work in the Related topics section. You can exercise this paravirtualization infrastructure today in the Linux kernel. All you need is a kernel to act as the hypervisor, a guest kernel, and QEMU for device emulation. Both of these virtualization solutions support virtio along with QEMU for system emulation and libvirt for virtualization management. Going further Although you may never develop front-end or back-end drivers for virtio, it implements an interesting architecture and is worth understanding in more detail. Linux continues to prove itself as a production hypervisor and a research platform for new virtualization technologies. Xen also includes the concept of paravirtualized drivers. Paravirtual Windows Drivers discusses both paravirtualization and also hardware-assisted virtualization HVM in particular. One of the most important benefits of virtio is performance in paravirtualized environments. This blog post from btm. This article touched on the intersection of libvirt an open virtualization API and the virtio framework. The libvirt wiki shows how to specify virtio devices in libvirt. This article discussed two hypervisor solutions that take advantage of the virtio framework: One interesting use of virtio was the development of shared-memory message passing to allow VMs to communicate with one another through the hypervisor, as described in this paper from SpringerLink.

Chapter 2 : Applications of artificial intelligence - Wikipedia

AI Application Programming covers both the theory and the practical applications to teach devel It covers a wide variety of techniques currently defined as "AI" and shows how they can be useful in practical, everyday applications.

For example, the University of Southern California launched the Center for Artificial Intelligence in Society, with the goal of using AI to address socially relevant problems such as homelessness. At Stanford, researchers are using AI to analyze satellite images to identify which areas have the highest poverty levels. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed. The AOD has use for artificial intelligence for surrogate operators for combat and training simulators, mission management aids, support systems for tactical decision making, and post processing of the simulator data into symbolic summaries. Airplane simulators are using artificial intelligence in order to process the data taken from simulated flights. Other than simulated flying, there is also simulated aircraft warfare. The computers are able to come up with the best success scenarios in these situations. The computers can also create strategies based on the placement, size, speed and strength of the forces and counter forces. Pilots may be given assistance in the air during combat by computers. The artificial intelligent programs can sort the information and provide the pilot with the best possible maneuvers, not to mention getting rid of certain maneuvers that would be impossible for a human being to perform. Multiple aircraft are needed to get good approximations for some calculations so computer simulated pilots are used to gather data. It is a rule based expert system put together by collecting information from TF documents and the expert advice from mechanics that work on the TF The performance system was also used to replace specialized workers. The system allowed the regular workers to communicate with the system and avoid mistakes, miscalculations, or having to speak to one of the specialized workers. The AOD also uses artificial intelligence in speech recognition software. The programs that incorporate the speech software must be trained, which means they use neural networks. The program used, the Verbex , is still a very early program that has plenty of room for improvement. The improvements are imperative because ATCs use very specific dialog and the software needs to be able to communicate correctly and promptly every time. The Artificial Intelligence supported Design of Aircraft, [4] or AIDA, is used to help designers in the process of creating conceptual designs of aircraft. This program allows the designers to focus more on the design itself and less on the design process. The software also allows the user to focus less on the software tools. The AIDA uses rule based systems to compute its data. This is a diagram of the arrangement of the AIDA modules. Although simple, the program is proving effective. The neural network used in the software proved to be effective and marked a triumph for artificial intelligence. The Integrated Vehicle Health Management system, also used by NASA, on board an aircraft must process and interpret data taken from the various sensors on the aircraft. The system needs to be able to determine the structural integrity of the aircraft. The system also needs to implement protocols in case of any damage taken the vehicle. Many of their inventions have been adopted by mainstream computer science and are no longer considered a part of AI. AI can be used to potentially determine the developer of anonymous binaries. There have also been a rise of intelligent tutoring systems , or ITS, in higher education. Data sets collected from these large scale online learning systems have also enabled learning analytics, which will be used to improve the quality of learning at scale. Examples of how learning analytics can be used to improve the quality of learning include predicting which students are at risk of failure and analyzing student engagement. Automated trading systems are typically used by large institutional investors. Its wide range of functionalities includes the use of natural language processing to read text such as news, broker reports, and social media feeds. It then gauges the sentiment on the companies mentioned and assigns a score. Its machine learning systems mine through hoards of data on the web and assess correlations between world events and their impact on asset prices. For example, Digit is an app powered by artificial intelligence that automatically helps consumers optimize their spending and savings based on their own personal habits and goals. The app can analyze factors such as monthly income, current balance, and spending habits, then make its own decisions and transfer money to the savings account. AI, an upcoming startup in San Francisco, builds agents that analyze data that a consumer would leave behind, from Smartphone check-ins to tweets, to inform the consumer about their spending behavior. Robo-advisors provide financial advice and portfolio management with minimal human intervention. This class of financial advisers work based on algorithms built to automatically develop a financial portfolio according to the investment goals and risk tolerance of the clients. It can adjust to real-time changes in the market and accordingly calibrate the portfolio. Their technology will be licensed to banks for them to leverage for their underwriting processes as well. This platform utilizes machine learning to analyze tens of thousands traditional and nontraditional variables from purchase transactions to how a customer fills out a form used in the credit industry to score borrowers. The platform is particularly useful to assign credit scores to those with limited credit histories, such as millennials. It has simplified the process for both recruiters and job seekers i. According to Raj Mukherjee from Indeed. AI-powered engine streamlines the complexity of job hunting by operating information on job skills, salaries, and user tendencies, matching people to the most relevant positions. Machine intelligence calculates what wages would be appropriate for a particular job, pulls and highlights resume information for recruiters using natural language processing, which extracts relevant words and phrases from text using specialized software. Another application is an AI resume builder which requires 5 minutes to compile a CV as opposed to spending hours doing the same job. In the AI age chatbots assist website visitors and solve daily workflows. Moreover, the research proves automation will displace between and million employees. Robots have proven effective in jobs that are very repetitive which may lead to mistakes or accidents due to a lapse in concentration and other jobs which humans may find degrading. In the automotive industry , a sector with particularly high degree of automation, Japan had the highest density of industrial robots in the world:

Chapter 3 : Artificial Intelligence: A Systems Approach

M. Tim Jones covers pretty much every useful category of AI programming: neural networks, expert systems, fuzzy logic, genetic algorithms, rules-based systems, and more. While a few of the sample applications are theoretical, most are very practical, and drawn straight from the real world.

The file descriptor can represent a file, a socket, or even a pipe. The return value is zero on success or -1 on error, where errno is defined. To perform a read, the application must initialize the aiocb structure. At this point, your request has either succeeded or failed. Building with the AIO interface You can find the function prototypes and other necessary symbolics in the aio. When building an application that uses this interface, you must use the POSIX real-time extensions library librt. Note the similarities to reading from the file with the standard library functions. In a typical read call, the offset is maintained for you in the file descriptor context. For each read, the offset is updated so that subsequent reads address the next block of data. In a standard read call, the return status is provided upon return of the function. This function has the following prototype: Its function prototype is: This is similar to the read system call, but one behavior difference is worth noting. Recall that the offset to be used is important with the read call. A list of aiocb references is provided. If any of them complete, the call returns with 0. Otherwise, -1 is returned, indicating an error occurred. To cancel all requests for a given file descriptor, provide that file descriptor and a NULL reference for aiocbp. The list is a list of aiocb references, with the maximum number of elements defined by nent. This is illustrated in Listing 4. In this paradigm, the application defines a signal handler that is invoked when a specified signal occurs. The application then specifies that an asynchronous request will raise a signal when the request has completed. As part of the signal context, the particular aiocb request is provided to keep track of multiple potentially outstanding requests. Listing 5 demonstrates this notification method. In this way, when completion of one transfer has completed, you immediately start the next. Asynchronous notification with callbacks An alternative notification mechanism is the system callback. Instead of raising a signal for notification, this mechanism calls a function in user-space for notification. You initialize the aiocb reference into the sigevent structure to uniquely identify the particular request being completed; see Listing 6. You then specify the particular notification handler and load the context to be passed to the handler in this case, a reference to the aiocb request itself. In the handler, you simply cast the incoming sigval pointer and use the AIO functions to validate the completion of the request. The maximum is commonly 64KB, which is adequate for most applications. In the Design Notes for the 2. With IBM trial software , available for download directly from developerWorks, build your next development project on Linux.

Chapter 4 : Virtio: An I/O virtualization framework for Linux

Artificial intelligence Sortware engineer Jones demystifies techniques associated with artificial intelligence and shows how they can be useful in everyday applications. The book covers techniques including statistical algorithms, symbolic methods, evolutionary systems, and optimization methods.

Tim Jones , product architect and engineering author. On this page are book descriptions with errata and other online publications relating to virtualization, Linux, Linux internals, programming, network protocols, embedded development, and artificial intelligence. Follow me on Twitter or LinkedIn. The following books cover networking protocols, Sockets programming, artificial intelligence, and Linux user-space programming. Split into 5 distinct parts, the book covers GNU tools, topics in application development, shells and scripting, debugging and hardening, and introductory optics including the fundamentals of virtualization. New material in this edition includes exploration of advanced APIs, memory debugging, scripting with Python and Ruby, source control, and more. See more at Course Technology. This new AI text, focused on the academic market provides grounding in the theory and practice of the various AI algorithms and methods. The text also follows a "systems" approach, discussing the application of the algorithms in real-world environments. See more at Infinity Science Press. This second edition of AI Application Programming builds on the first edition with the addition of five new chapters classifier systems, natural language processing, particle swarm optimization, A-Star pathfinding and reinforcement learning. The text also includes updates for erratum for the first and second printing, additions to the backpropagation learning chapter for batch-updating and numerous other additions. Another review at a Sun blog. Artificial Intelligence " at Graceland University. My goal with this book was to illustrate that AI can be used as a tool to solve practical problems. Some of the AI algorithms that I discuss in the book include genetic algorithms, neural networks, rules-based systems, clustering algorithms, ant algorithms, fuzzy logic, hidden markov models, simulated annealing and intelligent agents. Sample applications include a personalization engine, a rules-based reasoning system, a character trainer for game AI, a Web-based news agent, a genetic code optimizer, an artificial life simulation, and others. More reviews are available through Amazon. This book is currently being used as a text book in Dr. An erratum is available for this book here. The Sockets API is useful not only in traditional high-level language environments such as C , but also in any worthwhile scripting language. This book is split into three parts. Part one details the Sockets API. Part two discusses the Sockets API in each of the languages explored. Finally, in Part three, software patterns are demonstrated for each of the languages, illustrating the strengths and weaknesses of the APIs for each language. An errata item is noted here. In this book, I explore a variety of application layer protocols and then illustrate how they can be used in embedded or non-embedded systems. For example, how can HTTP be used to control embedded devices, or how can SMTP be used for communication of data or control from remote embedded systems? This book explores these topics, discusses all protocols in detail with open source implementations , and demonstrates an application in a practical domain. The book has been translated into Chinese, the link is available here. The book can be purchased in India through Firewall Media here. However, as we know from traditional scheduling, not all algorithms are the same, and efficiency is workload and cluster dependent. Get to know Hadoop scheduling, and explore two of the algorithms available today: Spark is implemented in and exploits the Scala language, which provides a unique environment for data processing. Get to know the Spark approach for cluster computing and its differences from Hadoop. Virtualization as an abstraction from the physical platform creates a new opportunity for HA. In this article, explore some of the practical approaches to cloud-based HA, including stateless failover and the more useful stateful failover. Also, discover the various open source software components at play in HA systems. On average, Twitter users generate million tweets per day on a variety of topics. This article introduces you to data mining and demonstrates the concept with the object-oriented Ruby language. A physics engine is a software component that provides a simulation of a physical system. This simulation can include soft- and rigid-body dynamics, fluid dynamics, and collision detection. The open source community has a number of useful physics engines operating in the 2D and 3D domains targeted to games and simulations. This article introduces the use and basics of a physics engine and explores two options that exist: True advance, or just another language? Explore Ceylon and find out if this future VM language can find a place in enterprise software development. But VMs are simply an older concept of abstraction, a common method of abstracting one entity from another. This article explores two of the many newer open source VM technologies: Dalvik the VM core of the Android operating system and Parrot an open source VM technology for efficiently executing dynamic languages. The embedded domain has several useful applications for virtualization, including mobile handsets, security kernels, and concurrent embedded operating systems. Beyond the file system, Linux incorporates world-class NAS and SAN technologies, data protection, storage management, support for clouds, and solid-state storage. However, we can learn a lot from early computing history. Developing schedulers that provide suitable behavior for single-core machines to quad-core servers can be difficult. Luckily, the Linux Scheduler Simulator LinSched hosts your Linux scheduler in user space for scheduler prototyping while modeling arbitrary hardware targets to validate your scheduler across a spectrum of topologies. Learn about LinSched and how to experiment with your scheduler for Linux. Learn about platform emulation using Bochs and its approach to hardware emulation. It incorporates variable block sizes, compression, encryption, de-duplication, snapshots, clones, and as the name implies support for massive capacities. Learn about image processing and how to convert your Processing application into a Java applet suitable for the web, and explore an optimization algorithm that lends itself well to visualization. Further, the interface abstracts the location of the storage such that it is irrelevant whether the storage is local or remote or hybrid. Cloud storage infrastructures introduce new architectures that support varying levels of service over a potentially large set of users and geographically distributed storage capacity. An introduction to the language and environment " IBM developerWorks, November Building graphical applications and applications that present complex data can be difficult. Although many graphical libraries exist, they cater to advanced users or present non-trivial APIs. The Processing language and environment solves this problem by creating a portable environment and language for graphical presentation. Processing makes it simple to build applications that present static data, dynamic data such as animations , or interactive data. This first article in the series explores building applications for visualization in particular, simulations for life sciences. Explore the ideas behind distributed file systems and in particular, recent advances in NFS. One of the more recent areas is virtual networking. Early implementations of platform virtualization created virtual NICs, but today, larger portions of the network are being virtualized, such as switches that support communication among VMs on a server or distributed among servers. Explore the ideas behind virtual networking, with a focus on NIC and switch virtualization. But have you considered the process and underlying implementation of kernel logging? Explore the entire process of kernel logging, from printk to insertion into the user space log file. And cloud computing justifies all the focus. Explore the ideas behind virtual address spaces and the kernel APIs for data movement to and from user space, and learn some of the other mapping techniques used to map memory. This service mirrors an entire block device to another networked host during run time, permitting the development of high-availability clusters for block data. Application development " IBM developerWorks, July With configuration, installation, and the use of Hadoop in single- and multi-node architectures under your belt, you can now turn to the task of developing applications within the Hadoop infrastructure. This final article in the series explores the Hadoop APIs and data flow and demonstrates their use with a simple mapper and reducer application. The storage ecosystem is a great example, where change is not only occurring, but at all levels from the individual storage devices, to the baseline services and front-end protocols that are used to manipulate our growing masses of data. This article continues with a more advanced setup that uses multiple nodes for parallel processing. It demonstrates the various node types required for multinode clusters and explores MapReduce functionality in a parallel environment. This article also digs into the management aspects of Hadoop both command line and Web based. In this context, multiple operating system and application sets are virtualized on a single server, allowing it to be more efficiently and cost effectively used. While this drives much of the innovation and revenue around virtualization today, there are a multitude of virtualization schemes addressing a spectrum of applications. Learn how to install and configure a single-node Hadoop cluster, and delve into the MapReduce application. Finally, discover ways to monitor and manage Hadoop using its core Web interfaces. Explore the architecture of Ceph and learn how it provides fault tolerance and simplifies the management of massive amounts of data. Indeed, Linux is a unique operating system in its breadth of virtualization solutions that are available. KSM allows the hypervisor to increase the number of concurrent virtual machines by consolidating identical memory pages. Explore the ideas behind KSM such as storage de-duplication , its implementation, and how you manage it. Timers and Lists in the 2. Discover these APIs, and learn how to develop kernel applications with timers and lists. Explore the anatomy of the cloud, its architecture, and the open source technologies used to build these dynamic and scalable computing and storage platforms. Tasklets and work queues implement deferrable functionality and replace the older bottom-half mechanism for drivers. This article explores the use of tasklets and work queues in the kernel and shows you how to build deferrable functions with these APIs.

Chapter 5 : M. Tim Jones (Author of AI Application Programming)

M. Tim Jones is an embedded software architect and the author of numerous books, including AI Application Programming, Second Edition (Charles River Media), BSD Sockets Programming from a Multilanguage Perspective (Charles River Media), Artifi cial Intelligence: A Systems Approach, and many articles on a variety of technical subjects.

Chapter 6 : AI Application Programming by M. Tim Jones

M. Tim Jones is the author of AI Application Programming (avg rating, 28 ratings, 2 reviews, published), Artificial Intelligence (avg rati.

Chapter 7 : Microsoft Research â€“ Emerging Technology, Computer, and Software Research

Ai application programming - m. tim jones - google books AI Application Programming covers both the theory and the practical applications to teach developers Charles River Media, - Computers - pages.

Chapter 8 : AI | | AI Application Programming - M. Tim calendrierdelascience.com | | AI Game Developmen

If you are searched for the book AI Application Programming (Programming Series) (Charles River Media Programming) by M. Tim Jones in pdf format, then you have come on to the right site.

Chapter 9 : Boost application performance using asynchronous I/O

calendrierdelascience.com for Web Designers - download pdf or read online. Educating ASP. web in a non-linear layout that artistic thinkers can simply clutch and comprehend with no the common programming jargon. offers transparent and concise, hands-on, real-world examples correct from the start of the ebook.