International Academic Institute for Science and Technology

International Academic Journal of Science and Engineering Vol. 2, No. 2, 2015, pp. 258-264.
ISSN 2454-3896

International Academic Journal of Science and Engineering

www.iaiest.com

A Critical Review of Linux Memory Management Recent Articles

S. Abdolreza Pishvaei[a] , S.Talayeh Tabibi[b]

[a] *Assistant Professor of Software Engineering, Department of Software Engineering, Islamic Azad University, Karaj Branch.*

[b] *Ph.D. Student of Software Engineering, Department of Software Engineering, Islamic Azad University, Karaj Branch.*

Abstract

Open source software has been developed for several decades. Excellent design and extraordinary performance, coupled with the strong support from IBM, INTEL, CA, ORACLE and other internationally renowned companies, Linux has gradually become one of the major operating systems. This article is a critical research on Linux memory management. In this work, recent researches on challenges of Linux memory management and has been briefly described. Based on our research, 16 documents in the field of Linux memory management has been collected. All of these results in the form of author or authors, publication year, type of publication, specific field of the research, important results and future works, has been critically reviewed and compared. Linux memory management has many challenges and problems, especially, in embedded systems with limited memory and resources. Recently, Linux operating system has become very popular in developing embedded systems and wireless sensor networks; therefore, it is necessary to solve its challenges.

Keywords: Linux, Memory Management, Embedded Systems, Handling Low Memory, Virtual Memory Management

Introduction:

So far, open source software has been developed for several decades. Excellent design and extraordinary performance, coupled with the strong support from IBM, INTEL, CA, ORACLE and other internationally renowned companies, Linux has gradually become one of the major operating systems. [1]
Memory management is very important in the operating system. For most Windows applications, it is almost inevitable to operate and manage the memory. But there are differences in memory management between Windows and Linux. For example, Windows memory management emphasizes the two-tier structure: virtual memory management and heap management. While in Linux memory management is not like that. And system calls of heap management in Linux cannot use to implements Windows functions which called "fixed address redistribution of the heap". [1]
A major concern of almost any operating system is the methods and mechanisms it uses to manage its resources. Physical memory and, in many cases, how it is mapped to virtual memory is one resource that almost all aspects of an operating system and its applications require. Hence, the Memory
Manager (MM) and Virtual Memory Managers (VMM) play a critical role in the systems performance.

Due to its critical nature, it is important to understand all aspects of how the MM interacts with the operating system and the different types of processes that run on the machine. [2]
Historically, the memory manager was a core subsystem but not tightly integrated with other portions of the operating system. In recent times, particularly in monolithic kernels, the trend has been to tightly integrate the resource management of IO and filesystems with the memory manager. The integration is especially noticeable in Linux as portions of the IO manager are almost indistinguishable from the memory manager. [2]

This tighter integration implies that the overall performance of the operating system is increasingly dependent on the performance of the memory manager. Despite this, it is difficult to measure performance of just the memory manager.
Benchmarks tend to measure a secondary effect and extrapolate the performance of the memory manager based on the results. This makes it very difficult to determine what aspect of a memory manager may be resulting in poor performance. [2]
For many years, the majority of small-scale embedded systems have been implemented without an operating system (OS). Traditionally, such embedded systems have performed multitasking using various programming techniques such as a polled loop, co-routines, and interrupt-driven scheduling. As embedded systems grow in size and complexity, however, an OS has become essential to simplify their design. Today, any serious embedded software employs a real-time multitasking executive, or a fully fledged OS such as embedded Linux. The OS provides an effective development and execution environment for application programmers, enabling them to develop target systems with ease and efficiency.
During the design of an embedded system, small variations in a few parameters may exert significant effects on its performance and cost. It is therefore important to explore the design space carefully, and to examine different implementation choices, in order to determine appropriate trade-offs among conflicting objectives.
Most modern computer systems now support virtual memory, which provides a distinct virtual address space for each process and also offers hardware-assisted memory protection for multi-programming.
A virtual memory system is maintained by the memory management subsystem of the OS, and requires hardware support from an MMU (memory management unit) in the CPU. The presence of the MMU greatly helps the OS to control memory use at run-time, which improves functionality, productivity, and maintainability.

However, a significant increase in computational overhead is inevitable with an MMU, as resources are required to manage the virtual memory system as well as the MMU hardware, which of course also consumes power and incurs manufacturing cost. In this sense, how to configure the memory management subsystem in an OS becomes one of the important design parameters for embedded systems. [3]

Methods

This article is a critical review on Linux memory management. In this work, recent researches on challenges of Linux memory management and has been briefly described.

There are several challenges in Linux memory management such as handling low memory situations, robust memory acquisition, memory leak detection, partitioning scheduler, memory segmentation, secure memory management and etc.

Linux is a very applicable operating system which even runs on very small boards and systems like embedded systems. Therefore, Linux faces of embedded systems' problems such low power, security and so on. On the other hand, compatibility with other popular operating systems especially in the field of memory management is a big challenge for Linux developers.

In order to solve these challenges, many works have been done in various fields which some of them will be mentioned here.

Results

Based on our research, 16 documents in the field of Linux memory management has been collected. All of these results in the form of author or authors, publication year, type of publication, specific field of the research, important results and future works, has been summarized in table 1.

Table1: Summary Specifications of Reviewed Articles

Reference number	Author/authors	Title	Year	Type
1	Li, Rui. Yang, Nanjun. Ma, Shilong.	An Approach of Windows Memory Management Simulation on Linux	2012	World Congress IEEE
2	Gorman, Mel. Healy, Patrik.	Measuring the Impact of the Linux Memory Manager	2005	Conference
3	Park, Sangsoo. Shin,Heonshik.	Performance Evaluation of Memory Management Configurations in Linux for an OS-Level Design Space Exploration	2007	Springer
4	Kocoloski, Brian. Lange, John.	Lightweight Memory Management for High Performance Applications in Consolidated Environments	2015	IEEE
5	Wang, Li-xin. Kang, Jing.	MMAP System Transfer in Linux Virtual Memory Management	2009	Workshop, IEEE
6	Schirmeier,Horst. Neuhalfen, Jens. Korb, Ingo. Spinczyk, Olaf. and Engel,Michael.	RAMpage: Graceful Degradation Management for Memory Errors in Commodity Linux Servers	2011	IEEE, International Symposium

7	B. Patare. Prashant, Govindan, V. K.	Efficient Handling of Low Memory Situations in Linux	2015	International Journal of Engineering Research & Technology
8	Stüttgen, Johannes. Cohen, Michael.	Robust Linux memory acquisition with minimal target impact	2014	Digital Investigation, ELSEVIER
9	Lim, Yunjae. Nam, Young Jin. Yoo, Geel-Sang. and Seo, Dae-Wha.	Automatic Partitioning Technique for Flash Memory on Linux-Based Embedded Systems	2007	Springer
10	Beneder, Roman. Glatz, Bernd. Horauer, Martin. Rauscher,Thomas.	Memory Leak Detection Runtime-Service for Embedded Linux Devices	2014	IEEE Emerging Technology and Factory Automation
11	Kown,Cheolsoon. Kim, Duksoo. Joe, Hyunwoo. Kim, Hyungshin.	Linux-based Memory Efficient ARINC 653 Partition Scheduler	2014	IEEE Emerging Technology and Factory Automation
12	Miyazaki,Kiyohito. Nomura,Yoshinari. Taniguchi, Hideo.	Memory Segmentation and Transfer in Mint Operating System	2013	International Conference, IEEE
13	Gottscho, Mark. et al.	ViPZonE: Hardware Power Variability-Aware Virtual Memory Management for Energy Savings	2014	IEEE TRANSACTIONS ON COMPUTERS
14	Lin, Mingwei. Chen, Shuyu.	Flash-aware Linux Swap System for Portable Consumer Electronics	2012	IEEE Transactions on Consumer Electronics
15	Chen, Xian. et al.	SEMMA: Secure Efficient Memory Management Approach in Virtual Environment	2013	International Conference
16	Liu, Lei. et al.	Going Vertical in Memory Management: Handling Multiplicity by Multi-policy	2014	IEEE

Table 2 show the frequency of articles based on type of publication and table 3 represents the Frequency of Articles based on Type of Journals and Conferences.

Table 2: Frequency of Articles based on Type of Publication

Type of Document Publication	**Frequency**	**Percentage**
Published Articles in Journals	10	60
Articles in Conferences	6	40
Total	16	100

Table 3: Frequency of Articles based on Type of Journals and Conferences

Type of Journals and Conferences	Frequency	Percentage
IEEE	9	58
Springer	2	12
Elsevier	1	6
Other Journals and Conferences	4	25
Total	16	100

[1] in 2012 represented an approach of Windows Memory Management Simulation on Linux, to build an middle layer which between application and operating system to shield the differences between the underlying operating system for the upper layer application. In [4] HPMMAP provides lightweight memory performance transparently to HPC applications by bypassing Linux's memory management layer. Using HPMMAP, HPC applications achieve consistent performance while the same local compute nodes execute competing workloads likely to be found in HPC clusters and "in-situ" workload deployments. This approach is dynamically configurable at runtime, and requires no resources when not in use. They show that HPMMAP can decrease variance and reduce application runtime by up to 50% when executing a co-located competing commodity workload. MMAP [5], system transfer has a very important effect in Linux virtual memory management, but its work process is not well-known. The paper is focus on this point, analyses system transfer functions, and offer system flow chart at last. RAMpage: Graceful Degradation Management for Memory Errors in Commodity Linux Servers in 2011 have focused on memory errors in Linux servers [6], proposed to increase system reliability, an online memory testing infrastructure for commodity x86-64-based Linux servers, which is capable of efficiently detecting memory errors and which provides graceful degradation by withdrawing affected memory pages from further use. [7], has worked on Efficient Handling of Low Memory Situations in Linux and proposed an approach where the system sends signals to currently running processes so that the processes by themselves can free some memory. This proposed approach can be used along with the approach of considering the application's past usage history. It is also suggesting some better ways for controlling the OOM killer management for some of the processes. [8], presents a novel technique to safely load a precompiled kernel module for acquisition on a wide range of Linux kernel versions and configuration. Our technique injects a minimal acquisition module (parasite) into another valid kernel module (host) already found on the target system. The resulting combined module is then relinked in such a way as to grant code execution and control over vital data structures to the acquisition code, whilst the host module remains dormant during runtime. [9], proposes an automatic partitioning technique for flash memory in Linux-based mobile embedded systems. The proposed technique creates a set of flash memory partitions whose sizes are exactly the same as the software modules stored in the flash memory, resultantly maximizing the space for user applications. The performance evaluations reveal that the proposed technique not only guarantees about 10% more space for the user area in the flash memory, but also increases the speed of the software module updates by eight times, as compared with a manual update. [10], presents a mechanism to detect memory leaks in embedded systems software at runtime that can be used to counter software aging. In particular, it presents and compares two measurement based algorithms to identify memory leaks on Embedded Linux devices at runtime. Both algorithms have been implemented and evaluated using an industrial room controller targeting building automation. Based on this approach we were able to identify an existing memory leak we were unaware of beforehand. [11], introduces a Linux-based memory efficient ARINC 653 partition scheduler, which employs fixed-priority

preemptive scheduling algorithm for partitions scheduling. The implemented partition scheduler has small memory footprint for each partition and produces low partition switching overhead. The prototype was executed on a LEON4 processor, which is the European Space Agency (ESA) Next Generation Multicore Processor (NGMP) in the space sector. [12], developed the Mint operating system, which enables running multiple Linux instances on a multi-core processor. Mint is different from traditional virtualization technologies in that Mint can achieve performance isolation between Linux instances by segmenting computer resources. It describes the requirements for segmentation and transfer of physical memory and provides a solution**.** [13], ViPZonE: Hardware Power Variability Aware Virtual Memory Management for Energy Savings presents ViPZonE, a memory management solution in conjunction with application annotations that opportunistically performs memory allocations to reduce DRAM energy. ViPZonE's components consist of a physical address space with DIMM-aware zones, a modified page allocation routine, and a new virtual memory system call for dynamic allocations from userspace. [14], has worked on Flash-aware Linux Swap System for Portable Consumer Electronics in 2012. It proposed a flash-aware Linux swap system, called FLSS, which adopts Linux kernel 2.6 to manage flash memory-based swap space directly without FTL. It introduces: 1) a partial block alignment scheme to perform an efficient swap-in read-ahead algorithm, 2) a swap-aware victim block selection method and the redefined concept of hot page and cold page to design a swap-aware garbage collection policy called SACATA, and 3) the notion of overage as well as the notion of frozen applied to blocks to exploit a wear leveling-aware block management scheme. [15], SEMMA: Secure Efficient Memory Management Approach in Virtual Environment, propose a group-based secure page sharing model (or GSKSM), in which we consider both VM processes and normal processes. They have successfully implemented it in Linux kernel 3.6.6, and the experiment results show it works well with negligible overhead. [16], studies the performance of these policies for over 2000 workloads and correlate the results with application characteristics via a data mining approach. Based on this correlation they derive several practical memory allocation rules that they integrate into a unified multi-policy framework to guide resources partitioning and coalescing for dynamic and diverse multi programmed/ threaded workloads. They implement our approach in Linux kernel 2.6.32 as a restructured page indexing system plus a series of kernel modules.

Discussion and Conclusion

In this study, we have reviewed recent articles in the field of Linux memory management. Some important challenges of this field like low memory management, memory leaking, etc. and proposed solutions have been described briefly.

As we mentioned before, Linux memory management has many challenges and problems, especially, in embedded systems with limited memory and resources. Recently, Linux operating system has become very popular in developing embedded systems and wireless sensor networks; therefore, it is necessary to solve its challenges.

For this article, 16 researches have been studied; most of them haven't focused on Linux memory management in embedded systems especially, in wireless sensor networks. They have evaluated their solution on common platforms not on small embedded systems.

Since Linux is a very applicable and useful operating system in wireless sensor networks, memory management could be important in this field and working on it is inevitable.

On the other hand, incompatibility between different operating systems due to different design concepts, have different concrete realization which leads to correspond in semantic incomplete. The absence of corresponding implementation in semantic between different operating systems could be an obstacle to Windows application that need to migrate to Linux with no or little changes. Hence, there is a need to develop firmware to conquer this problem in future.

References:

B. Patare. Prashant, Govindan, V. K. "Efficient Handling of Low Memory Situations in Linux", International Journal of Engineering Research & Technology (IJERT), 2015.

Beneder, Roman. Glatz, Bernd. Horauer, Martin. Rauscher, Thomas. "Memory Leak Detection Runtime-Service for Embedded Linux Devices", IEEE Emerging Technology and Factory Automation (ETFA), 2014.

Chen, Xian. et al. "SEMMA: Secure Efficient Memory Management Approach in Virtual Environment", International Conference on Advanced Cloud and Big Data, 2013.

Gorman, Mel. Healy, Patrik. "Measuring the Impact of the Linux Memory Manager", LIBRE05 '05 Libre Software Meeting, Dijon France, 2005.

Gottscho, Mark. et al. "ViPZonE: Hardware Power Variability-Aware Virtual Memory Management for Energy Savings", IEEE TRANSACTIONS ON COMPUTERS, 2014.

Kocoloski, Brian. Lange, John. "Lightweight Memory Management for High Performance Applications in Consolidated Environments", TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS (TPDS), IEEE, 2015.

Kown, Cheolsoon. Kim, Duksoo. Joe, Hyunwoo. Kim, Hyungshin. "Linux-based Memory Efficient ARINC 653 Partition Scheduler", IEEE Emerging Technology and Factory Automation (ETFA), 2014.

Li, Rui. Yang, Nanjun. Ma, Shilong. "An Approach of Windows Memory Management Simulation on Linux", Third World Congress on Software Engineering, IEEE, 2012.

Lim, Yunjae. Nam, Young Jin. Yoo, Geel-Sang. and Seo, Dae-Wha. "Automatic Partitioning Technique for Flash Memory on Linux-Based Embedded Systems", Springer-Verlag Berlin Heidelberg, 2007.

Lin, Mingwei. Chen, Shuyu. "Flash-aware Linux Swap System for Portable Consumer Electronics", IEEE Transactions on Consumer Electronics,2012.

Liu, Lei. et al. "Going Vertical in Memory Management: Handling Multiplicity by Multi-policy", IEEE, Minneapolis, MN, USA, 2014.

Miyazaki, Kiyohito. Nomura, Yoshinari. Taniguchi, Hideo. "Memory Segmentation and Transfer in Mint Operating System", 16th International Conference on Network-Based Information Systems, IEEE, 2013.

Park, Sangsoo. Shin, Heonshik. "Performance Evaluation of Memory Management Configurations in Linux for an OS-Level Design Space Exploration" , Springer-Verlag Berlin Heidelberg, 2007.

Schirmeier, Horst. Neuhalfen, Jens. Korb, Ingo. Spinczyk, Olaf. and Engel, Michael. "RAMpage: Graceful Degradation Management for Memory Errors in Commodity Linux Servers", 17th IEEE Pacific Rim International Symposium on Dependable Computing, 2011.

Stüttgen, Johannes. Cohen, Michael. "Robust Linux memory acquisition with minimal target impact", Digital Investigation 11 (2014) S112–S119, ELSEVIER, 2014.

Wang, Li-xin. Kang, Jing. "MMAP System Transfer in Linux Virtual Memory Management", First International Workshop on Education Technology and Computer Science, IEEE, 2009.