XIAMEN UNIVERSITY MALAYSIA

Course Code	:	SOF201
Course Name	:	Operating System
Lecturer	:	Al-Daweri Muataz Salam Kadhim
Academic Session	:	2021/09
Assessment Title	:	Tanenbaum-Torvalds Debate Summary, Discussion, and Microkernel Virtualization
Submission Due Date	:	23/10/2021

Prepared by :

Student ID	Student Name
SWE2009510	Liu Aofan
SWE2009513	Su Yanyu
SWE2009864	Abubadi Mohammed Ali Abdulrahman Ali

Date Received : 10/23/2021

Feedback from Lecturer:

Mark:

XIAMEN UNIVERSITY MALAYSIA

Own Work Declaration

I/We hereby understand my/our work would be checked for plagiarism or other misconduct, and the softcopy would be saved for future comparison(s).

I/We hereby confirm that all the references or sources of citations have been correctly listed or presented and I/we clearly understand the serious consequence caused by any intentional or unintentional misconduct.

This work is not made on any work of other students (past or present), and it has not been submitted to any other courses or institutions before.

Signature:

Liu Aofan

Su Yanyu

Abubadi Mohammed Ali Abdulrahman Ali

Date:
23.10.2021

XIAMEN UNIVERSITY MALAYSIA

Tanenbaum-Torvalds Debate Summary, Discussion, and Microkernel Virtualization

Aofan Liu
SWE2009510
Software Engineering
Xiamen University Malaysia
SWE2009510@xmu.edu.my

Su Yanyu
SWE2009513
Software Engineering
Xiamen University Malaysia
SWE200951@xmu.edu.my

Abubadi Ali Abudulrahman
SWE2009864
Software Engineering
Xiamen University Malaysia
SWE2009864@xmu.edu.my

***Abstract*—In the history of operating system development, microkernels and monolithic kernels have been thought of as a key factor in design. In this assignment, we will try to give a summary on this debate, as well as the views of both parties, and finally state our opinions after discussion. In addition, we will also try to use microkernel based MINIX and try to do something with it.**

Keywords—microkernel, monolithic kernel, virtualization, MINIX, Tanenbaum-Torvalds Debate

I. Debate Summary

The **Tannenbaum-Torvalds debate** is a written debate that occurred between **Andrew Tannenbaum**, the author of the Minix, and **Linus Torvalds**, the author of the Linux kernel. In 1992, Tannenbaum caused a controversy on the Usenet newsgroup comp.os.minix, arguing that micro-cores were replacing monolithic kernels even in 1992.

Conflicts flared up, and with the publication of every post and the deliberately hostile exchange of messages between them (in other words, between Linux and Minix, between supporters of monolithic kernel and supporters of microkernel), conflicts grew progressively more sophisticated. The discussion clarified the advantages and disadvantages of each system's kernel architecture.

Monolithic kernel means that the whole kernel itself is an indivisible executable program running in **kernel mode** and requires a **page mechanism** to manage memory, which strengthens the **protection of memory space**. All the functions of the kernel are linked **inside** this program. The processes in the kernel can make a **system call** directly and **communicate** with other processes. Common Unix-like systems, such as UNIX, Linux and Solaris, are designed with a monolithic kernel. Supporters of microkernel believe that the cost of Linux hardware was expensive at that time, and it was not the future choice.

The microkernel itself only provides the most **basic** operating system functions, such as **process scheduling and message passing**, but other functions are independent of it. The microkernel uses a **modular** approach to design the operating system, each functional module is independent of each other, so that even if one of the functional modules crashes, the system will not be down. When the system is running, you can **dynamically** enable or disable the corresponding modules as needed to release computer resources. In theory, this design is **more portable**, but it will **lose efficiency** due to the **overhead** generated by the **message passing mechanism**.

The debate between two operating system academic experts undoubtedly brought valuable inspiration to the methodology and thinking mode of the academic circles at that time. From the perspective of contemporary people, the microkernel architecture is excellent in **aesthetics**. After the emergence of the network, the microkernel system can adapt to the modification of hardware design more quickly and smoothly. In contrast, the monolithic kernel is more practical, and its running performance and **compatibility** at user level are also better. Generally speaking, the trade-off between microkernel and monolithic kernel depends entirely on the **operating environment** and the **purpose of use**.

II. Debate Discussion

A. Liu Aofan's View

The processes of the monolithic kernel all run in the **kernel mode** and are in the same **address space**, which is **simple** and **high-performance**. Moreover, the free and open-sourced monolithic based Linux is adopted in all areas of the 21st century.

As for the microkernel design, it will be **easier to migrate** to other architectures. At the same time, the microkernel is often just a **message forwarding station**. When a module wants to send a message to another module, the message is forwarded through the kernel. This way helps to achieve **isolation between modules**, but also causes a **loss of efficiency**.

From the above description, the microkernel operating system seems to be the general trend in the future. However, this theoretically perfect design did not win the market over time.

The monolithic kernel is now the mainstream, but it does not mean that the microkernel is a bad design. With the development of time, the design of the monolithic kernel is gradually adopting the excellent features of the microkernel. Linux has absorbed the essence of the microkernel: its **modular design**, **preemptive kernel**, support for **kernel threads**, and the **ability to dynamically load kernel modules**. Not only that, but Linux also avoids the performance loss of its micro-kernel design, allowing everything to run in the kernel state, calling functions directly, without message passing. So far, Linux is a **modular**, **multi-threaded** and **schedulable operating system** by the kernel itself.

SOF201 Operating System
Lecturer: Al-Daweri Muataz Salam Kadhim

The essence of the battle between microkernel and monolithic kernels is trying to find method to **improve efficiency** of kernel, so the boundary between the two is becoming blurred over time. **Pragmatism** once again prevailed.

B. Su Yanyu's View

I think the mainstream of Operation System might be monolithic kernel and hybrid kernel.

During the debate between Tanenbaum and Torvalds, they exchanged views on the operation principle (design concept), portability, running performance and positioning advantages of micro kernel and monolithic kernel. What is certain is that these two kernel architectures are not perfect, and they are quite different in **internal design**, **performance** and **compatibility**.

In the kernel layer of monolithic architecture, the **coupling degree** of each functional module is extremely high, and it can interact directly through method calls, thus having the advantage of **high performance**. But at the same time, the failure of one module may affect the use of other modules. On the contrary, the microkernel defines a mechanism of **inter-process communication**, and completes related service processes through **"messages"**. Lower coupling degree sacrifices certain performance but ensures the stability, scalability and **portability** of modules.

Therefore, it may be a good choice to combine the features of the two kernel architectures. Hybrid kernel is similar to micro kernel, but it puts some functional modules running in the user layer back into the kernel, while modules that are not commonly used or require a long time are still placed in the user layer. This not only improves performance, but also gives consideration to **security**. Nowadays, the most used operating systems (such as Apple, NeXT and Microsoft) are basically based on hybrid kernel or monolithic kernel architecture. Therefore, I think these two kernel architectures are superior.

C. Mohammed's View

This is a very useful discussion about designing operating systems. The subject of controversy has been the Linux kernel and kernel architecture in general. Tannenbaum started this controversy in 1992 on the Usenet newsgroup comp.os.minix, arguing that microkernels are superior to monolithic kernels, and thus Linux was out of date. Some famous hackers in the Linux development community (such as Theodore Y. Ts'o, etc.) also joined the debate.

This debate clarified the advantages and disadvantages of the two kernel architectures. One disadvantage of Linux is that it is a single kernel linked to x86. However, the architecture of the system is constantly changing, so the kernel version must be **flexible** and **extensible** and not associated with a single technical project. From this point of view, although Linux is superior in performance, it may not be the future choice.

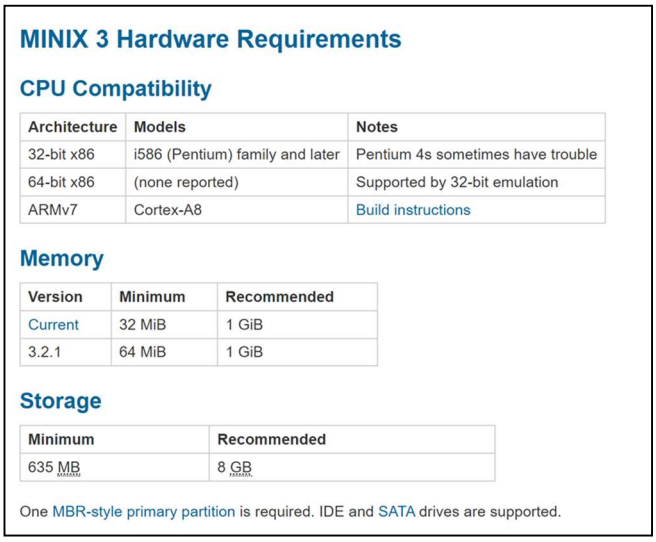
Although Minix does not fully support the updated hardware, Tannenbaum thinks that x86 will be surpassed by other architectures in the future, so he does not need to solve this problem. Undeniably, the core mechanism of microkernel architecture, **"message passing"**, greatly aggravates the **complexity** of feature implementation, which needs to be optimized.

The debate between the creators of Linux system and Minix system explained the methodology and scientific thinking mode of scientists in their time, and brought their experience to the peak of success and development.

III. MINIX HARDWARE JUSTIFICATION

The MINIX official organization gave a recommended configuration of 1GB main memory and 8GB secondary memory, but it is worth noting that this recommended configuration was written for the **3.2.1** version which was released in 2012.

Fig. 1. Recommend configuration from MINIX official.



MINIX 3 Hardware Requirements

CPU Compatibility

Architecture	Models	Notes
32-bit x86	i586 (Pentium) family and later	Pentium 4s sometimes have trouble
64-bit x86	(none reported)	Supported by 32-bit emulation
ARMv7	Cortex-A8	Build instructions

Memory

Version	Minimum	Recommended
Current	32 MiB	1 GiB
3.2.1	64 MiB	1 GiB

Storage

Minimum	Recommended
635 MB	8 GB

One MBR-style primary partition is required. IDE and SATA drives are supported.

Main Memory: It has been 9 years from the 3.2.1 version, the latest beta version is **3.4.0RC6**, and various tools supported by MINIX3 are also iterating. In addition, I have 16G of main memory. Therefore, I think if we use MINIX3 as a learning operating system for a long time. It would be appropriate to have allocate more main memory.

TABLE I. CONFIGURATION OF MY LAPTOP

Component	Configuration
Operating System	Window 11 insider build 22478
Processor	AMD Ryzen 5 5600H with Radeon Graphics 3.30 GHz (12CPUs)
Main memory	16.0GB Dual-Channel Unknown @ 1598MHz (22-22-22-52) (13.9 GB Available)
Secondary memory	476GB Western Digital WDC PC SN730 SDBPNTY-512G-1101 (Unknown (SSD))
MotherBoard	LENOVO LNVNB161216 (FP6)
Graphics	4096MB NVIDIA GeForce GTX 1650 2048MB ATI AMD Radeon Graphics
Audio	Realtek High Definition Audio

Secondary Memory: After I installed the **Xterm** graphical interface for my Minix and equipped it with **Git, Vim, Clang, gcc, ssh, vsftp** and other common tools, my 3.4.0RC6 MINIX3 has only taken up about **1.1 GB** of secondary memory space. Considering the fact that Minix supports fewer packages than other distribution, most of the

tools adapted to Minix are simplified, so I think 8GB is enough for Minix.

Chipset: PIIX3 and ICH9 both support I/O devices. ICH9 is still experimental. Although the PIIX3 chipset is older, it works best for VirtualBox.

Processor: By default, MINIX3 is built as single-processor system. Even if multiple processors are allocated, only one can be recognized.

Video Memory: I have 4096MB plus 2048 MB video memory on my host. In order to give the virtual machine better display performance, so allocate it with 128MB video memory (largest video memory for Virtual Box).

Network Mode: Bridge mode may encounter configuration network problems, so we choose NAT mode which allow virtual machine access the public network through the address where the host machine is located.

USB: MINIX3 original system does not support USB at all. We just disable it.

In summary, the following table is my Minix3 configuration information.

TABLE II. CONFIGURATION OF MINIX

Component	***Configuration***
Operating System	MINIX3.4.0RC6
Processor	1 Processor
Main memory	2 GB @ 1598MHz
Secondary memory	8 GB Western Digital SSD
MotherBoard	LENOVO LNVNB161216 (FP6)
Graphics	128 MB Video Memory
Audio	Realtek High-Definition Audio

IV. WORKING OF MINIX

After a series of basic configurations, we can enter the Minix operating system interface. Commonly, the Minix system is a command line interface.

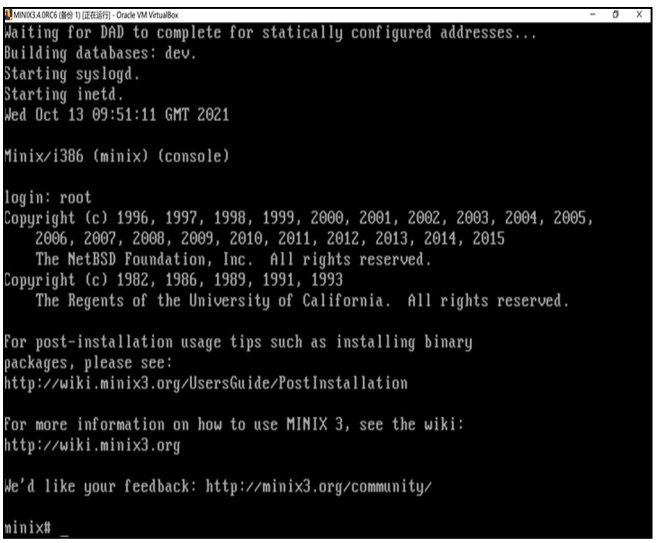


Fig. 2. Working Status of Minix OS.

A. *Using Browser on Minix*

Link is a web browser however it doesn't have an graphical interface.

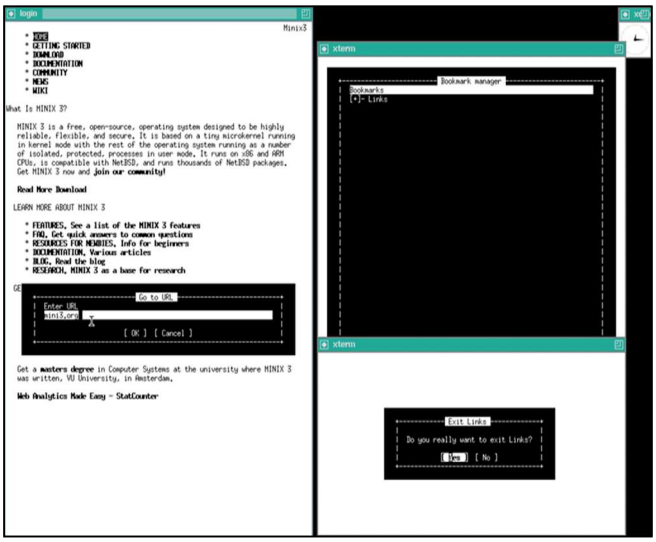


Fig. 3. Link browser on Minix3.

B. *Compile a calculator and clock in Minix3*

Xterm is a graphical interface system supported on Minix system. It provides corresponding clock and calculator components. Through simple compilation and running, we can run a graphical clock and calculator on Minix.

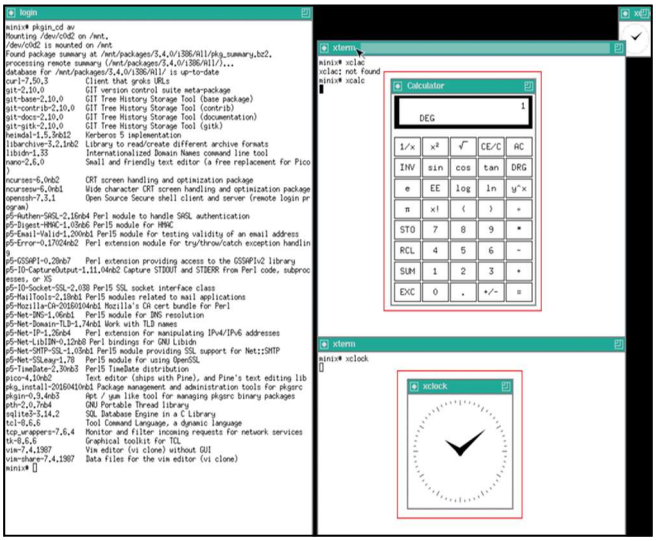


Fig. 4. Calculator and clock on Minix 3.

C. *Using SQLite database on Minix3*

SQLite is a lightweight database, It works fine for Minix Operating System. Here I create database, table, and insert dummy data in sequence.

```
0    main           /root/test.db
sqlite> .open test.db
sqlite> .tables
COMPANY
sqlite> .schema
CREATE TABLE COMPANY(
ID INT PRIMARY KEY NOT NULL,
NAME TEXT NOT NULL,
AGE INT NOT NULL
);
sqlite> .header on
sqlite> .mode column
sqlite> .timer on
sqlite> SELECT * FROM COMPANY
   ...> ;
ID          NAME        AGE
----------  ----------  ----------
1           David       20
2           Allen       25
3           Teddy       23
4           Mark        25
5           Paul        25
6           Kim         22
Run Time: real 0.000 user 0.000000 sys 0.000000
sqlite> _
```

Fig. 5. Select data from table.

D. *Using Git tool on Minix3*

As the most popular version management tool, Git is widely used by software developers, and Minix is also adapted to Git tool. Next, we try to use Git on Minix

```
--more--

minix#
minix# ls
.git            .gitattributes  readme.md
minix# cd ..
minix# rm -rf xuyichenmo/
minix# ls
.Xauthority         .klogin             .shrc               test.db
.cshrc              .login              .sqlite_history     test.txt
.gitconfig          .profile            Selenium-Document   testdir
minix# git clone https://github.com/xuyichenmo/xuyichenmo.git
Cloning into 'xuyichenmo'...
remote: Enumerating objects: 50, done.
remote: Counting objects: 100% (50/50), done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 50 (delta 15), reused 33 (delta 8), pack-reused 0
Unpacking objects: 100% (50/50), done.
minix# cd xuyichenmo/
minix# ls
.git            .gitattributes  readme.md
minix# cat .gitattributes
# Auto detect text files and perform LF normalization
* text=auto
minix#
```

Fig. 6. Using git tool.

Note:

Abubadi Ali Abudulrahman said he had the flu and would like to submit the video later!

Video Link:

SWE2009513 Su Yanyu:
https://xmueducn-my.sharepoint.com/:v:/g/personal/swe2009510_xmu_edu_my/EZ0cnUiFpf1LpxM04e1bXBkB5v-juHjL5d_9b5r9pYnjgg?e=TKYvZM

SWE2009510 Liu Aofan:
https://xmueducn-my.sharepoint.com/:v:/g/personal/swe2009510_xmu_edu_my/EREjPVuZDVVBuuuQxV8UAAQBG33zPRDGDpPA7yc6OVqgMw?e=JF7axv

Back Up Link:

SWE2009513 Su Yanyu:
https://drive.google.com/file/d/1h0opRiqWcwvZnTnm0tQVupv39rfCOZLu/view?usp=sharing

SWE2009510 Liu Aofan:
https://drive.google.com/file/d/150fkvDCMgub5Ypp3YpMAYksTJLos9Hoc/view?usp=sharing

APPENDIX 1

MARKING RUBRICS

Component Title	**Practical Implementation & Documentation**		**Percentage (%)**	**70%**	
Criteria	**Score and Descriptors**			**Weight (%)**	**Marks**
	Excellent (5)	**Good-Average (3-4)**	**Need Improvement (1-2)**		
Debate summary	The students summarize the key points of the debate very clearly.	The students miss some of the critical points of the debate.	The students are not able to summarize and understand.	**20**	
Debate discussion	The students show excellence in demonstrating their opinion(s) on the debate. Also, the students can relate the discussion with the current advancement of the technologies/systems.	The discussion is valid, but few critical issues are not discussed.	The students need to have a better understanding of the debate in order to have a valid discussion.	**30**	
Minix 3 virtualization	The students select and justify proper hardware for the operating system. Also, the students can run the operating system and use it for a creative task (e.g., using commands or compiling a code).	The hardware selected is not optimum, but the operating system is running.	The students attempted but failed to run the operating system.	**20**	
			TOTAL		

Component Title	**Video Presentation (Individual)**		**Percentage (%)**	**30%**	
Criteria	**Score and Descriptors**			**Weight (%)**	**Marks**
	Excellent (5)	**Good-Average (3-4)**	**Need Improvement (1-2)**		
Video and presentation quality	The presenter shows remarkable skill in demonstrating. The video quality is clear, and the student is able to cover the contents within the given time limit.	The presenter's sound is clear and understandable. The video quality is average and less than five minutes.	The quality of the presentation and video needs improvement.	**10**	
Presentaiton contents	The student manages to summarize and discusses the main critical points of the debate. Also, the student can demonstrate their contribution to the assignment.	The student can demonstrate their understanding on the debate.	The student shows a lack in demonstrating their understanding and opinion on the debate.	**20**	
			TOTAL		

Note to students: Please print out and attach this appendix together with the submission of coursework