Lens: Experiencing Multi-level Page Tables at Close Quarters

Arun KP Rohit Singh Debadatta Mishra

Indian Institute of Technology, Kanpur

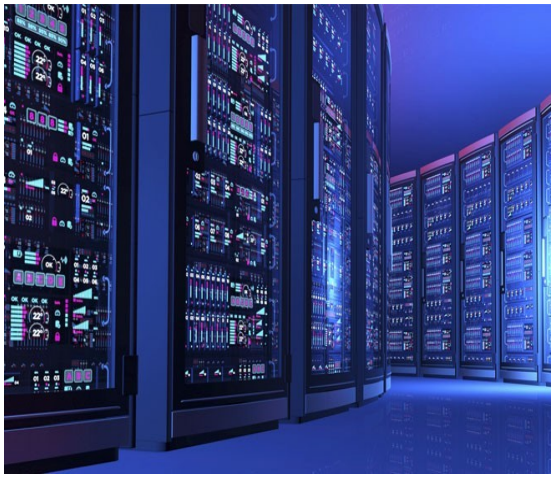
Importance of Operating Systems

Importance of Operating Systems

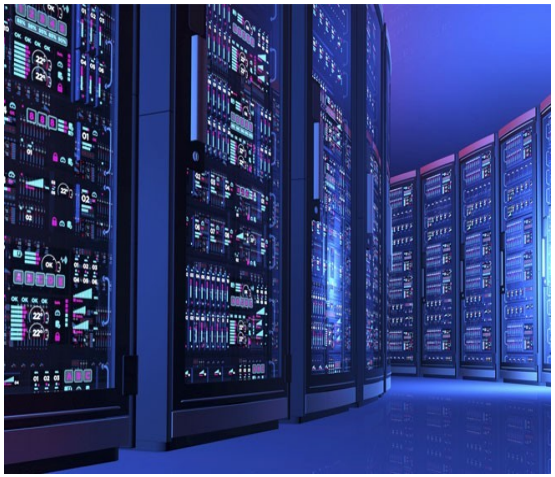
- OS is a building block in today's digital world.

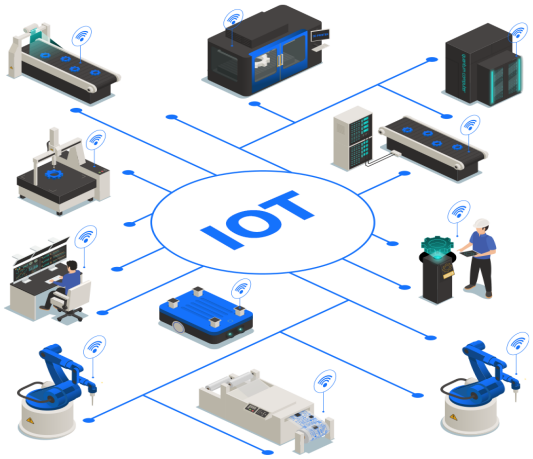
Importance of Operating Systems

Importance of Operating Systems

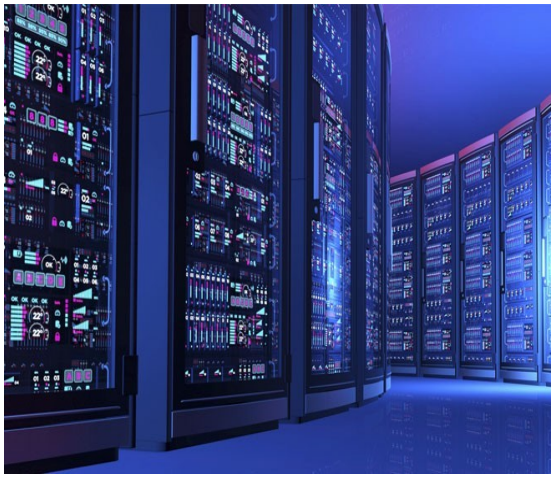


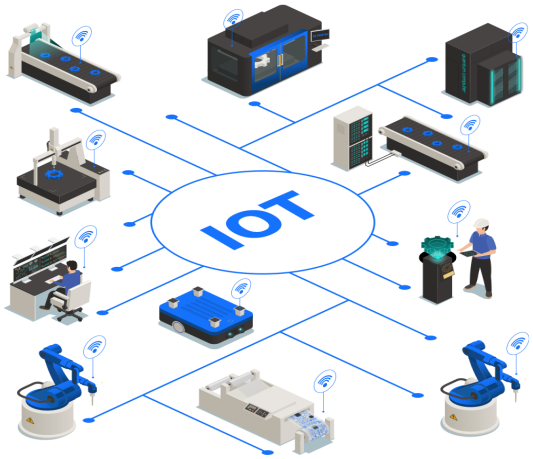
Importance of Operating Systems



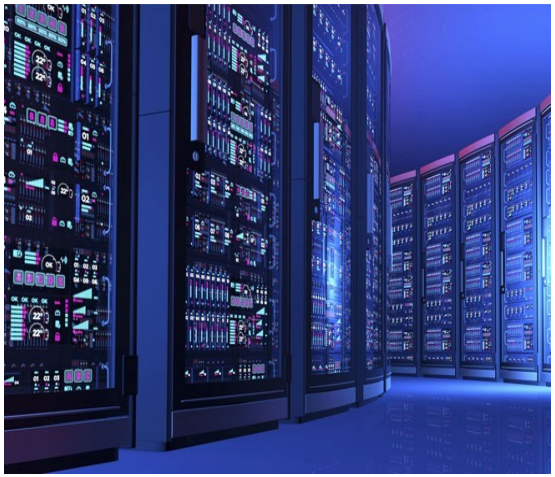


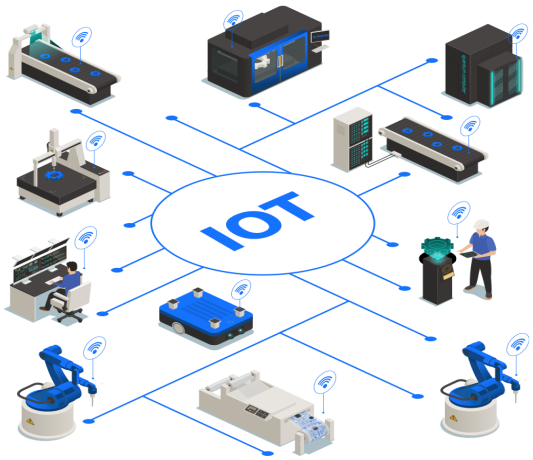
Importance of Operating Systems



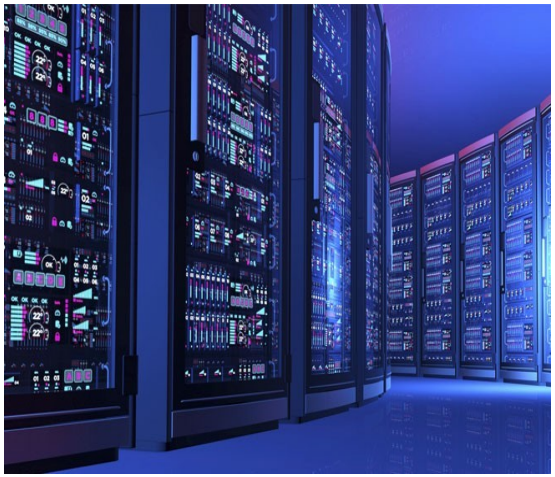


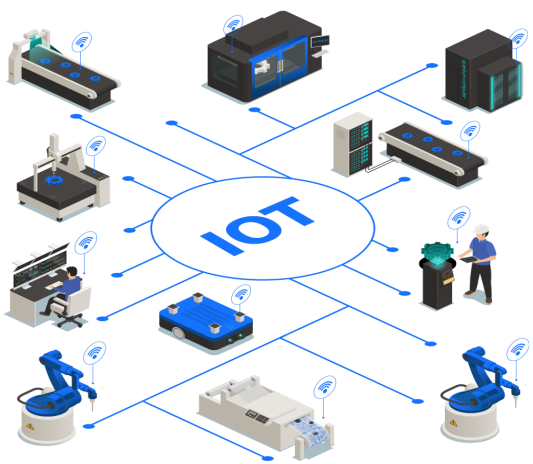
Importance of Operating Systems





Importance of Operating Systems





Importance of Operating Systems

- OS is a building block in today's digital world.
- OS helps us to perform our day-to-day digital activities better.




Importance of Operating Systems

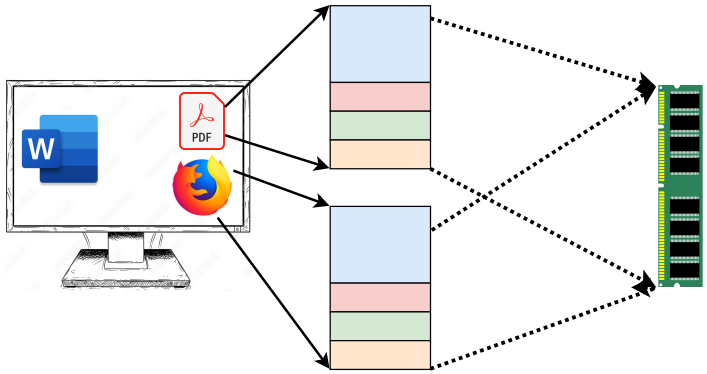
- OS is a building block in today's digital world.
- OS helps us to perform our day-to-day digital activities better.
- OS encompasses multiple subsystems such as File system, Virtual Memory, and the interplay between them.




The Virtual Memory Sub-system

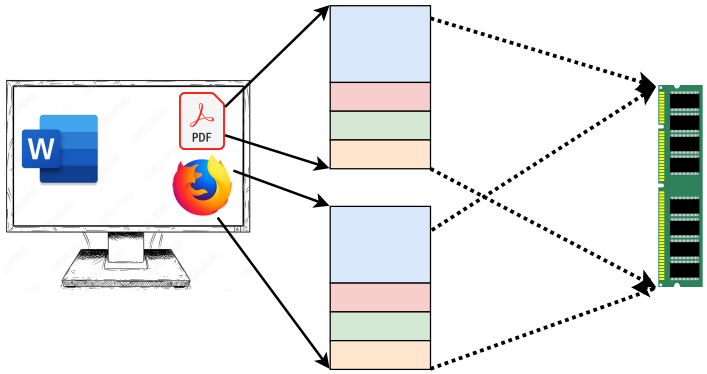
Understanding the VM Sub-system

- Virtual memory allows inter-process isolation.



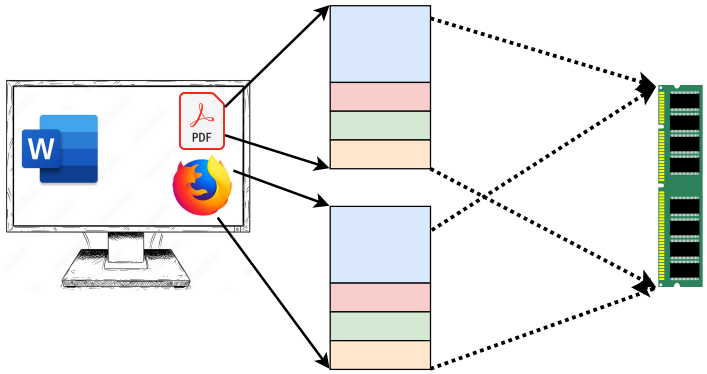
Understanding the VM Sub-system

- Virtual memory allows inter-process isolation.
- Each process has an independent view of memory.



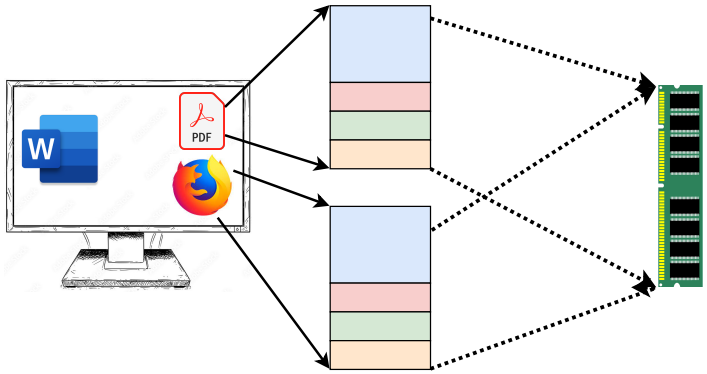
Understanding the VM Sub-system

- Virtual memory allows inter-process isolation.
- Each process has an independent view of memory.
- Virtual memory makes a programmer's life easy.



Understanding the VM Sub-system

- Virtual memory allows inter-process isolation.
- Each process has an independent view of memory.
- Virtual memory makes a programmer's life easy.
- Virtual memory facilitates memory access permission.



Understanding the VM Sub-system

Text Books

✓ Use simple examples to explain concepts

Educational OS

Debugging Interfaces

Understanding the VM Sub-system

Text Books

✓ Use simple examples to explain concepts

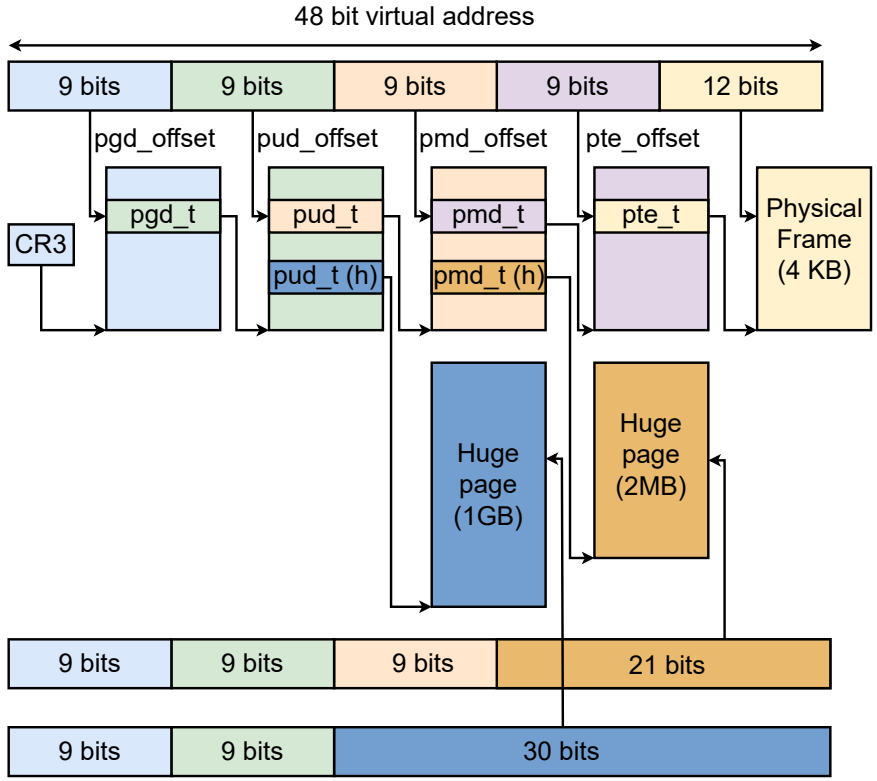
✗ Do not capture real-world complexities

Educational OS

Debugging Interfaces

Page table in Intel x86-64 processor

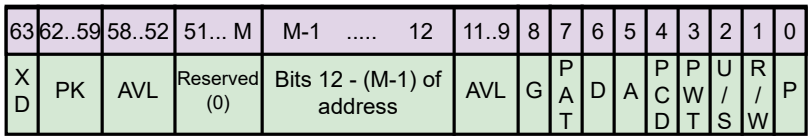
- 4 level paging translates 48 bit virtual address to physical address.
- Allows translation into 4 KB, 2MB, and 1 GB pages.
- Each entry provides details about the next level page.



Ref: [Guide(2011)]

Page table entry in Intel x86-64 system

- PTE captures access permissions, access mode and other details.
- Intel software manual details page table structures and different configurations.
- Challenging for a beginner to correlate page table details with a program's virtual memory state.



P: Present
R/W: Read/Write
U/S: User/Supervisor
PWT: Write-Through
PCD: Cache Disable
A: Accessed
D: Dirty
G: Global
AVL: Available
PAT: Page Attribute Table
M: Maximum Physical Address Bit
PK: Protection Key
XD: Execute Disable

Ref: [Guide(2011)]

Understanding the VM Sub-system

Text Books

Educational OS

- ✓ Profiling the code with debug statements

Debugging Interfaces

Understanding the VM Sub-system

Text Books

Educational OS

✓ Profiling the code with debug statements

✗ Requires understanding the code base

Debugging Interfaces

Understanding the VM Sub-system

Text Books

Educational OS

Debugging Interfaces

✓ Reading the interface for fetching predefined virtual memory details

Understanding the VM Sub-system

Text Books

Educational OS

Debugging Interfaces

✓ Reading the interface for fetching predefined virtual memory details

✗ Gives limited information for education

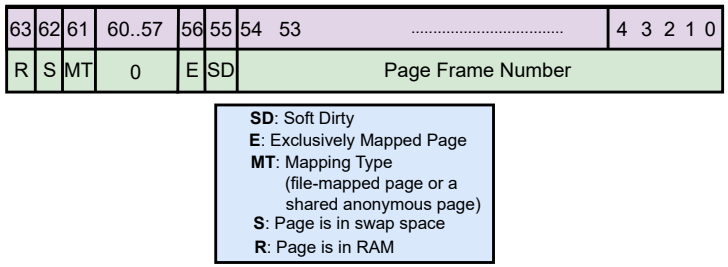
VM interfaces in Linux

- */proc/pid/maps* shows currently mapped memory regions.
- */proc/pid/maps* is a pseudo file maintained under proc file system for each process.
- It is not straightforward to correlate a program variable address with information from */proc/pid/maps*.

```
55a4c3600000-55a4c3601000 r-xp 00000000 08:02 25958718 program_code
55a4c3800000-55a4c3801000 r--p 00000000 08:02 25958718  program_code
55a4c3801000-55a4c3802000 rw-p 00001000 08:02 25958718  program_code
55a4c55fa000-55a4c561b000 rw-p 00000000 00:00 0         [heap]
7f4407200000-7f44073e7000 r-xp 00000000 08:02 36177205  libc-2.27.so
7f44073e7000-7f44075e7000 ---p 001e7000 08:02 36177205  libc-2.27.so
7f44075e7000-7f44075eb000 r--p 001e7000 08:02 36177205  libc-2.27.so
7f44075eb000-7f44075ed000 rw-p 001eb000 08:02 36177205  libc-2.27.so
7f44075ed000-7f44075f1000 rw-p 00000000 00:00 0
7f4407600000-7f4407629000 r-xp 00000000 08:02 36177201  ld-2.27.so
7f4407829000-7f440782a000 r--p 00029000 08:02 36177201  ld-2.27.so
7f440782a000-7f440782b000 rw-p 0002a000 08:02 36177201  ld-2.27.so
7f440782b000-7f440782c000 rw-p 00000000 00:00 0
7f4407929000-7f440792b000 rw-p 00000000 00:00 0
7ffd9028a000-7ffd902ab000 rw-p 00000000 00:00 0         [stack]
7ffd9034b000-7ffd9034f000 r--p 00000000 00:00 0         [vvar]
7ffd9034f000-7ffd90351000 r-xp 00000000 00:00 0         [vdso]
ffffffffff600000-ffffffffff601000 --xp 00000000 00:00 0 [vsyscall]
```

VM interfaces in Linux

- */proc/pid/pagemap* shows virtual to physical page mapping.
- It requires calculating the file offset corresponding to a program variable in the pagemap file to get its physical page.
- Only gives information in the last level of a page table (PTE).



Tool for Learning Virtual Memory

Features of an educational tool for VM

- Augments concept learning with practical exposure.

Features of an educational tool for VM

- Augments concept learning with practical exposure.
- Provides details from a real system without missing actual complexities.

Features of an educational tool for VM

- Augments concept learning with practical exposure.
- Provides details from a real system without missing actual complexities.
- Presents an interface that is easy-to-use.

Features of an educational tool for VM

- Augments concept learning with practical exposure.
- Provides details from a real system without missing actual complexities.
- Presents an interface that is easy-to-use.
- Expects only basic C programming knowledge from learners.

Features of an educational tool for VM

- Augments concept learning with practical exposure.
- Provides details from a real system without missing actual complexities.
- Presents an interface that is easy-to-use.
- Expects only basic C programming knowledge from learners.
- Allows learners to correlate program execution with memory state.

Features of an educational tool for VM

- Augments concept learning with practical exposure.
- Provides details from a real system without missing actual complexities.
- Presents an interface that is easy-to-use.
- Expects only basic C programming knowledge from learners.
- Allows learners to correlate program execution with memory state.
- Visualizes details for easy understanding by learners.

Features of an educational tool for VM

- Augments concept learning with practical exposure.
- Provides details from a real system without missing actual complexities.
- Presents an interface that is easy-to-use.
- Expects only basic C programming knowledge from learners.
- Allows learners to correlate program execution with memory state.
- Visualizes details for easy understanding by learners.
- Shows changes in real-time by updating visualization in sync with program changes.

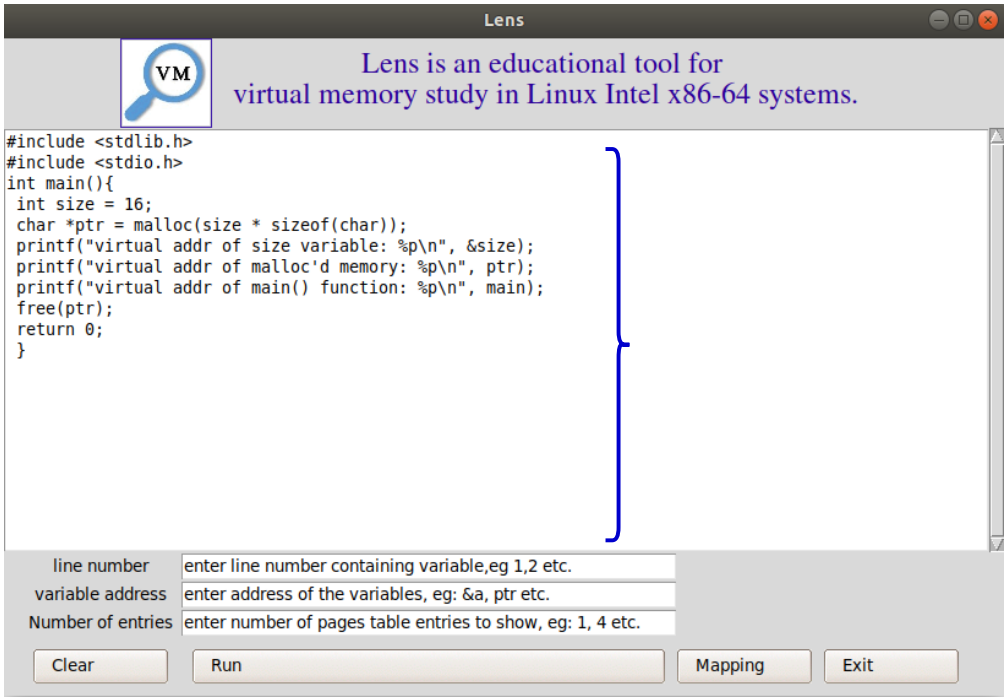
Lens: An Education Tool for VM

What does Lens offer?

- Provides a practical exposure to virtual memory in Linux.

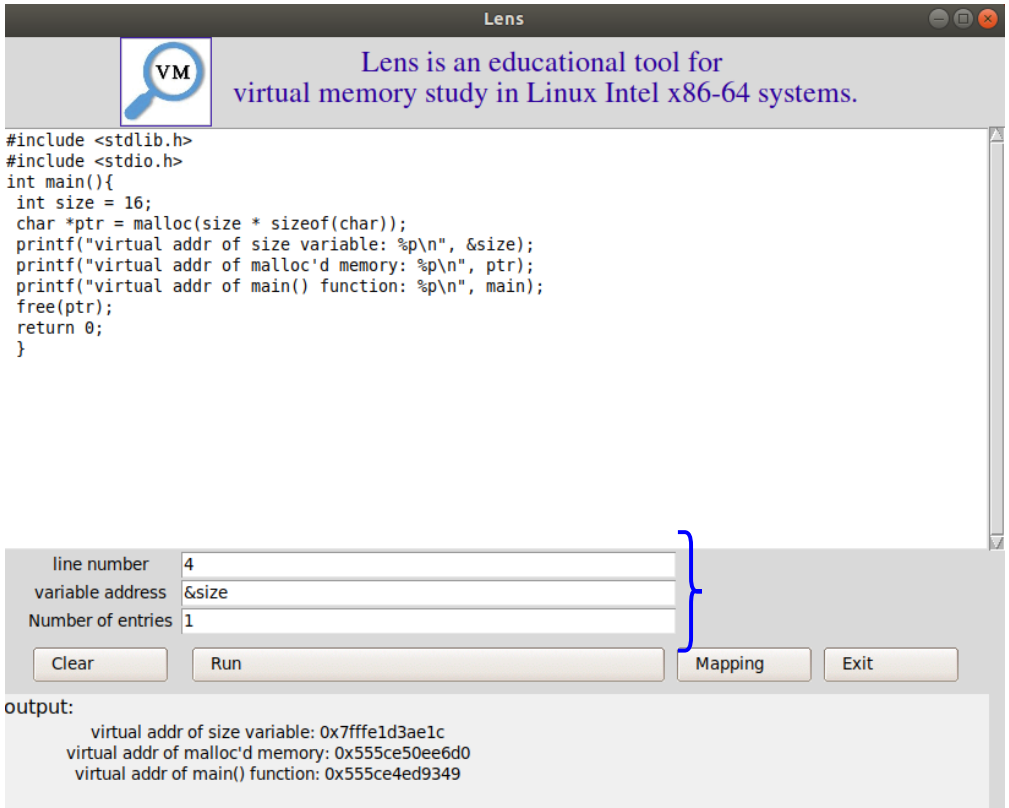
What does Lens offer?

- Provides a practical exposure to virtual memory in Linux.
- Provides an interface to
  - Write C programs.



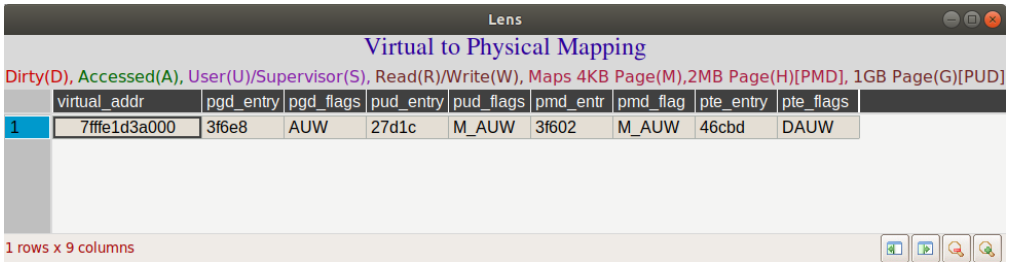
What does Lens offer?

- Provides a practical exposure to virtual memory in Linux.
- Provides an interface to
  - Write C programs.
  - Correlate program execution with memory state.



What does Lens offer?

- Provides a practical exposure to virtual memory in Linux.
- Provides an interface to
  - Write C programs.
  - Correlate program execution with memory state.
  - Visualize virtual to physical address translation.



Start Video

What does Lens offer?

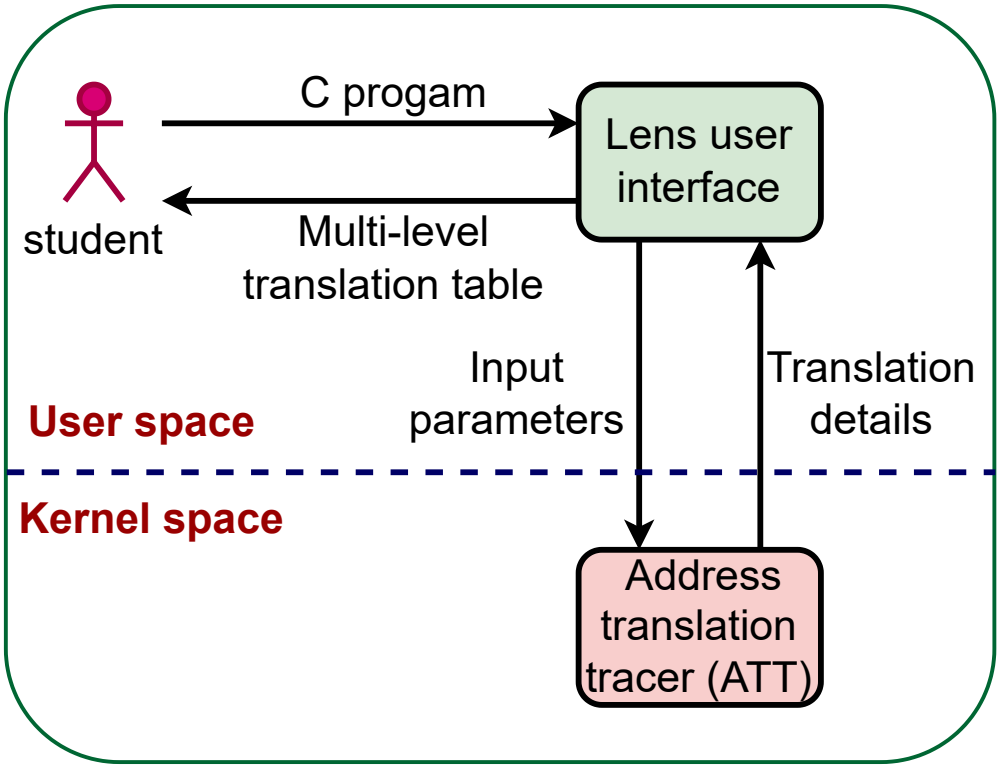
- Provides a practical exposure to virtual memory in Linux.
- Provides an interface to
  - Write C programs.
  - Correlate program execution with memory state.
  - Visualize virtual to physical address translation.
- Covers underlying hardware-OS complexities.

What does Lens offer?

- Provides a practical exposure to virtual memory in Linux.
- Provides an interface to
  - Write C programs.
  - Correlate program execution with memory state.
  - Visualize virtual to physical address translation.
- Covers underlying hardware-OS complexities.
- Shows address translation table with latest updates in program.

High level view of Lens

- Student interacts with graphical interface.
- ATT handles underlying hardware-OS complexities.



Example use cases with Lens

- Lazy page allocation in Linux.
- Linux allocates physical pages on first access.

Inspecting lazy page allocation

```
1 #define TWOMB 2097152
2 int main(int argc, char* argv[]){
3     char * ptr = (char*)mmap(NULL, TWOMB, PROT_READ|PROT_WRITE, MAP_PRIVATE|
      MAP_ANONYMOUS,0,0);
4     ptr[0] = 'A';
5     printf("ptr:%p\n",ptr);
6     ptr[4096] = 'B';
7     mprotect(&ptr[4096],4096,PROT_READ);
8     munmap(ptr, TWOMB);
9     return 0;}
```

Inspecting at line number 4

Inspecting lazy page allocation

```
1 #define TWOMB 2097152
2 int main(int argc, char* argv[]){
3     char * ptr = (char*)mmap(NULL, TWOMB, PROT_READ|PROT_WRITE, MAP_PRIVATE|
       MAP_ANONYMOUS,0,0);
4     ptr[0] = 'A';
5     printf("ptr:%p\n",ptr);
6     ptr[4096] = 'B';
7     mprotect(&ptr[4096],4096,PROT_READ);
8     munmap(ptr, TWOMB);
9     return 0;}
```

Inspecting at line number 4

pud_entry	pud_flags	pmd_entry	pmd_flags	pte_entry	pte_flags
36d4b	M_AUW	ffa2	M_AUW	1c8e4	DAUW
36d4b	M_AUW	ffa2	M_AUW	0	____

Example use cases with Lens

- Lazy page allocation in Linux.
- Linux allocates physical pages on first access.
- Changing memory access permission.
- Linux provides *mprotect* sys call to change access protection of pages.

Changing memory access permission

```
1 #define TWOMB 2097152
2 int main(int argc, char* argv[]){
3     char * ptr = (char*)mmap(NULL, TWOMB, PROT_READ|PROT_WRITE, MAP_PRIVATE|
       MAP_ANONYMOUS,0,0);
4     ptr[0] = 'A';
5     printf("ptr:%p\n",ptr);
6     ptr[4096] = 'B';
7     mprotect(&ptr[4096],4096,PROT_READ);
8     munmap(ptr, TWOMB);
9     return 0;}
```

Inspecting at line number 7

Changing memory access permission

```
1 #define TWOMB 2097152
2 int main(int argc, char* argv[]){
3     char * ptr = (char*)mmap(NULL, TWOMB, PROT_READ|PROT_WRITE, MAP_PRIVATE|
       MAP_ANONYMOUS,0,0);
4     ptr[0] = 'A';
5     printf("ptr:%p\n",ptr);
6     ptr[4096] = 'B';
7     mprotect(&ptr[4096],4096,PROT_READ);
8     munmap(ptr, TWOMB);
9     return 0;}
```

Inspecting at line number 7

pud_entry	pud_flags	pmd_entry	pmd_flags	pte_entry	pte_flags
36d4b	M_AUW	ffa2	M_AUW	1c8e5	DAUR

Example use cases with Lens

- Lazy page allocation in Linux.
- Linux allocates physical pages on first access.
- Changing memory access permission.
- Linux provides *mprotect* sys call to change access protection of pages.
- Lens can also provide insights to other virtual memory concepts like hugepages.

Future Directions

- Use Lens in an OS course for collecting feedback.
- Show address translation table as a radix tree and make the interface interactive for students to zoom-in.
- Visualize more virtual memory changes such addition, deletion and merging of VM areas, stack/heap growth and shrink with changes in program.
- Visualize more OS activities related to virtual memory, like page faults.

Conclusion

- Practical understanding of virtual to physical address translation is essential in learning virtual memory.
- Students require hands-on experience with virtual memory concepts to enhance their learning.
- Lens provides an easy interface for students to write C code and correlate program state with changes in virtual memory.
- Lens helps students in gaining practical exposure to virtual memory concepts.
- Lens is available at https://github.com/arunkp1986/Lens.git

Thank You!

Questions?

Arun KP
kparun@cse.iitk.ac.in

References

Part Guide.
Intel® 64 and ia-32 architectures software developer's manual.
Volume 3B: System programming Guide, Part, 2:5, 2011.