# LINUX KERNEL DEVELOPMENT (LKD)

Session 1



# Loadable Kernel Modules (LKM): Laboratory

Paulo Baltarejo Sousa pbs@isep.ipp.pt 2017



ISCP Instituto Superior de Engenharia do Porto



## 1 Introduction

In the Linux Kernel Development (LKD) module, we will use a virtual machine called CISTER. This virtual machine was created using the VirtualBox software running the operating system Linux with distribution Ubuntu 16.04 Linux (Xenial Xerus) and with the kernel version 4.10.0-28-generic. This system is configured with only one user called cister and the password is also cister.

# 2 Get Virtual Machine

Download the virtual machine from http://www.cister.isep.ipp.pt/sum mer2017/w1/CISTER.ova

1. From VirtualBox main menu select File > Import Appliance.



2. Select the CISTER.ova file and click on Next.



3. Check the Appliance Settings and click on Import.

😣 🗉 Import Virtual Appliance				
	Appliance settings			
	These are the virtual machines contained in the appliance and the suggested settings of the imported VirtualBox machines. You can change many of the properties shown by double-clicking on the items and disable others using the check boxes below.			
	Description	Configuration		
	Virtual System 1	_		
	😪 Name	CISTER_1		
	🗏 Guest OS Type	🐕 Ubuntu (64-bit)		
	CPU	1		
	RAM	4096 MB		
	OVD			
	USB Controller			
	<u>R</u> einitialize the MAC add	Iress of all network cards		
	Restore Defaults	< Back Import Cancel		

4. Start the CISTER virtual Machine.

New Settings State Card		🙆 Details 🗯 Snapshots
Prevent Of	📇 General	E Preview
	Name: CISTER Operating System: Ubuntu (64-bit)	
	3 System	
	Base Memory - 4094 MB Book Offer: Flogs (CRDW), Isard Dak Accidenstor: V7 #JAMD V, Nested Paging	CISTER
	B Display	
	Video Memory: 12 MB Remote Desktop Server: Disabled Video Capture: Disabled	
	Storage	
	Controller-IDE IDE Secondary Master: [Cx]DVD] Empty Controller SATA SATA Part c: CLSTER.xmdl, Neomal, 16,00 C8)	
	De Audio	
	Host Driver: PulseAudio Controller: ICH AC97	
	P Network	
	Adapter 1: Intel PRO/1000 MT Desktop (NAT)	
	Ø use	
	Device Filters: 0 (0 active)	
	Shared folders	
	Shared Folders: 1	
	Secretaria	
	None	

5. Type the password: cister



# 3 Tools and packages

In order to install the required tools and packages, open a terminal



and type the following linux commands:
> sudo apt-get update
> sudo apt-get install build-essential

### 4 Learning examples

Download the learning examples from http://www.cister.isep.ipp.pt/s ummer2017/w1/S01-LKM.tar.gz. In this compressed file you can find several source code modules and a Makefile. Extract it and enjoy!. Feel free to inquiry the Lecturer.

#### 4.1 hello.c

This is the simplest Linux kernel module.

- 2. Insert it into the Linux kernel:
   > sudo insmod ./hello.ko
- 3. Check printk messages:
   > dmesg
- 4. Check list modules:
   > lsmod

- 5. Remove it from the Linux kernel: > sudo rmmod hello
- 6. Check printk messages: > dmesg

#### 4.2 hello2.c

This is a basic module that create a new entry in the **proc** directory. Further, it uses **file\_operations** structure.

- Change Makefile: obj-m:=hello2.o
- 3. Insert it into the Linux kernel:
   > sudo insmod ./hello2.ko
- 4. Change /proc/hello2 permissions:
   > sudo chmod 666 /proc/hello2
- 6. Perform write operation:
   > echo "aa" > /proc/hello2
   > dmesg
- 7. Remove it from the Linux kernel: > sudo rmmod hello2
- 8. Check printk messages:
   > dmesg

#### 4.3 hello3.c

This module allows to read and write messages from/to kernel.

 Change Makefile: obj-m:=hello3.o

- 3. Insert it into the Linux kernel:
   > sudo insmod ./hello3.ko
- 4. Change /proc/hello3 permissions:
   > sudo chmod 666 /proc/hello3
- 5. Perform read operation: > cat /proc/hello3
- 7. Perform read operation: > cat /proc/hello3
- Remove it from the Linux kernel:
   > sudo rmmod hello3
- 9. Check printk messages:
   > dmesg

#### 4.4 ring\_buffer.c

This module implements a First In First Out (FIFO) ring buffer. A ring buffer is a container that uses a single, fixed-size buffer as if it was connected end-to-end.



- Change Makefile: obj-m:=ring\_buffer.o
- 3. Insert it into the Linux kernel:
   > sudo insmod ./ring\_buffer.ko
- 4. Change /proc/ring\_buffer permissions:
   > sudo chmod 666 /proc/ring\_buffer
- 5. Test it by performing a set of write and read operations:

```
> echo "1" > /proc/ring_buffer
> echo "2" > /proc/ring_buffer
> echo "3" > /proc/ring_buffer
> echo "4" > /proc/ring_buffer
> ...
> cat /proc/ring_buffer
> cat /proc/ring_buffer
> cat /proc/ring_buffer
```

- > cat /proc/ring\_buffer
- 6. Remove it from the Linux kernel: > sudo rmmod ring\_buffer
- 7. Check printk messages:
   > dmesg

#### 4.5 fifo\_queue.c

This module implements a FIFO queue through a linked list.



- Change Makefile: obj-m:=fifo\_queue.o
- 3. Insert it into the Linux kernel:
   > sudo insmod ./fifo\_queue.ko

- 4. Change /proc/fifo\_queue permissions:
   > sudo chmod 666 /proc/fifo\_queue
- 5. Test it by performing a set of write and read operations:
  - > echo "1" > /proc/fifo\_queue > echo "2" > /proc/fifo\_queue > echo "3" > /proc/fifo\_queue > echo "4" > /proc/fifo\_queue > ... > cat /proc/fifo\_queue > cat /proc/fifo\_queue > cat /proc/fifo\_queue > cat /proc/fifo\_queue
- 6. Remove it from the Linux kernel:
- > sudo rmmod fifo\_queue
- 7. Check printk messages:
   > dmesg

#### 4.6 sorted\_queue.c

This module implements a sorted queue through a Red-Black Tree.



- Change Makefile: obj-m:=sorted\_queue.o

- 3. Insert it into the Linux kernel:
   > sudo insmod ./sorted\_queue.ko
- 4. Change /proc/sorted\_queue permissions:
   > sudo chmod 666 /proc/sorted\_queue
- 5. Test it by performing a set of write and read operations:
  - > echo "1" > /proc/sorted\_queue > echo "4" > /proc/sorted\_queue > echo "3" > /proc/sorted\_queue > echo "2" > /proc/sorted\_queue > ... > cat /proc/sorted\_queue > cat /proc/sorted\_queue > cat /proc/sorted\_queue > cat /proc/sorted\_queue
- 6. Remove it from the Linux kernel: > sudo rmmod sorted\_queue
- 7. Check printk messages: > dmesg

### 5 Assignment

- Write the pseudo-code of all functions of the ring\_buffer.c, fifo\_queue.c and sorted\_queue.c;
- 2. Update the ring\_buffer.c, fifo\_queue.c and sorted\_queue.c in order to support concurrency.