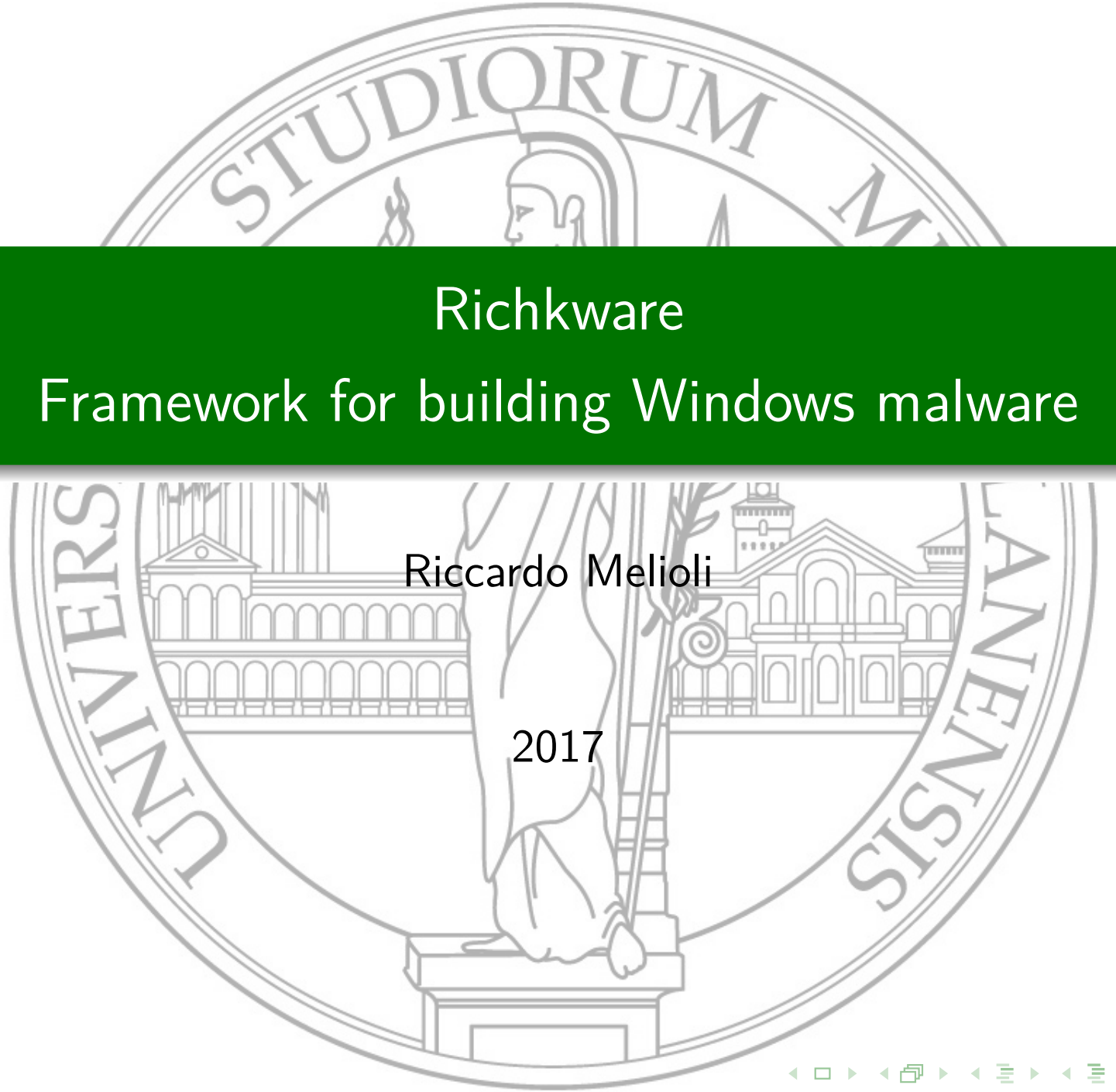


Richkware

Framework for building Windows malware

Riccardo Melioli

2017

Goal

Goal

Create a library that allows the development of any kind of malware in a simple way.

Target OS

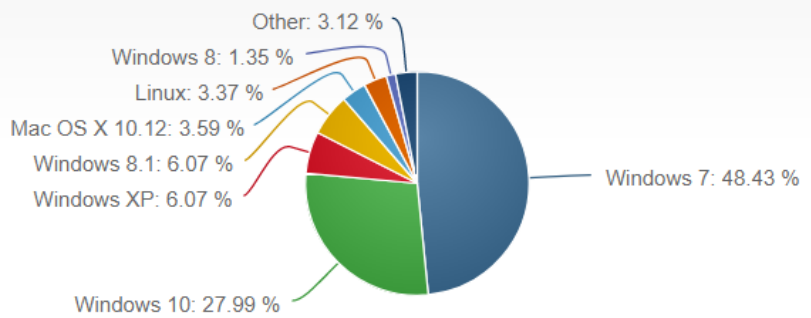
Target OS

The project is developed for the Microsoft Windows operating system as the target of attacks, because many **vulnerabilities** are continually discovered, which could be exploited to gain more functionality. Windows is the **most common** operating system, so there is more computers being infected

OS share

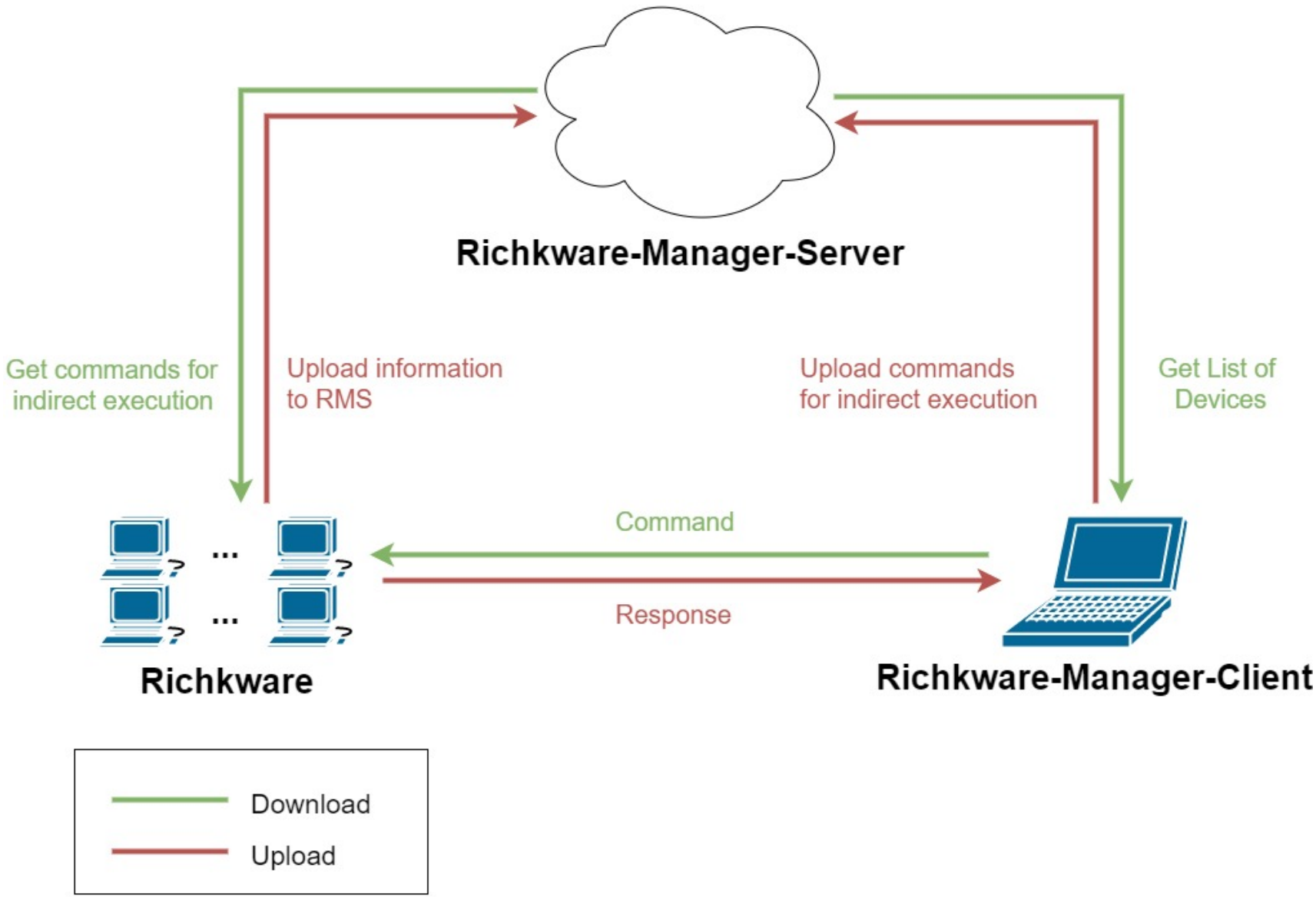
Desktop Operating System Market Share

August, 2017



OPERATING SYSTEM	TOTAL MARKET SHARE
Windows 7	48.43%
Windows 10	27.99%
Windows XP	6.07%
Windows 8.1	6.07%
Mac OS X 10.12	3.59%
Linux	3.37%
Windows 8	1.35%
Mac OS X 10.11	1.09%
Mac OS X 10.10	0.72%
Windows Vista	0.46%
Windows NT	0.31%
Mac OS X 10.9	0.26%
Mac OS X 10.6	0.09%
Mac OS X 10.8	0.09%
Mac OS X 10.7	0.07%
Mac OS X 10.13	0.02%
Mac OS X 10.5	0.01%
Windows 2000	0.00%
Mac OS X 10.4	0.00%

Project Structure



Richkware

What is it?

It's a library of network and OS functions, that you can use to create malware. The **composition** of these functions permits the application to assume behaviors referable to the following types of malware:

- Virus
- Worm
- Bot
- Spyware
- Keylogger
- Scareware

RMS - Richkware-Manager-Server

What is it?

Service for management of hosts where is present a malware developed using Richkware framework. It **stores** all malware informations in a SQL database:

- **Name**: name of device, where malware is present
- **IP**: malware IP address
- **Server Port**: TCP port opened by the malware, it allows the remote connection and the remote commands execution
- **Last Connection**: date and time of last malware connection
- **Encryption Key**: Server-side generated Encryption Key, is used by the malware to encrypt data.

RMS - Richkware-Manager-Server

List of Devices

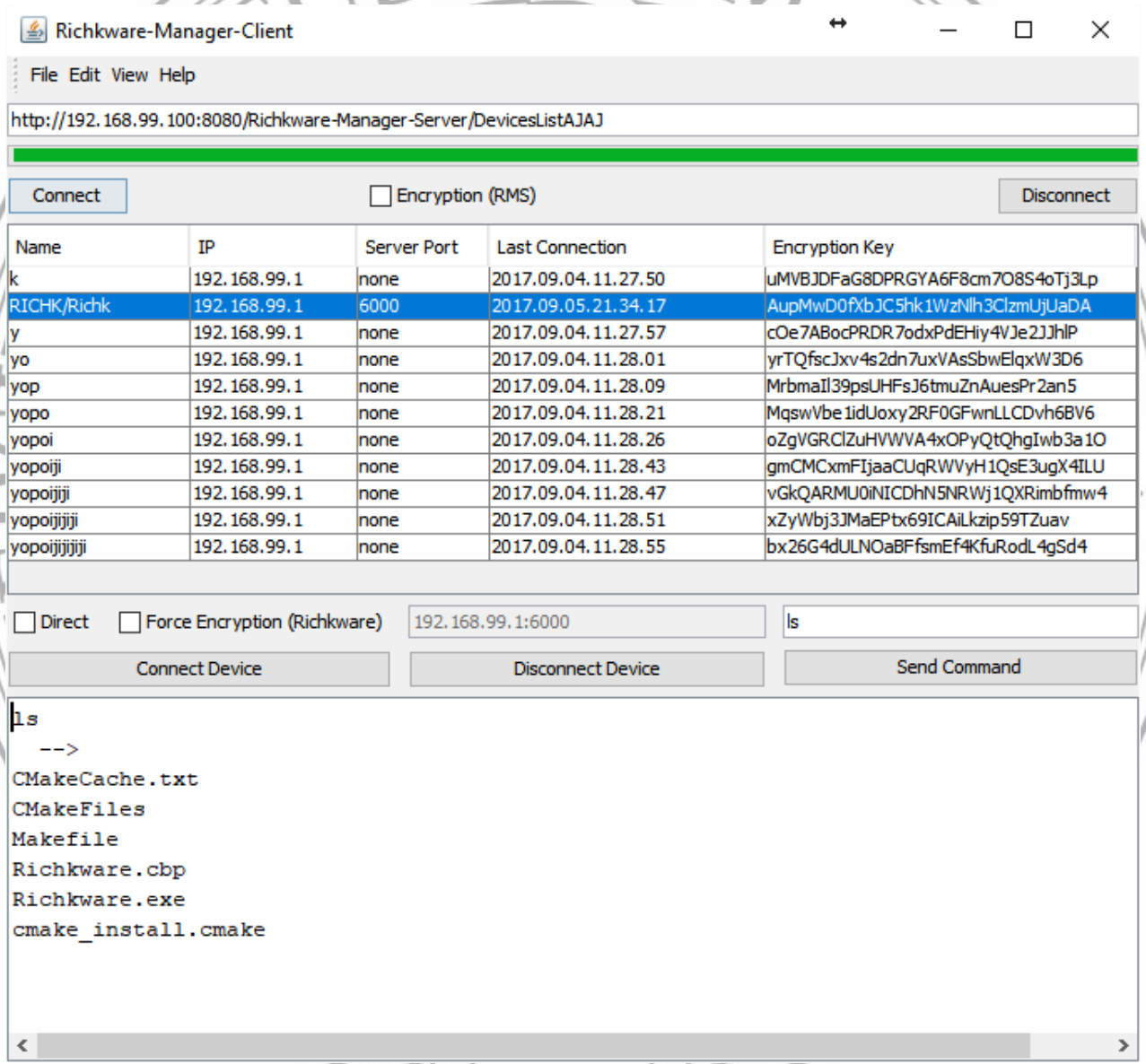
Name	IP	Server Port	Last Connection	Encryption Key		
k	192.168.99.1	none	2017.09.04.11.27.50	uMVBJDFaG8DPRGYA6F8cm7O8S4oTj3Lp	Edit	Remove
RICHK/Richk	192.168.99.1	6000	2017.09.05.13.27.44	AupMwD0fXbJC5hk1WzNlh3ClzmUjUaDA	Edit	Remove
y	192.168.99.1	none	2017.09.04.11.27.57	cOe7ABocPRDR7odxPdEHiy4VJe2JJhlP	Edit	Remove
yo	192.168.99.1	none	2017.09.04.11.28.01	yrTQfscJxv4s2dn7uxVAsSbwElqxW3D6	Edit	Remove
yop	192.168.99.1	none	2017.09.04.11.28.09	MrbmaII39psUHFsJ6tmuZnAuesPr2an5	Edit	Remove
yopo	192.168.99.1	none	2017.09.04.11.28.21	MqswVbe1idUoxy2RF0GFwnLLCDvh6BV6	Edit	Remove
yopoi	192.168.99.1	none	2017.09.04.11.28.26	oZgVGRClZuHVWVA4xOPyQtQhgIwb3a1O	Edit	Remove
yopoiji	192.168.99.1	none	2017.09.04.11.28.43	gmCMCxmFIjaaCUqRWVyH1QsE3ugX4ILU	Edit	Remove
yopoijiji	192.168.99.1	none	2017.09.04.11.28.47	vGkQARMU0iNICDhN5NRWj1QXRimbfmw4	Edit	Remove

RMC - Richkware-Manager-Client

What is it?

Richkware-Manager-Server Client, gets the list of all hosts from the server and allows to **send commands** to run on the infected pc, by safe communication.

RMC - Richkware-Manager-Client



Communication Protocol between RMC and Richkware

Communication Protocol

The protocol allows to the RMC user to **interact** with the pc where Richkware is installed.
In Richkware, the requests received from the RMC are sent to a **dispatcher** (implemented in protocol.h), which dispatches the request by a defined code, it executes the request, and returns the response to RMC.
The dispatcher is implemented as follows:

Dispatcher

```
...

switch (commandID) {
        case 0:
                response = "***quit***";
                break;
        case 1:
                response = CodeExecution(command);
                break;
        case 2:
                //...
                break;
        default:
                response = "error: Command ID not found\n";
}
```

Syntax of the request

Syntax of the request

The syntax of the request is as follows:

[[1]]ls

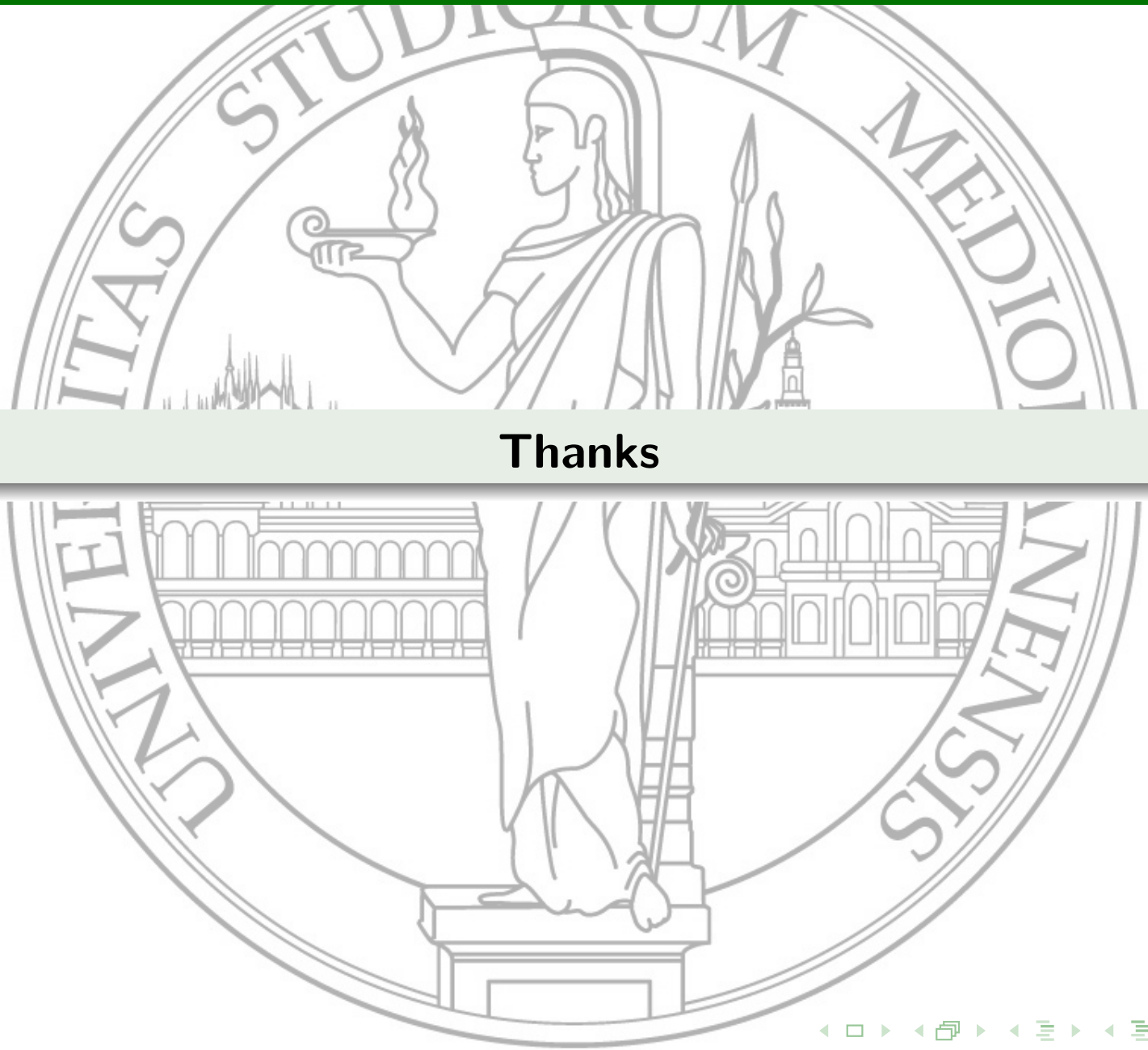
The previous command, having parameter "1", means that you are requesting the execution of the following string as a shell command, then "ls" will run from the Windows shell and the response will be sent to the client.

Future development

Possible future developments are:

- creating a **smart main** that applies artificial intelligence concepts, such as hiding itself by antivirus and it takes decisions based on the external situation.
- extend the library with **new features**, such as creating a ransomware.

End



Thanks