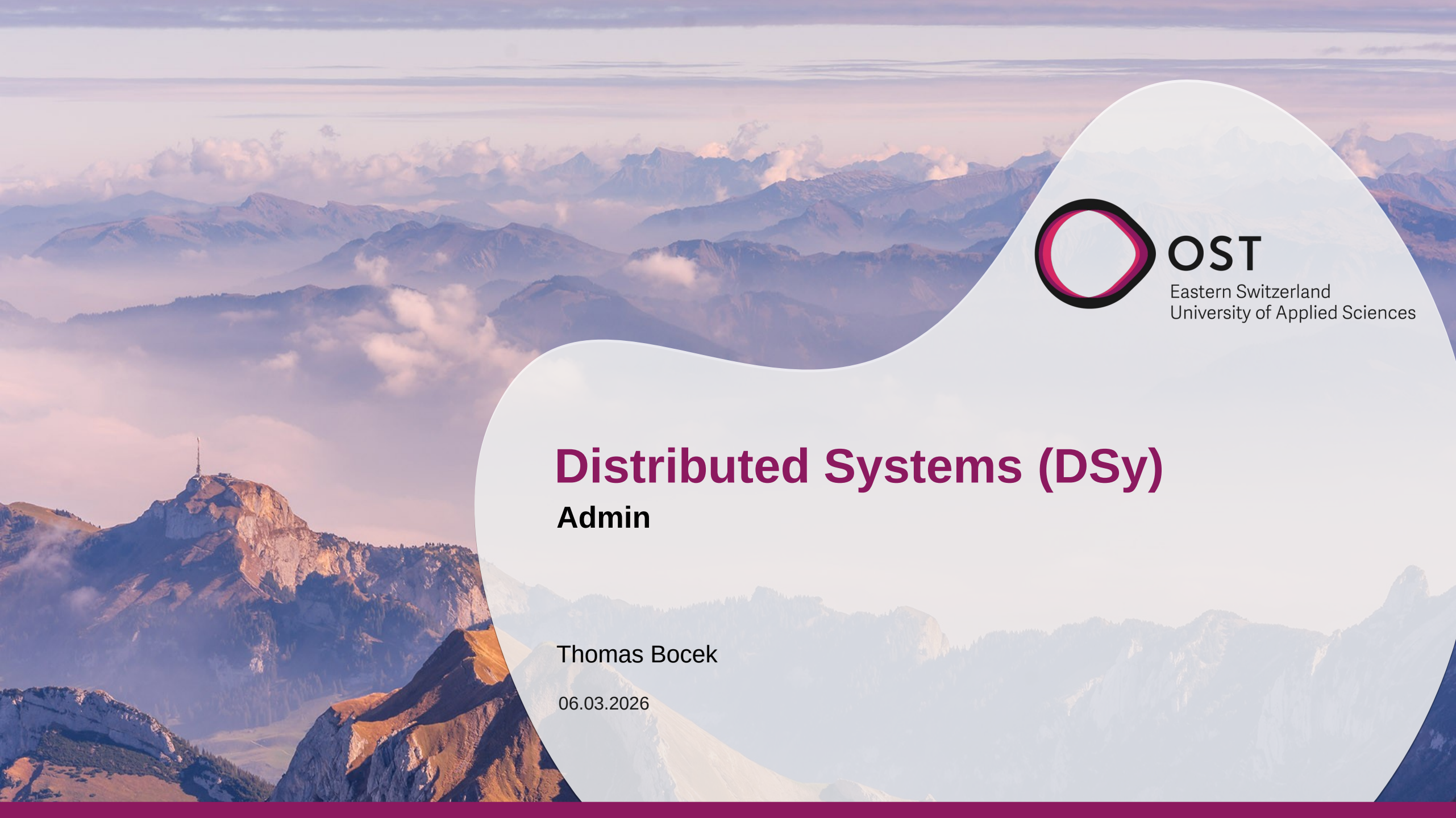

OST
Eastern Switzerland
University of Applied Sciences

Distributed Systems (DSy)

Admin

Thomas Bocek

06.03.2026

Nr	Date	Lecture*	Lecturer
01	17.02.2026	Admin, Introduction / Motivation, part 1	Bocek
02	23.02.2026	Introduction / Motivation, part 2	Bocek
03	02.03.2026	Containers and VMs, Docker	Bocek
04	09.03.2026	Monorepos / Polyrepos	Bocek
05	16.03.2026	Load Balancing	Bocek
06	23.03.2026	Web Architectures	Bocek
07	30.03.2026	Categorization	Bocek
-	-	-	-
08	13.04.2026	Authentication	Bocek
09	20.04.2026	Protocols	Bocek
10	27.04.2026	Application Protocols	Bocek
11	04.05.2026	Exam preparation	Bocek
12	11.05.2026	Deployment, Performance, Blockchain, Bitcoin	Bocek
13	18.05.2026	Challenge Task Presentations	Bocek
-	-	-	you
14	01.06.2026	Challenge Task Award Winner Announcement, Q&A	Bocek

* topics may change

OST

DSy Exercises

- Hand in earlier → YES!
- Secrets don't belong in the repo; they must be provisioned separately beforehand (e.g. .env file, docker secret create, kubectl create secret, Vault
- Default password for localhost “12345678” in .env.example, but never external services
- Update on monorepo vs polyrepo [link]

- Type 1 (bare-metal): runs directly on hardware, no host OS needed (e.g. Xen, ESXi).
  - minimal hypervisor for CPU scheduling and memory partitioning
  - HW → Hypervisor → VMs = Type 1
- Type 2 (hosted): runs on top of a host OS as a regular process (e.g. VirtualBox, VMware Workstation).
  - HW → OS → Hypervisor = Type 2
- KVM: HW → Linux+KVM → VMs = ?
  - KVM is a hypervisor, so its Type1
  - KVM needs OS, so its Type2

Virtualization vs. Emulation

- Virtualization
  - `int i = 42 → mov ax, 0x42`
  - Hypervisor: "are you allowed to run this?”
    - Yes: execute mov ax, 0x42 on real CPU
- Emulation
  - `int i = 42 → mov ax, 0x42`
  - Instruction never touches the real CPU
  - Emulator reads it like a script:
    - `if opcode == "mov ax, *" {register_ax = 0x42}`
    - register_ax: variable in memory, emulated register

- Performance
  - Virtualization = one real CPU instruction
  - Emulation = the emulator must parse the opcode, match it, update memory, turning one instruction into many software operations.
  - Orders of magnitude slower
- Trade-off
  - Virtualization needs same architecture on both sides
  - Emulation can cross architectures
- Apple emulation: translates x86 instructions to ARM ahead of time - faster