Microservices Deployment with Containers – Part 2

This lesson continues the discussion about microservice deployment with containers.

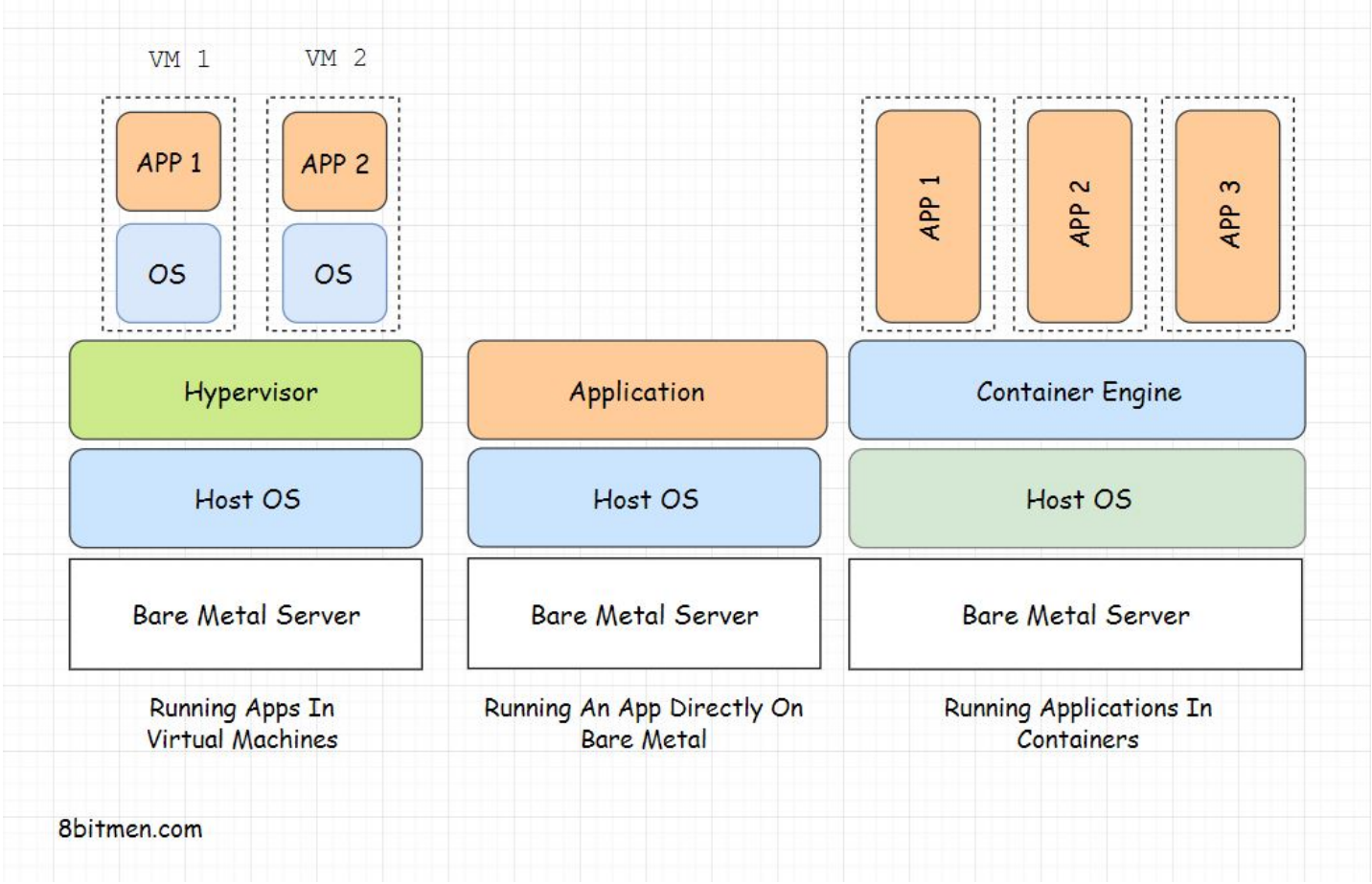
We'll cover the following

Every microservice has its own specific configuration, dependencies, deployment requirements like scalability requirement, resource monitoring requirement, and so on.

When there is a large number of microservices, containers cut down a great deal of hassle in application deployment and can assist the business decrease the time it takes to ship the software.

The primary ways of deploying and running applications on the cloud are:

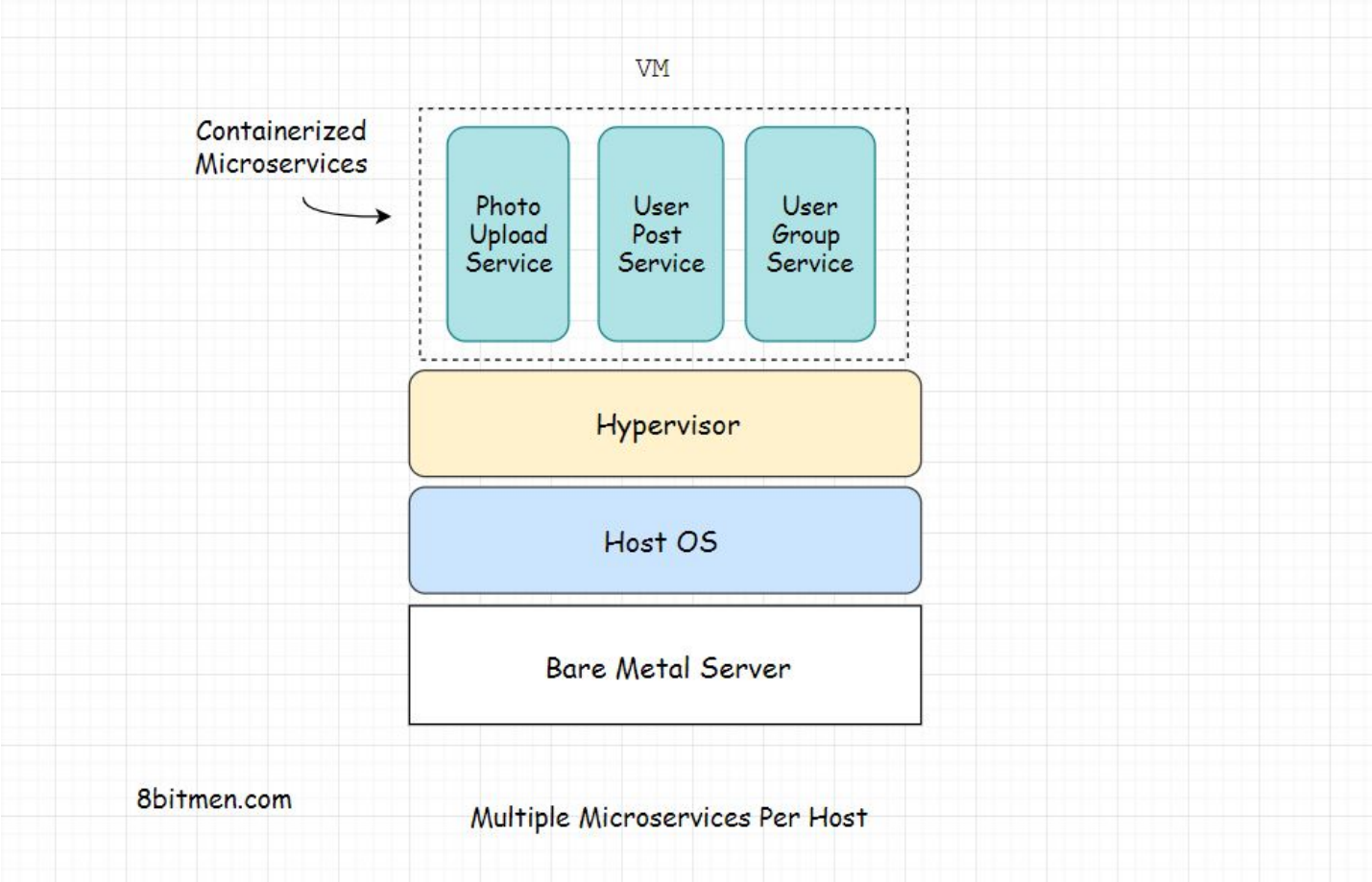
- Running the application directly in a virtual machine
- Running the application directly on bare-metal
- Running the application in a container running on either bare-metal or a virtual machine



Now let's talk about some of the common deployment patterns of microservices with containers.

Multiple microservices per host

In this microservice deployment pattern, multiple microservices are deployed in a single host. The host can be a single virtual machine or a single physical *bare-metal* server.



One straight-forward upside of doing this is the efficient use of the host resources.

In our social network application use case, we have multiple microservices built using different technology stacks. Imagine configuring a single machine to run multiple microservices without using the containers. Installing multiple technologies on a single machine trying to make it compatible for all the microservices would be a mess.

To avoid this mess without containers, we don't have many options apart from running one service in one machine.

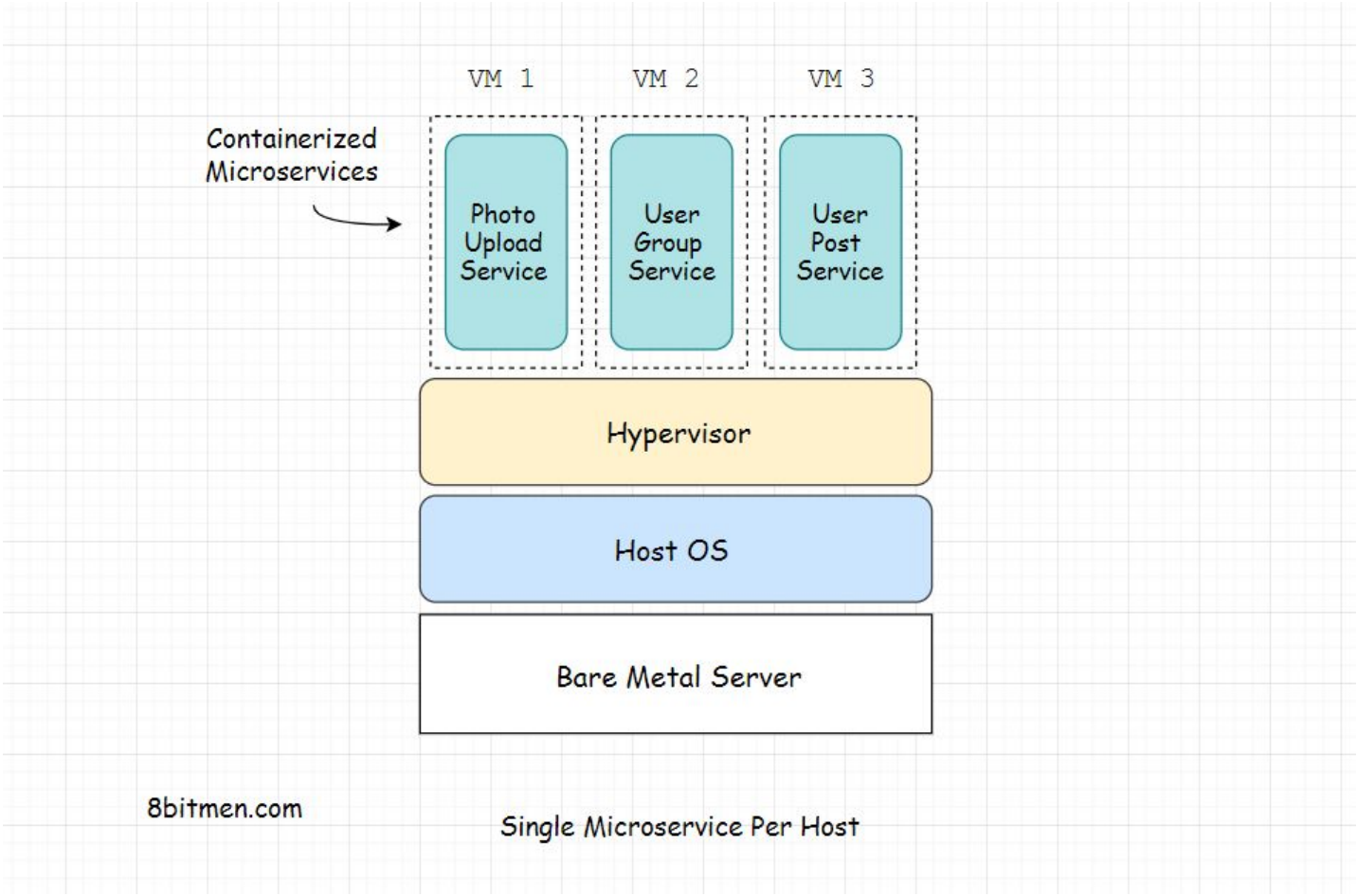
However, with containers, we can easily implement the multiple services in one host approach without trouble because they keep the application isolated from the environment. Every container has everything it needs to run the microservice.

One downside of using this deployment approach is the noisy neighbor problem, which we discussed earlier.

> ***A good read***: Why bare metal is challenging *VMs* in microservices deployment?

Single microservice per host

In this deployment approach, only one microservice is deployed in a single host. There is no noisy neighbor problem in this approach. A single microservice has access to 100% of the host resources, making the approach more secure and reliable in comparison to the former approach.



As the number of containers running gets higher, the infrastructure complexity increases. We need a system that we can leverage to easily manage so many containers. This is where container orchestration tools like *Kubernetes, Docker Swarm, and Apache Mesos* come in.

You will learn more about these tools in the next lesson.