Artem Chernitsa, B20-AI-01, a.chernitsa@innopolis.university

Would setting time in a container update host time as well? If yes, then could it be prevented? If not, how is it limited?

По-умолчанию недоступно, но можно запустить контейнер привилегированно и можно будет менять время изнутри (--privileged флаг). Ещё *cap_sys_time* – работает с системным RTC (real-time clock) и может изменять время. Поэтому есть *seccomp profile* – дополнительная фильтрация (ограничение системных вызовов для процесса в linux), чтобы сказать, что такие-то вызовы запрещены, но с этой штукой теряем в производительности. По умолчанию, выставлен seccomp profile, который запрещает вызов около 44 системных функций.

I. https://docs.docker.com/engine/security/seccomp/
II. https://man7.org/linux/man-pages/man2/seccomp.2.html
III. https://kubernetes.io/docs/tutorials/security/seccomp/

What are problems about running a few processes inside one container (if any)?

1. There's a good chance you'd have to scale APIs and front-ends differently than databases.
2. Separate containers let you version and update versions in isolation.
3. While you may use a container for the database locally, you may want to use a managed service for the database in production. You don't want to ship your database engine with your app then.
4. Running multiple processes will require a process manager (the container only starts one process), which adds complexity to container startup/shutdown.

Хочется также отметить, что при в случае использования большого количества ресурсов система может убить какой-то процесс (контейнер) и тогда несколько приложений, запущенных там будут завершены, что может быть проблемой. Кроме того, процессы так нарушают изоляцию, мешают потреблению

ресурсов друг другу, и более того, если будет установлено ограничение на контейнер, то процессы будут делить оставшиеся ресурсы, что может быть губительно.

Ещё это ограничения, например, на доступ к файловой системе, на сетевое соединение и т.д. Поэтому технически запустить несколько можно, но это небезопасно и плохо управляется, как если бы каждый процесс жил в своём контейнере.

I. https://docs.docker.com/get-started/07_multi_container/
II. https://docs.docker.com/config/containers/multi-service_container/
III. https://docs.docker.com/config/containers/resource_constraints/

What is better and why: running similar or unsimilar containers on the one physical node?

Думаю, что есть смысл запускать похожие, как минимум потому, что они могут делить **образ**, что экономит ресурсы на диске.

“So, how do you backup your container, you don’t. Your data doesn’t live in the container, it lives in a named volume that is shared between 1-N containers that you define. You backup the data volume, and forget about the container. Optimally your containers are completely stateless and immutable.” Но! Они могут создавать гонку за ресурсами, если используют доступ к одним ресурсам.

Кроме того, проще будет делать paging, если будет похожая read-only память. Т.е. запущенные контейнеры могут делить общие данные (на уровне хостовой ОС, которые она сама предоставит), что-то **закэширует** и, таким образом, повысит производительность. Есть так называемые **volumes** – механизм сохранения данных, генерируемых и используемых контейнерами Docker, которые могут быть поделены между контейнерами.

Однако всё это может нарушить изоляцию и безопасность, поэтому однозначного ответа на этот вопрос дать нельзя, но точно можно сказать, что определить повышенную производительность, а затем масштабировать и настраивать эффективнее при наличии схожих докер-контейнеров.

I. https://www.docker.com/blog/containers-are-not-vms/
II. https://docs.docker.com/storage/volumes/