Hello everyone. I am sadid Rafsun tulon and welcome to my presentation. My topic is “**An Effective way to optimize performance and battery life of smart devices using multi-kernel approach”.**

So transistors are reaching their size limit. Industry still trying to come up with clever methods to to shrink it even further but those methods are really expensive and complex, So complex that only few companies have the capacity to manufacture them. Battery technology has also slowed down. Scientists are trying to breakthrough but those will take years to come to the mass market. So Now we need a approach to improve performance and battery life that is not constrained by hardware. Most of our modern chips are using multiple cores. Multi-kernel system is a new promising method that can leverage these multi-core chips.

So what is a multikernel system?

Multi-kernel system is a technique to run multiple operating system (OS) kernels, typically Linux kernel and a lightweight kernel simultaneously. They can communicate with each other using shared memory or message passing. Here the full weight kernel mainly handles compatibility and the light weight sub kernel handles the high performance tasks. So, I have basically explained a new concept using another new concept, light weight kernel.

So what is light weight kernel and Unikernels?

The term light weight kernel is kind of vague as there are many types of kernel that are considered lightweight. In most of the cases light weight kernel means those kernels that are specialized for high end HPC, have small footprint, provide faster performance and are more efficient.

Here is some difference between the fwk and lwk. Basically fwk are general purpose os that are built with compatibility in mind while lwks are designed to be recourse efficient in very specialized systems.

Beside lwk, I have also mentioned another type of kernel called unikernel. Unikernel concept is relatively new. Unikernel uses library operating system. custom operating systems compiled from source code that only contain the functionality required for specific operation. This makes some Unikernels even lighter than lightweight kernels in some cases. We can see from the picture that unikernel houses all the necessary modules in all-in-one package.

Even though lwk and Unikernels are not same, in this project our main focus is on performance and power efficiency, not the underlying architecture. In this regard, they are similar when comparing to Full weight kernel like linux. So we will be generalizing them and call them sub kernel.

So we have tried to test multiple sub kernels on our 4th gen core i5 system. As we can see , most of them did not compile as they are intended for server cpus like intel xeon phi and fujutsu a64fx

We only succeed to compile and boot Unikraft and ran flask server on it. The boot time of the os on average was 3 seconds.

As we can see unikraft is pretty lightweight and fast. Even running in kvm, it is still faster than native linux. It also requires less memory than the competitors.

As for power consumption, we can see that flask sever running on linux is using less power than flask running on unikraft. Why is that? Actually this comparison is unfair as unikraft flask server is a whole operating system but we are not including linux power consumption as well.

If we include linux as well the chart would become like this.

So multi-kernel system seems promising, but we also have to face reality.

First of all, availability is really low right now. As mentioned previously most only available for server cpus. Some available for generic consumer grade cpus. Arm cpu support is almost nonexistent. As our projects main target is battery powered smart devices, arm cpus should have been the main focus. Unfortunately, we could not run anything meaningful on arm powered raspberry pi 4.0.

Now for disadvantages Multikernel systems are fairly new experimental concept and currently cpus are not made to explicitly support such thing. For this reason, even though not impossible, implementing multikernel systems are hard. Even I was not able to fully implement multikernel system. While I was able to boot unikraft it was not true multikernel system as unikraft is running on kvm.

As mentioned previously, as hardware requirement is very specific for most multikernel systems, community support is also very limited.

In conclusion, multikernel system is a promising concept but needs more support. Currently, some kernels such as Unikraft, Include os, mirage os are receiving decent support but there are still room for improvement. As the support for these kernels gets better, they should also be more available to wider range of systems.

So that is it for my presentation. Now I have two questions for the audience. As I have mentioned previously, kernels in multikernel system can communicate with each other using ither shared memory or message passing. I intentionally did not explain this in detail so I could ask this question. So my first question is “For multi-kernel communication, between shared memory and message passing, which one is more scalable?”

The answer is message passing, because In the case of message passing, client core issue a lightweight remote procedure call and generally some cores are responsible to pass the message. In case of shared memory cores have to take turns to update message in shared memory. This waiting time grows exponentially as the number of cores increase. So message passing is more scalable.

My second question is “Is there any similar technique that is currently being used on consumer products?” It does not have to be multikernel, but any similar technique would do.

There is a smart watch called ticwatch pro that uses something similar to save battery life. That watch has a mode called essential mode. In that mode all the smart features of the watch turn off including main lcd panel. A secondary display only shows time and some limited fitness data.