SRECON25EMEA

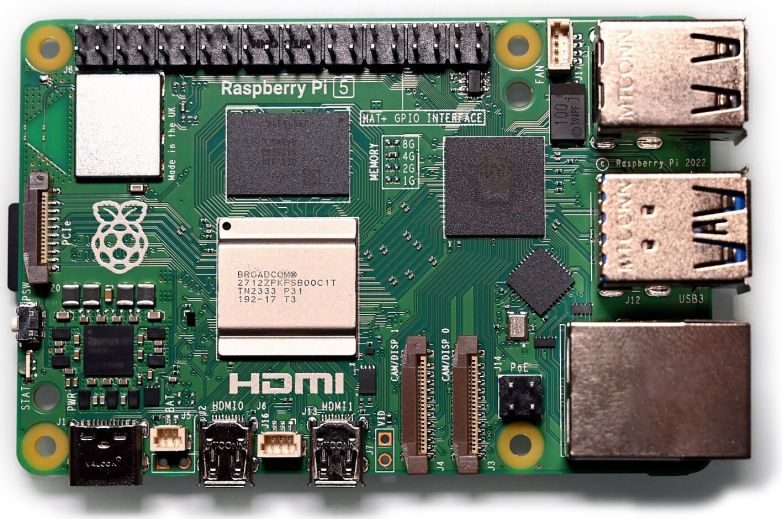
#25emea-day2-track1

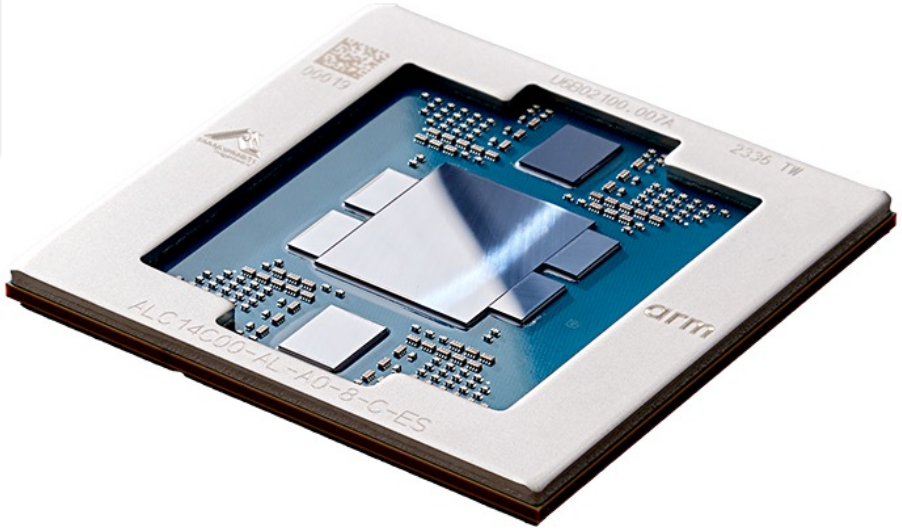
ARM Migration Made Practical

Nati Cohen

Sr. Specialist Solutions Architect, Compute
AWS

It’s not just an x86 world anymore





Why should you consider arm64?

Better price / performance

Improve sustainability

Support different hardware

Different CPU architecture?

x86_64

```
mov     DWORD PTR [rbp-4], 10
mov     DWORD PTR [rbp-8], 20
mov     edx, DWORD PTR [rbp-4]
mov     eax, DWORD PTR [rbp-8]
add     eax, edx
```

arm64

```
mov     w0, 10
str     w0, [sp, 12]
mov     w0, 20
str     w0, [sp, 8]
ldr     w1, [sp, 12]
ldr     w0, [sp, 8]
add     w0, w1, w0
```

different ISA, different instructions

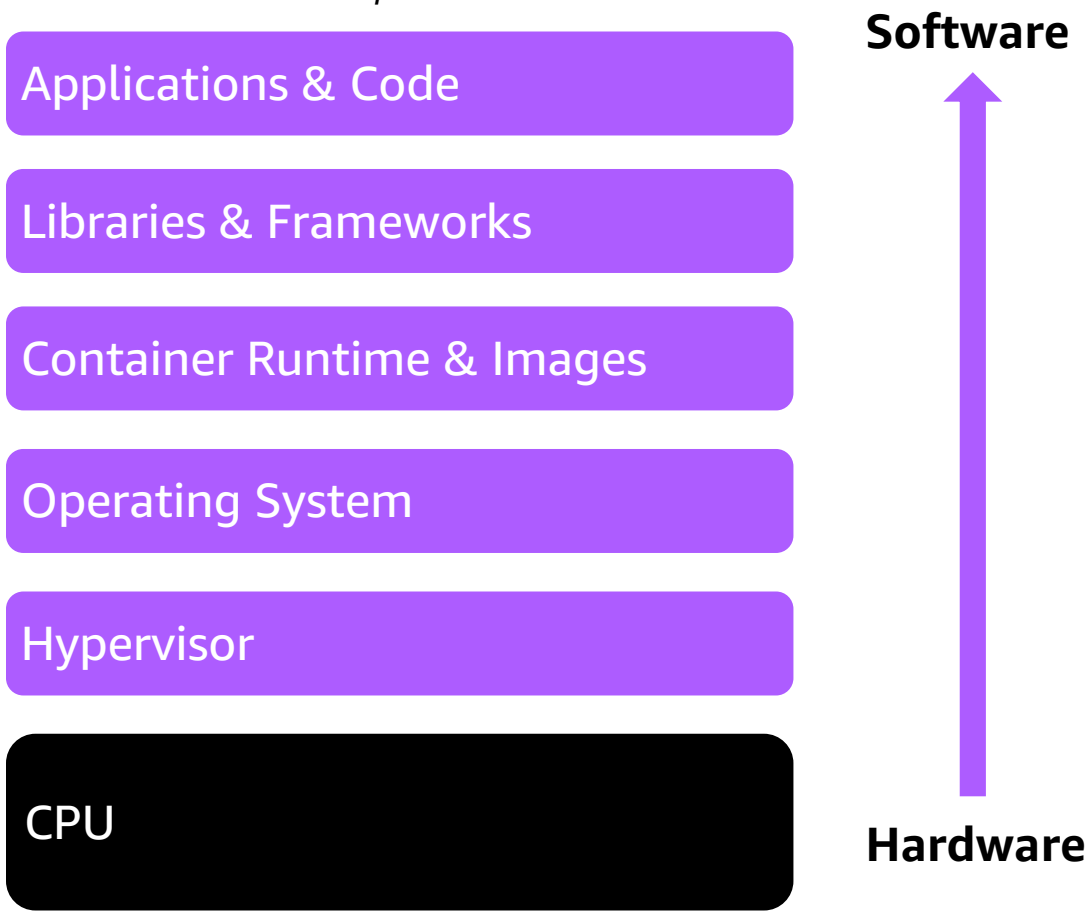
```
int sum() {
    int a = 10, b = 20;
    return a + b;
}
```

same code

-O0 (no optimizations)

Impact on the stack layers

*abstract representation**



Where should I start?

- Software someone runs for you
- Software you run
- Software you build & run

Software someone runs for you

You get- compatibility, migration path, support

You own- performance testing, rightsizing

A word about performance metrics

Prefer realistic load tests, compare $/request or $/job

Validate acceptable latency

CPU% will mislead you (due to SMT in x86)

You might need to adjust your scaling thresholds

Software someone runs for you

You get- compatibility, migration path, support

You own- performance testing, rightsizing

Note: minimal version, newer is better

Software you run

More control / more responsibility

Prefer newer OS, runtime, application version

Note: migration mechanism, daemon support

(in addition to performance testing)

Another thing about performance testing

Less moving parts

Compare both config and metrics

e.g., with github.com/aws/aperf

Software you build & run

Different binaries = different artifacts

Interpreted and JIT languages are easier

Native dependencies need arm64 support

Intrinsics in compiled languages

How can we estimate the effort?

Scan your repository for unsupported dependencies / intrinsics / …

github.com/aws/porting-advisor-for-graviton

Software you build & run

Update your CI pipeline to build arm64 artifacts

Avoid using emulators to build

Non artifact specific steps can happen once

Build/post-build steps can happen in parallel

Should we build multi-arch?

During migration to allow rollback

Enjoy arm64 where possible

When you customers need it (internal/external)

Trade-offs: pipelines are a bit more expensive,
test carefully (e.g., IEEE 754),
choose a scaling metric

What about containers?

One image per-architecture

Then create an image index

docker manifest create/push/inspect


```
{
  "manifests": [
    {
      "digest": "sha256:...",
      "mediaType": "application/vnd.oci.image.manifest.v1+json",
      "platform": {
        "architecture": "amd64",
        "os": "linux"
      }, ...
    },
    ...
  ],
  "mediaType": "application/vnd.oci.image.index.v1+json",
  "schemaVersion": 2
}
```

What about containers?

One image per-architecture

Then create an image index

Use Automatic platform ARGs in Dockerfile

Automatic platform ARGs example

```
...

ARG TARGETARCH
ARG TARGETOS

ARG GO_VERSION=1.21.7
ARG GO_URL=https://go.dev/dl/go${GO_VERSION}.${TARGETOS}-${TARGETARCH}.tar.gz

RUN curl -Lo /tmp/go.tgz ${GO_URL} \
   && tar -xzf /tmp/go.tgz -C /usr/local/ \
   && rm /tmp/go.tgz
```

We run a bunch of container images on k8s, which should we build/upgrade?

github.com/ArmDeveloperEcosystem/kubearchinspect

Are there reasons not to migrate?

Unsupported operating system

Unsupported runtime

Unsupported application

Unsupported feature

Are there reasons not to migrate?

Non-technical reasons

- License cost
- arm64 not available for your deployment option

Are there reasons to postpone migration?

Maybe, if you are running very old OS or runtime

AI tools can help here!

e.g., Amazon Q Developer /transform

Optimize

Right-size (can you use smaller / less VMs?)

ARM Performance Libraries

ARM Software Optimization Guide (SWOG)

AWS Graviton Performance Runbook

aws.github.io/graviton/perfrunbook

Summary

Migrating to arm64 can deliver better performance and lower cost

When doing performance testing, ask the right questions

This is getting easier, with application support, docs, and tools

#25emea-day2-track1

Thank you!

Nati Cohen
Sr. Specialist Solutions Architect, Compute
AWS
linkedin.com/in/natict