



Running Arm nodes with AWS Graviton on Amazon EKS

Michael Hausenblas, Product Developer Advocate
2020-08-17

Table of contents

- Why bother?
- Multi-arch Apps
- Arm-support in Amazon EKS
- Demo

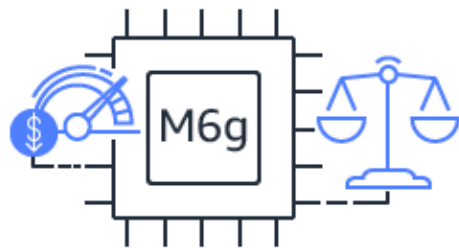
Why bother?

Benefits

- Best price performance
- Extensive ecosystem support
- Enhanced security for cloud applications

20% lower cost and up to 40% higher performance for M6g, C6g, and R6g instances over M5, C5, and R5 instances, based on internal testing of workloads.

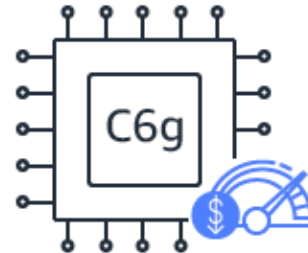
General Purpose



Best price performance for general purpose workloads with balanced compute, memory, and networking

Built for: General-purpose workloads such as application servers, mid-size data stores, microservices, and cluster computing.

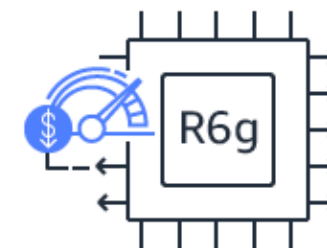
Compute Optimized



Best price performance for compute-intensive workloads

Built for: Compute-intensive applications such as high performance computing, video encoding, gaming, and CPU-based machine learning inference acceleration.

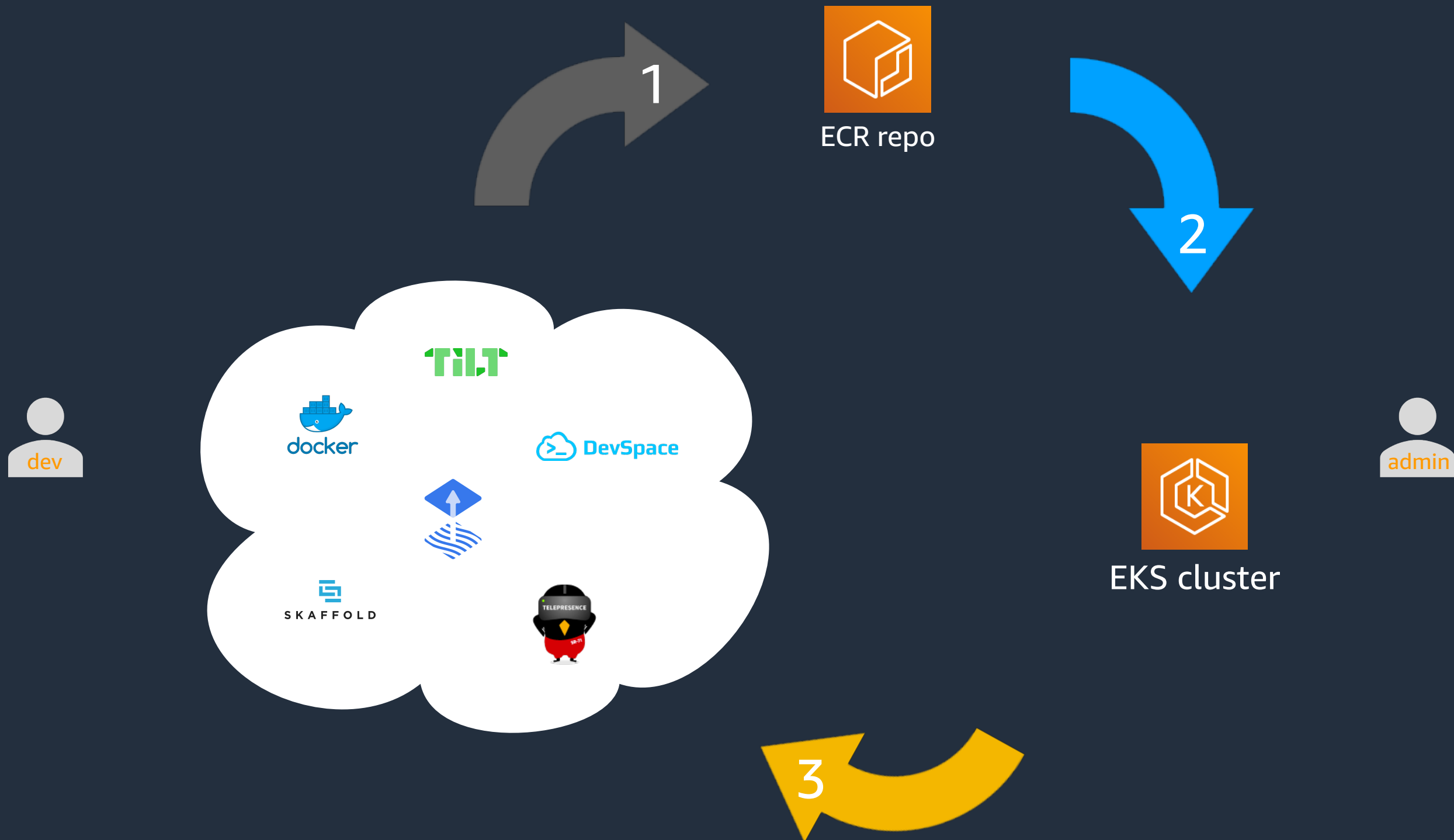
Memory Optimized



Best price performance for workloads that process large data sets in memory

Built for: Memory-intensive workloads such as open-source databases (MySQL, MariaDB, and PostgreSQL), or in-memory caches (Redis, KeyDB, Memcached).

Multi-arch Apps



Arm-support in Amazon EKS

AWS Graviton2 for Amazon EKS is GA!

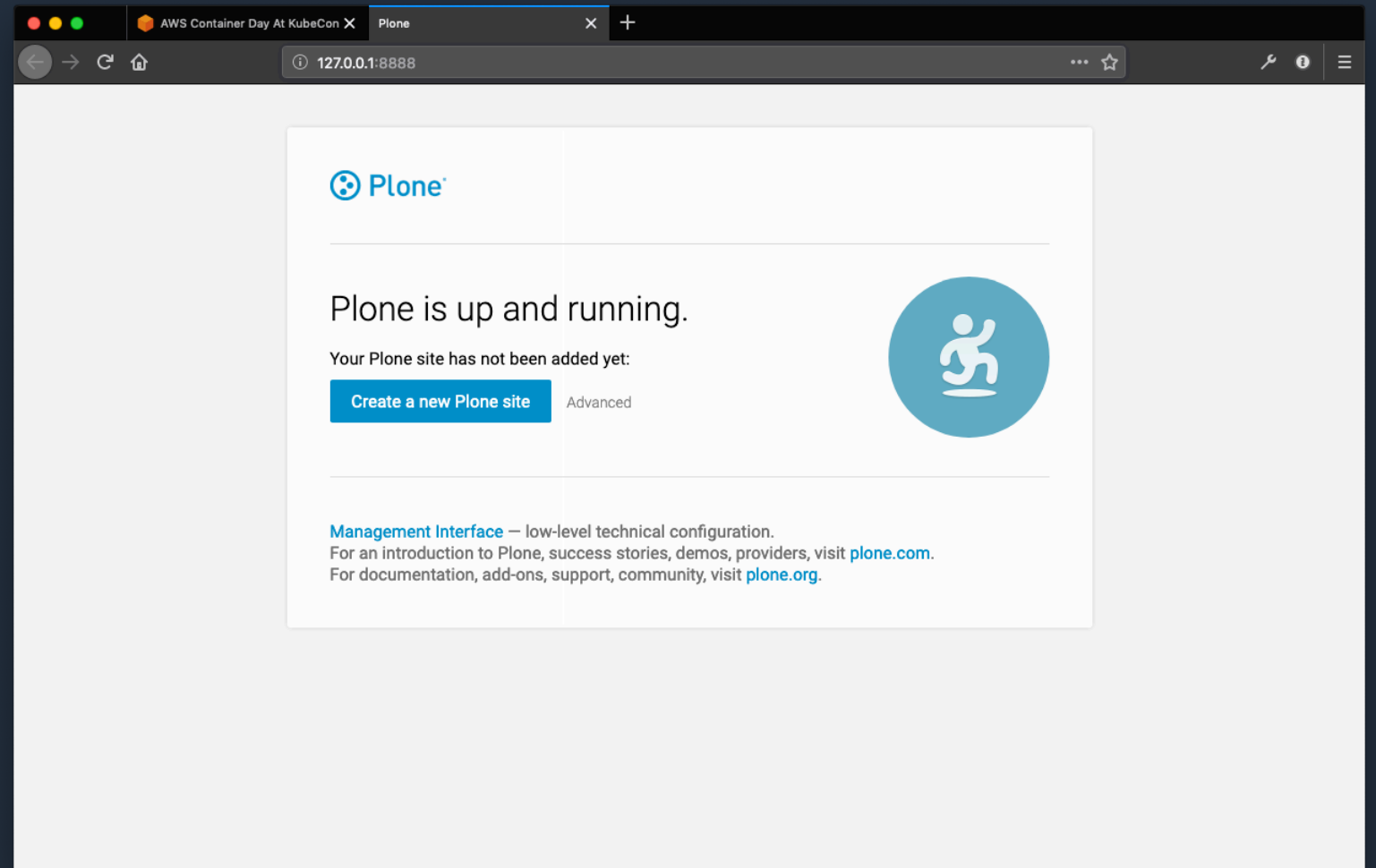
- Supporting Arm V8.2 (and others) architectures
- Along with [multi-arch support in Amazon ECR](#), end-to-end coverage
- Amazon EKS & tooling taking care of architecture-specific config
- Mixed managed node groups now supported
- Getting started: [aws/aws-graviton-getting-started](#)

Arm in Action

```

1  apiVersion: apps/v1
   kind: Deployment
2  metadata:
3    name: plone
4  spec:
5    replicas: 1
6    selector:
7      matchLabels:
8        app: plone
9    template:
10     metadata:
11       labels:
12         app: plone
13     spec:
14       containers:
15         - name: main
16           image: arm64v8/plone:5.2.1
17           ports:
18             - containerPort: 8080
19 ---
20 apiVersion: v1
21 kind: Service
22 metadata:
23   name: plone
24 spec:
25   ports:
26     - port: 80
27       targetPort: 8080
28   selector:
29     app: plone

```



\$ kubectl port-forward svc/plone 8888:80



Questions? Suggestions?

Ask on the chat or visit our virtual booth during KubeCon EU 2020!

Twitter: [@mhausenblas](https://twitter.com/mhausenblas)

Mail: hausenbl@amazon.com