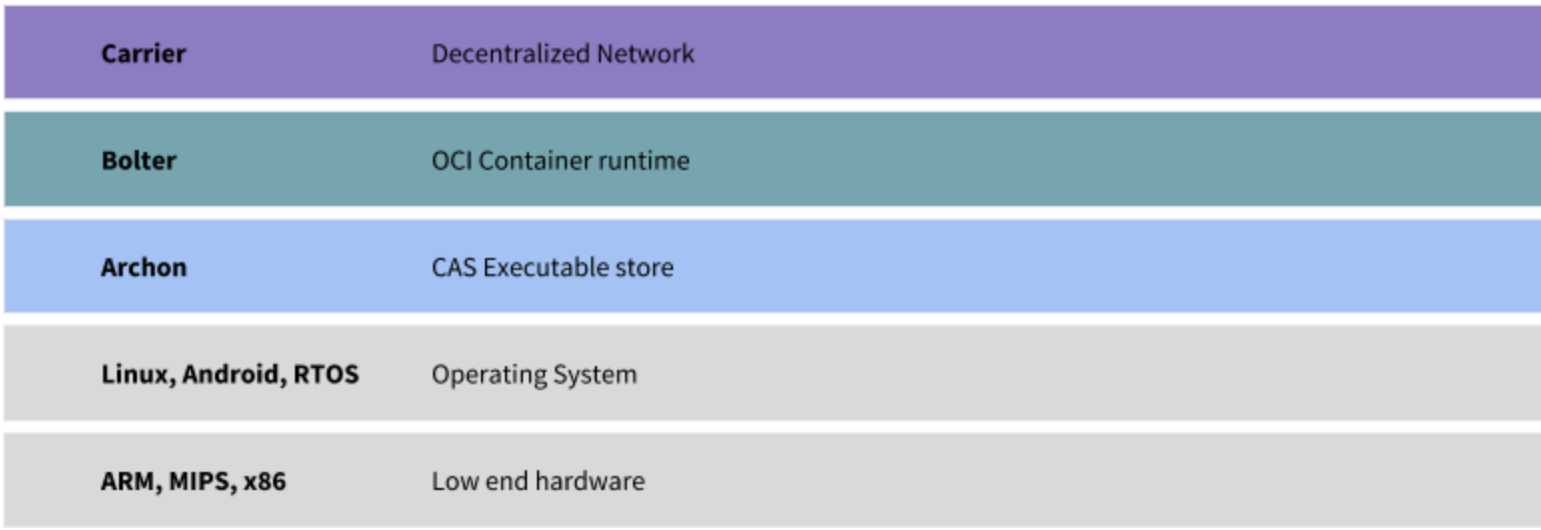
Application container orchestration for the distributed internet of things

Korhal is an open source toolbox for IoT developers dealing with **updates**, **configuration**, **monitoring** and **decentralized networking** of millions of devices.

Like docker, for IoT.



Motivation

The very way we develop software on embedded devices has not changed much since the advent of software. Our primary deployment vehicle are system-updates. It is considered a basic hygene feature (thankfully, nowadays) to be able to update devices over the internet on demand. There are many strategies for updates, and Korhal is a unique way of doing this in a way that is very distinct from traditional system deployment by both its technology and its workflow.

We enable safe and secure software updates through containerization. Similar to docker/kubernetes orchestration capabilities. But instead of a hundred servers, millions of IoT devices can receive a feature update with just a few commands, and be roll-backed just as easily. Or have 3 versions ready at once, all without requiring duplicated disk space. Essentially, you can have your own app-store, with highly efficient deduplicated storage.

The explosion of third party IoT cloud solutions are good for diversity, but we're convinced that no single provider can or should be in control of the entire infrastructure. Neither will a new standard truly become dominant which could enable simple migration of a device from one cloud provider to another, or one system provider to another, or one programming language to another.

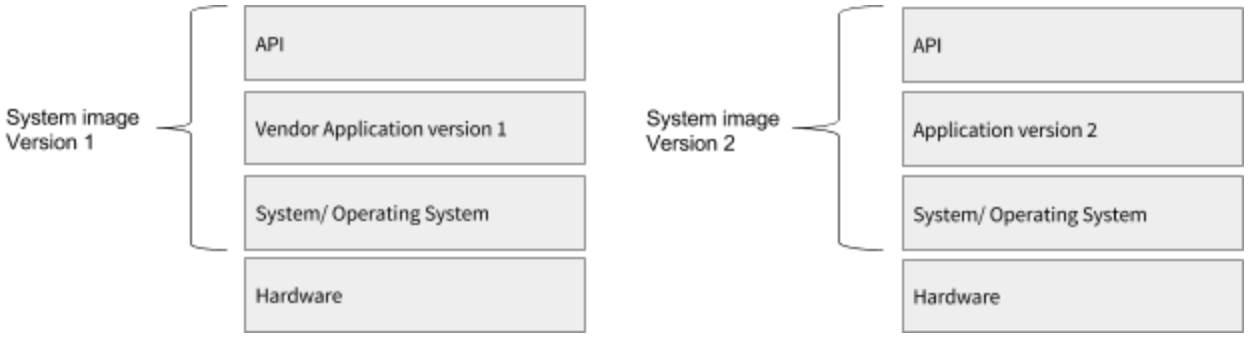
If anything, docker has taught us that the key to a competitive software ecosystem is the capability to exchange one component for another component on the fly without any of them having to implement a common standard.

Iot-Platform providers have a singular motivation to create lock-in through monopolization of infrastructure. They're not technology driven and can't in any way create meaningful cross-device standards if their incentive is to create an ecosystem they control.

What we present in this whitepaper is essentially a deconstruction of the IoT gatekeeper business models, and an alternative way of providing a radically different, decentralized, monopolization-resistant infrastructure that is in the control of it's user rather than the company building it.

Containers Instead of Systems

Abstraction in IoT is difficult. On one hand tight integration of application and system code is actually very desirable to maximize software usefulness for specific hardware capabilities. On the other hand, tight integration is well known to lead to instability, and inflexibility for developing new unthought of capabilities in the future..



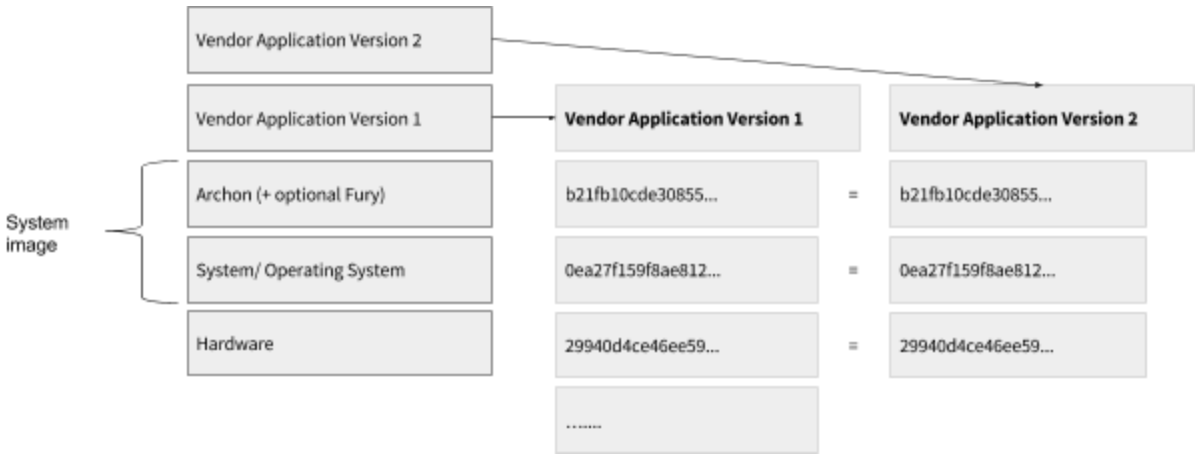
We learned from docker that the best bet for common standards to become dominant in practice, is to use what has always been stable: the berkeley socket api and the ELF abi interface. Both are so ubiquitous that almost all existing code is already compatible with any abstraction interface that solely relies on these two factors. In cloud computing we call the practical implementation of that interface into an easy to use toolset “containerization”.

Korhal Bolter is an **Open Container Initiative compatible** container runtime for embedded devices, specifically designed for devices without a Floating Point Unit, on which Docker cannot run. We’re planning to enable semi-virtualization on devices even without a memory management unit. With Korhal being part of the Linux Foundation family, we’re compatible out of the box with a wide range of industry standard embedded systems including OpenWrt, Buildroot and Yocto. However, any linux based system can be supported.
Bolter can execute OCI images, such as from Docker Hub, or even Archon executables for higher space efficiency and peer to peer image streaming.

```
$ bolter run amazon-aws/aws-iot-connector --region eu-west-1
> receiving archon index .. complete
> 4 shards with 924Kb (downloading 2 new shards with 43Kb)
> receiving shards from peers
  70637d03280af6127aa1b664be4cfdbe1d9c0c8c226129940d4ce46ee595741a
  ccc63915b21fb10cde30855ef75596c2564c3280ea27f159f8ae81400fa317eb
Welcome to AWS IoT
Connecting to the aws cloud...
```

Container Orchestration instead of System Updates

Korhal Archon is a **Content Addressable Storage** built for executable code, storing executables in a way that common artifacts are deduplicated for storage-space and link time efficiency. **Content Addressable Linking,** built on top of this is a new concept pioneered by Korhal's founders. In the Superscale Networks infrastructure with 10'000 MIPS devices, we're using it to modify arbitrary executables at runtime without having to update the system image. Archon updates can be rolled out immediately and **rolled back safely** in the same instance, enabling **continuous deployment** for constrained devices.



Binaries built with the Archon linker are location-independent, ASLR compatible and are linked statically, so that **no dependency management** is necessary on the target hardware. Instead, executables automatically bring in dependencies by addressing them by the hash of their content rather than the name of the original library. Updating a single executables dependencies does not risk incompatibilities with other executables on the system, since other executables will only share the code that matches the cryptographic hash of the code they were originally linked with.

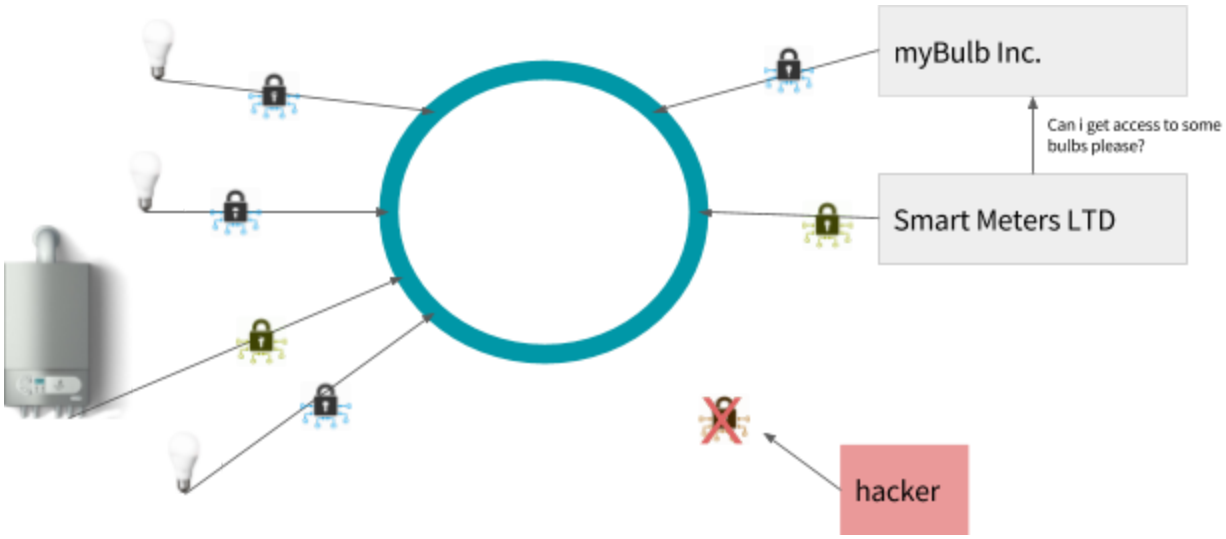
In practice this means an application update for a system of several megabytes, is a measly kilobytes in size download that is **CDN optimized.** This is due to its content-address-naming, which means it can also be shared via peer to peer networks such as BitTorrent or IPFS to significantly reduce server load for mass updates.

Decentralized Edge Access with Korhal Carrier

With cloud connected devices at scale comes **risk at scale.** While scaling stateless transactions like http are a well established art, scaling real-time streaming and always-on stateful command and control is still mostly arcane magic requiring specialists with experience in IoT or automated trading. Competition for those talents is fierce. Downtime of millions of IoT devices can be quite costly also. Yet, using such service from 3rd parties comes with its own cost of lock-in inflexibility, and lack of knowledge distribution.

Korhal Carrier suffers none of those issues. Being a **decentralized edge access** network of equal peers, we are not a cloud provider - You are, and so are all of your peers. All of our code is **open source,** so there are **zero contractual obligations** attached to joining a Carrier Ring. Neither do we offer a specific implementation of a management protocol, instead Carrier establishes peer to peer connections between your management platform and your devices, while allowing interchangeability of both.

The Carrier Ring acts like an airport hub for device connections. Each device connected to Carrier, is actually connected to the whole ring that can span multiple datacenters or even companies. Failure by malice or accident of up to half the ring can be tolerated and does not result in loss of access to those IoT devices.



Connecting a device to Carrier is trivial if you're already running Bolter:

```
$ bolter run korhal/carrier
> receiving archon index .. complete
> 18 shards with 329Kb (downloading 1 new shards with 2Kb)
> receiving shards from peers
  5cd8955da3d28a504aabf130beaebbcda2159e5babe02868e480904cf7cd09e4
Carrier has arrived
Generating private keys . . .
Connecting to public pool carrier.korhal.io as 8ae4b542974111eddd
Accepted proof of network from cxp43.europe.carrier.korhal.io
Ready
```

The public pool is for demonstration purpose only. For production grade safety we recommend joining a large production ring, such as the ones offered by a Korhal membership.

Each device has a unique identifier, which is a cryptographic hash of an asymmetric public key, by which it identifies itself to the ring, and by which the device vendor can connect to that device. Cryptographic anonymous access allows completely trust-free operation, where none of the server systems involved have access to any device data or company internal authentication keys.

Instead, Carrier establishes an end-to-end peer-to-peer connection with included NAT-traversal and firewall piercing between you and your device. This can be completely stateless, if you're planning to deploy only an http-api for example. Instead of having to connect the device to an http server, and have it continuously deliver data, it is now effort free to simply reverse the roles and expose an http server on the device via the carrier ring, which can be asked for data only when you require it.

All Open Source

Korhal is an open source tech company. None of what we do is designed to lock in customers, and we instead focus on added services for companies that wish to get certain guarantees like SLAs for Korhal Carrier, pre-built images for their platform, exclusive training content, custom engineering, or voting rights on development priorities.