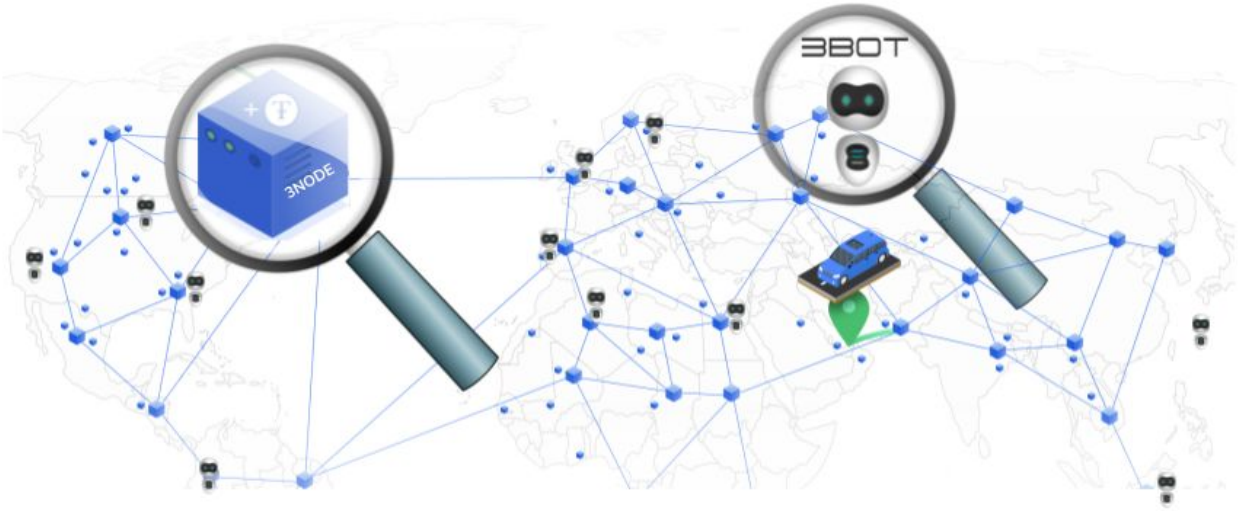
THREEFOLD.TECH



Grid of Compute, Storage and Network Capacity, we call this a ThreeFold Grid.
Deployed capacity in private & public context: 15.000 CPU cores + 40.000.000 GB

"IT" CAPACITY IS THE OIL THAT FUELS THE DIGITAL ECONOMY

ThreeFold Technology Whitepaper:

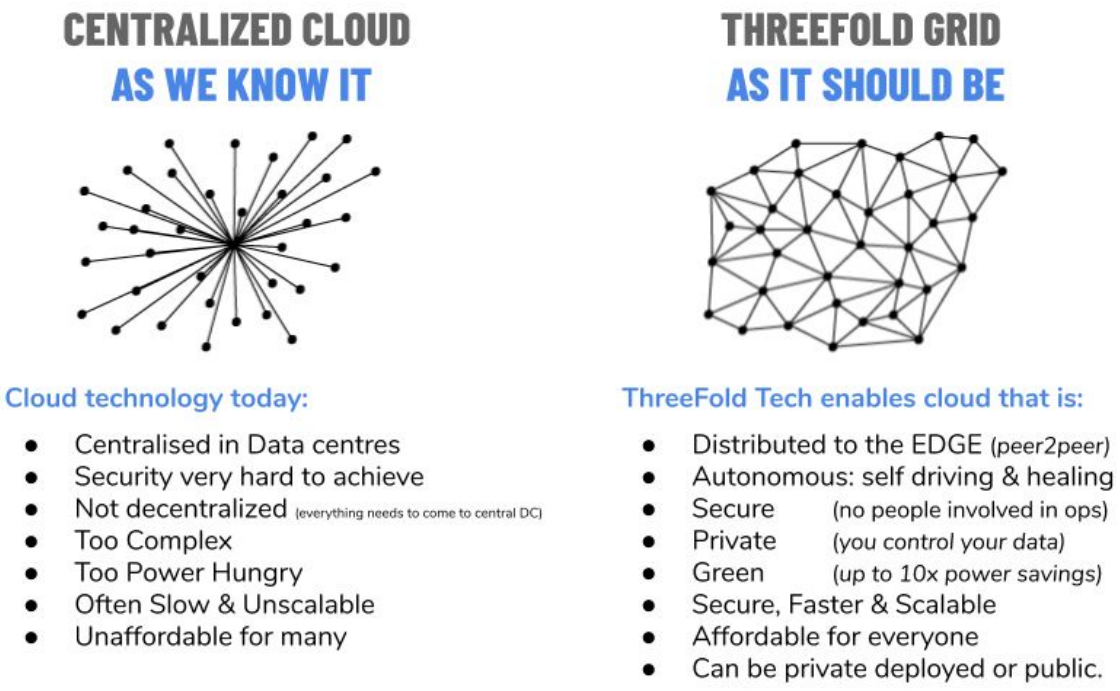
Capacity Layer

Author: Kristof de Spiegeleer
Version: 1.2 (2020)

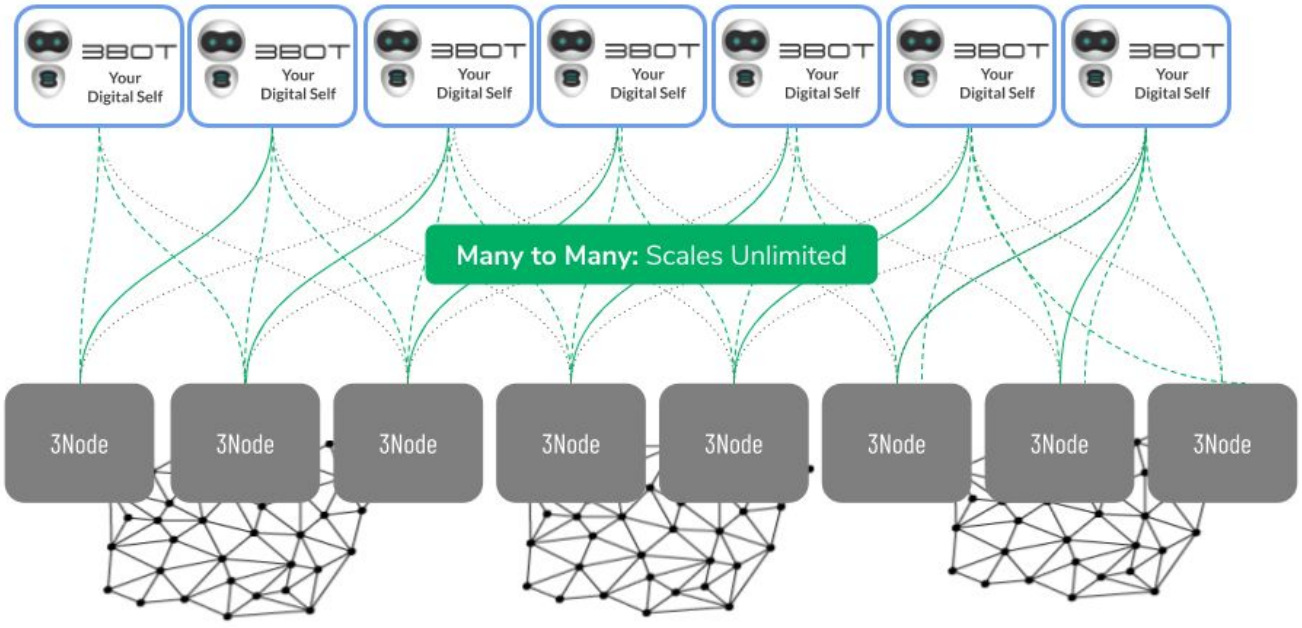
Table of Contents

Introduction

Today, IT capacity is mainly being delivered by means of centralized cloud technology which gets deployed in a private or public context. ThreeFold Tech has created technology to make the cloud 100% decentralized.

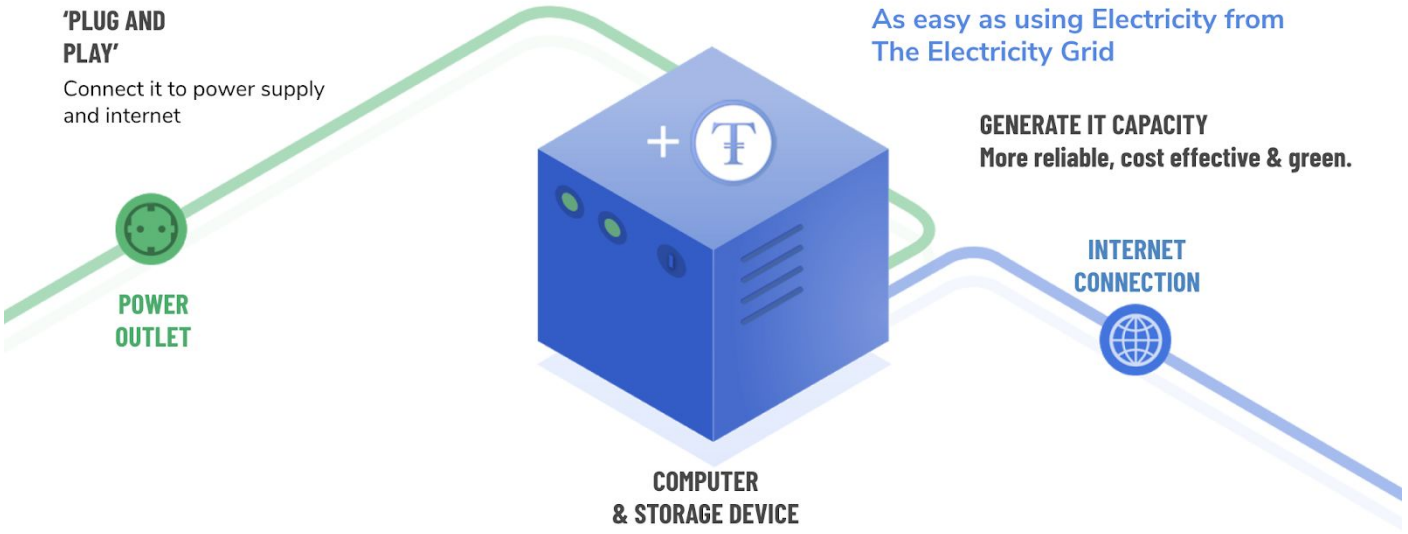


Our solution has two components: a 3Node, which is the box which can deliver capacity in a very compute/storage and power efficient manner, and the 3bot, which is our "digital self" – it is the decentralized component which makes autonomous IT possible and allows the grid to scale forever.

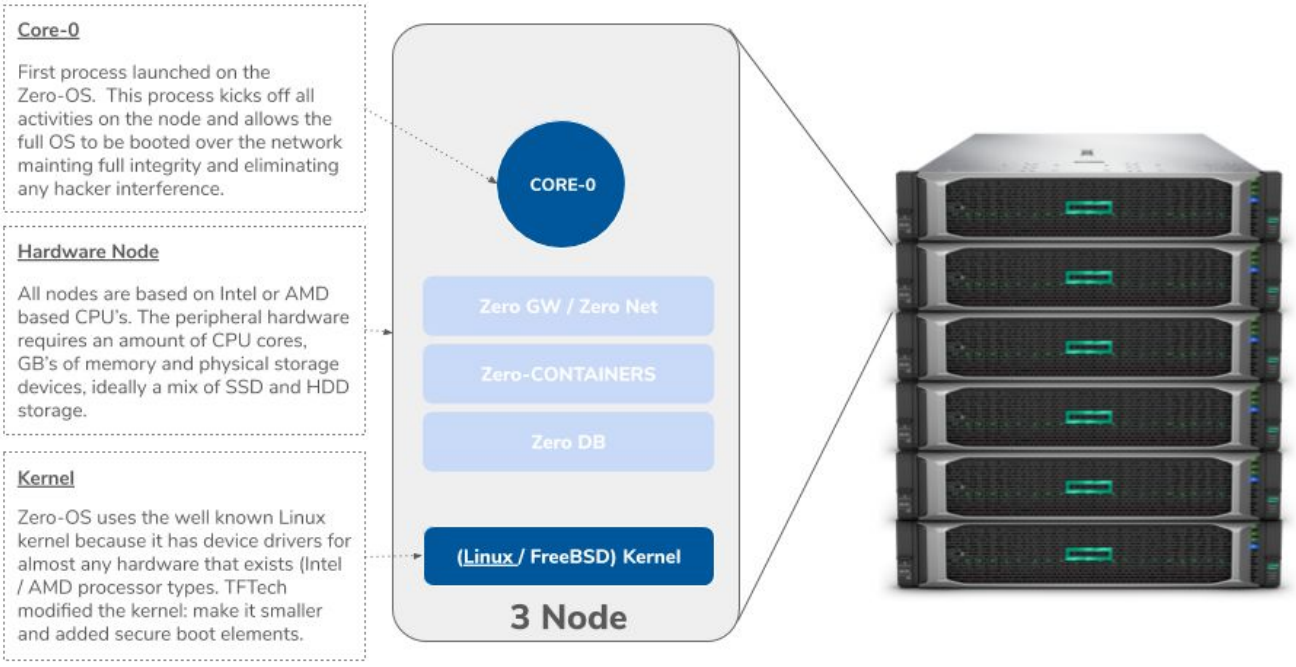


The 3bots are the coordinators which can run 100% autonomously – they are super intelligent digital creatures who know how to protect your data and how to deploy/manage any IT workload on the capacity layer, which is the grid of the 3Nodes.

The 3Node

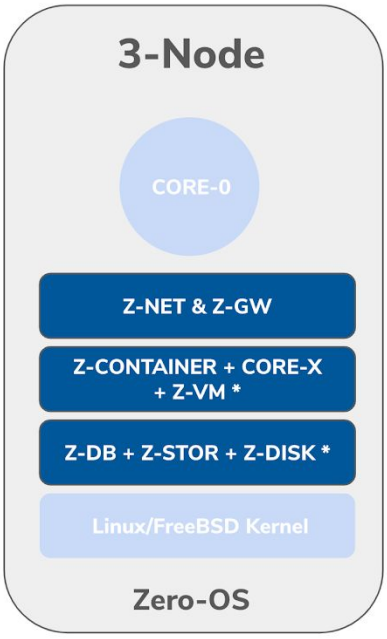


The 3Node is a physical box which is connected to the Internet or a local network. A 3Node can be deployed privately, as well as for high-security use cases. The 3Nodes deliver the required capacity for the TF Grid.



The 3Node boot process is ultra-safe and cannot be intercepted by hackers. It is designed in such a way that no person, not even ourselves, has access to the internals of the Zero-OS.

We do this by means of strong integration with the physical hardware security and boot capabilities (e.g. secure-boot). There is no shell in Zero-OS, no user interface, and no RPC layer (remote procedure call).
The only way how the Zero-OS can be commanded is by means of the Blockchain Database.



Z-CONTAINER

Container technology.
Compatible with Docker.
Much more efficient storage layers.

Core-X

Secure process manager which can be accessed by sysadmin 3Bot if required.
Web & SSH interface if required (optional).

Z-KUBE

Kubernetes Node. Integrates with QS-NET & QS-FS (3.0)

Smart Contract For IT Agent

Deploy IT workloads based on blockchain & peer2peer requests.
Enforces multisignature & strong verification of workloads.

Quantum Safe Network (QS-NET)

Overlay network.
Allows all containers to talk to each other in all privacy & security.

Quantum Safe Filesystem (QS-FS)

Signed & Fingerprinted File Parts
Deduped, Cache per ZOS
Hacker Safe, cannot modify files.

Z-DB

Optimized storage engine technology.
Knows how to write most efficient to a specific medium like SSD or HD.

Z-DISK & Z-VM (*OEM ONLY)

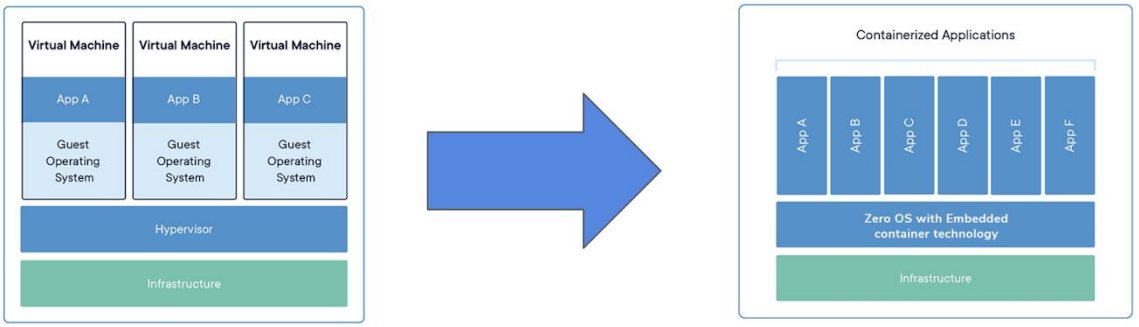
High performance Block Device & Virtual Machine Technology. Run any operating system at the edge.

Core-0 is our main process manager which will deploy and manage our primitive workloads. There are three types of workloads: compute, storage, & network.

3 Node Primitive Workloads

The primitive workloads which are delivered by the Zero-OS:

Compute = Containers-Based



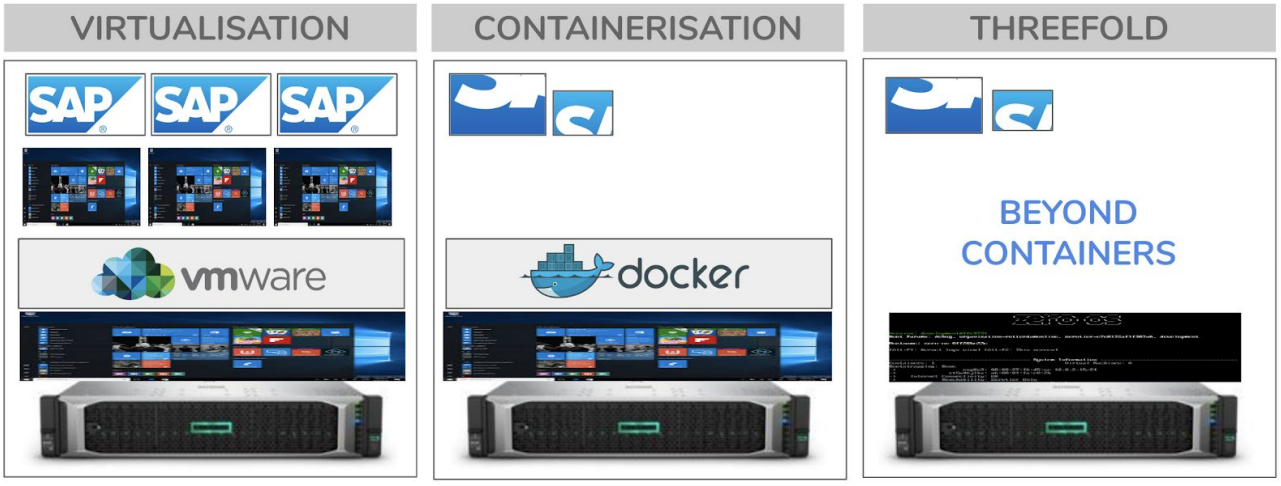
HARDWARE VIRTUALIZATION

The increase in hardware capabilities allowed for multiple Operating Systems to exist on one hardware platform. Monolithic central applications did no longer require dedicated hardware - SHARED HARDWARE

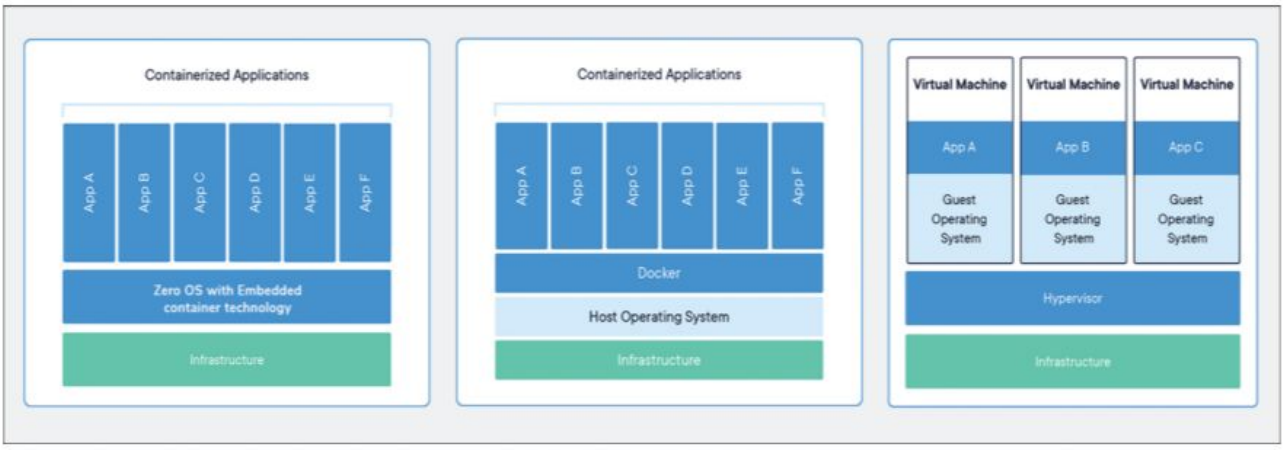
OS VIRTUALIZATION

Monolithic central applications are being overtaken by small, agile functionality “0-CONTAINERS” that are good at performing a few simple tasks (data collect -> analyse -> report). Operating System virtualization is needed to make this efficient. Aim is to have as few as possible layers between the app and the operating system while maintaining high level of security.

The world started with hardware virtualization (virtual machines). It was a big step in the right direction but has lead to many layers of complexity, and total cost of ownership did not get lower. Security is also a big issue.



Containerization is the current wave for deploying compute applications. It is more flexible and has big efficiency benefits, but complexity and security remain a big issue. A high level of automation can be achieved but it is a very centralized model where centralized managed applications are the big boss.



THREEFOLD

ZERO CONTAINERS
Docker import feature
Dedupe, thin provisioned virtual FS
Redundant Virtual Disks support
Public, private container marketplace
Security through signed workload defs

docker

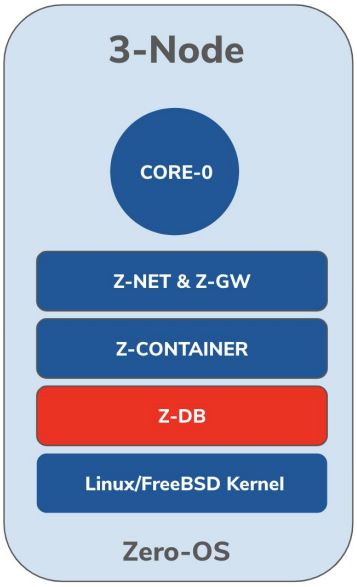
STATEFUL CONTAINERS
Docker images
Download / compile required images
Public hub and private store concept
Version control through encryption

vmware

VIRTUALIZED CLOUD
Virtual machines.
Example vendors Microsoft, VMWare, ...
Complex and not very efficient.
Still very manual approach.

In our Zero-OS, we have eliminated lots of layers and as such we can be much more efficient and we are not dependent on third-party software vendors. Our operating system is not managed by humans. It is an autonomous system which gives you the ability to run any container workload in all safety while achieving more performance and efficiency. Our container technology is compatible with docker, yet still has quite a lot of great benefits.

Storage = Zero-DB

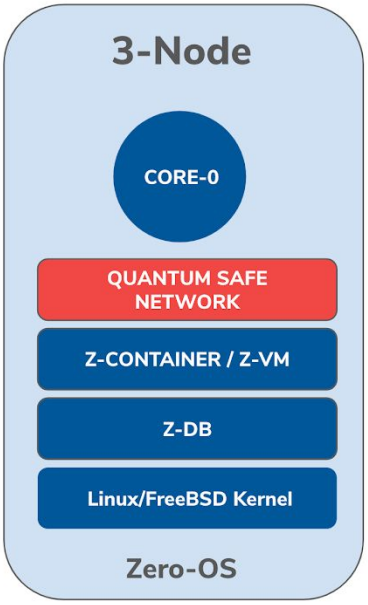


Backend Storage Engine

- Can do +50.000 transactions per second
- Can work on SSD & HD
- Optimized for easy (soft/green) operation on HD
- Always append store (can keep unlimited history)
- Master-Slave replication

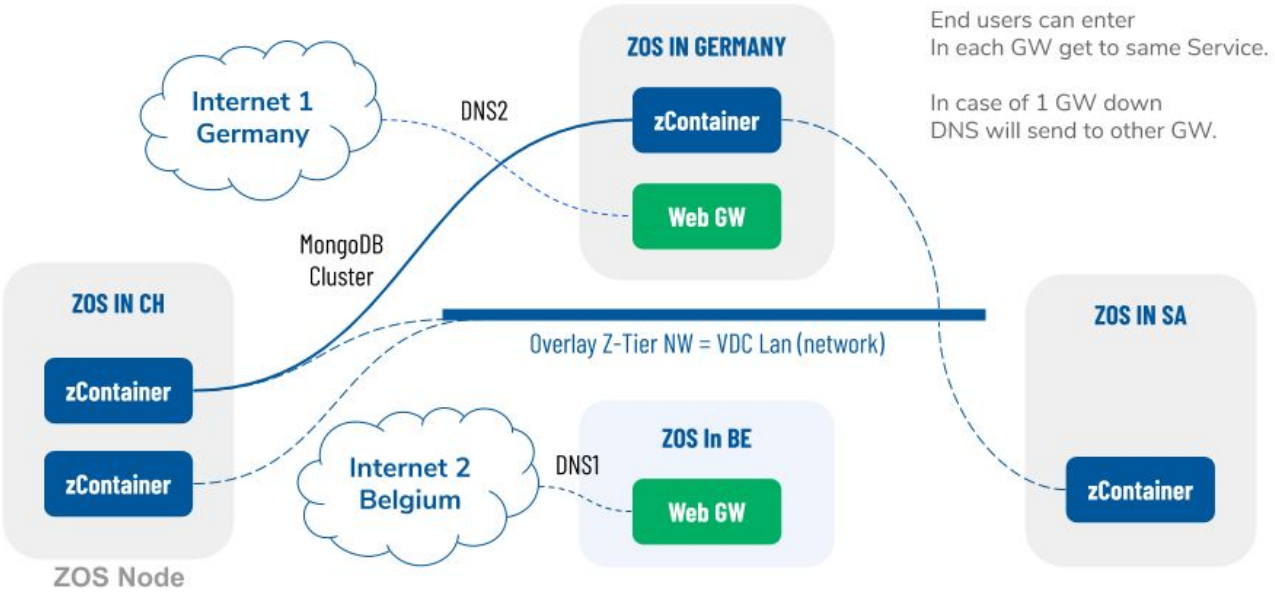
0-db is a super fast and efficient key-value store redis-protocol (mostly) compatible, which makes data persistent inside an always append datafile, with namespaces support.

Quantum Safe Network



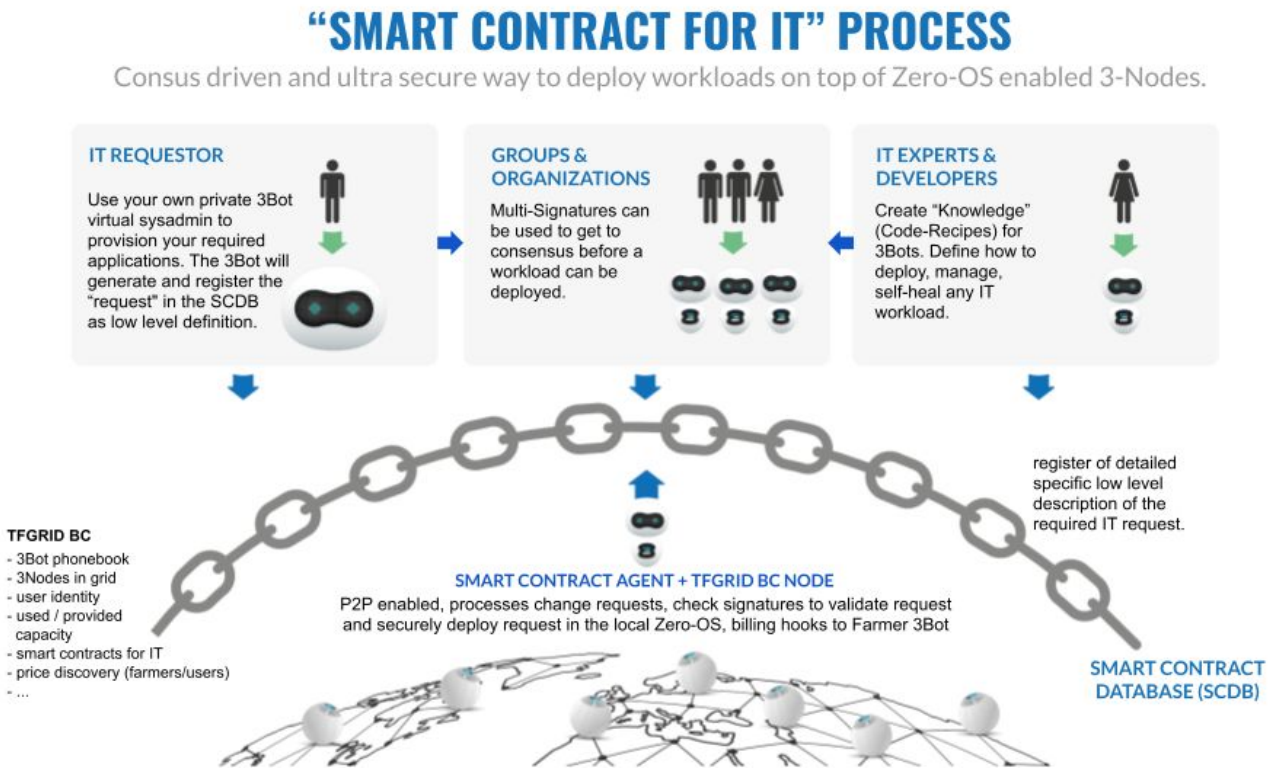
QUANTUM SAFE NETWORK

- Interfaces
  - VXlan
  - IPv6
  - Bridge to Public Network
  - WireGuard
  - HTTP(S) proxy
  - ...
- end2endencrypted overlay network between all IT workloads as running in Zero-OS
- Scales to planetary levels.



Are overlay networks connecting all the containers. They can exit on multiple areas using the gateways. Here we called them web gateways but many types of access methods can be used (e.g. VPN technology or port forwarding). This allows for achieving full network and systems redundancy.
In the above picture, any datacenter or network gateway can fall away, and the solution will still be available.

Self-Driving “IT” by means of an “IT” smart contract



Autonomous self-driving IT is possible when we adhere to a few principles from the start:

- Information technology architectures are configured and installed by bots not people
- Human beings cannot have access to these architectures and change things.

When we stick to these two principles we have to do a lot of complicated work and thinking upfront - before deploying any architecture components and seeing any benefit. While a typical IT setup today is based on trial and error continuously improving the installation to a point where everything works the basis of self driving IT is that everything needs to be considered and described in a “contract” type format before deploying any component. Then and only then you can deploy self-driving and self-healing applications on a grid of capacity

A system that enforces these two principles if the TF Technology stack. It works as follows:

STEP 1: IT Experts create smart contracts:

IT experts create smart contracts describing what needs to be done in order to deploy this architecture. The smart contract has to be specific and describe each little detail of the IT architecture. The experts create knowledge for the 3bots (it's like god defining our DNA of our cells)

STEP 2: Business and or Enduser customers consume smart contracts:

Users have digital needs and in order to procure services for their digital needs they will find smart contracts describing applications (application setups) meeting their needs. Consumers will instruct their 3bot to deploy an IT workload following their requirements buy using a smart contract created

- e.g. give me an archive of 1 PB in CH, e.g. deploy a CRM for 100 users, …
- e.g. deploy my new banking app feature X
- e.g. deploy my artificial intelligence data mining job for …

STEP 3: The 3bot registers the smart contract:

Creates & Registers the "IT" smart contract. This is a complete end-to-end deployment cycle for all sort of IT deployments - both simple and complicated, bound to one location or many. It will then leave instructions for the nodes in a digital notary system in order for nodes to be able to grab instructions on what they have to do in order to meet smart contract completion.

STEP 4: Business IT Workload Stakeholders:

is optional but when required stakeholder can be defined to give consensus and sign off on the successful execution of the "IT smart contract" delivering the appropriate digital service. Stakeholders can be defined in a "multi signature" blockchain to provide sign off on regulatory, commercial and other business requirements. Approvals can include IT expert checks the quality of the code, a legal guy checks GDPR, a business person checks budget etc.

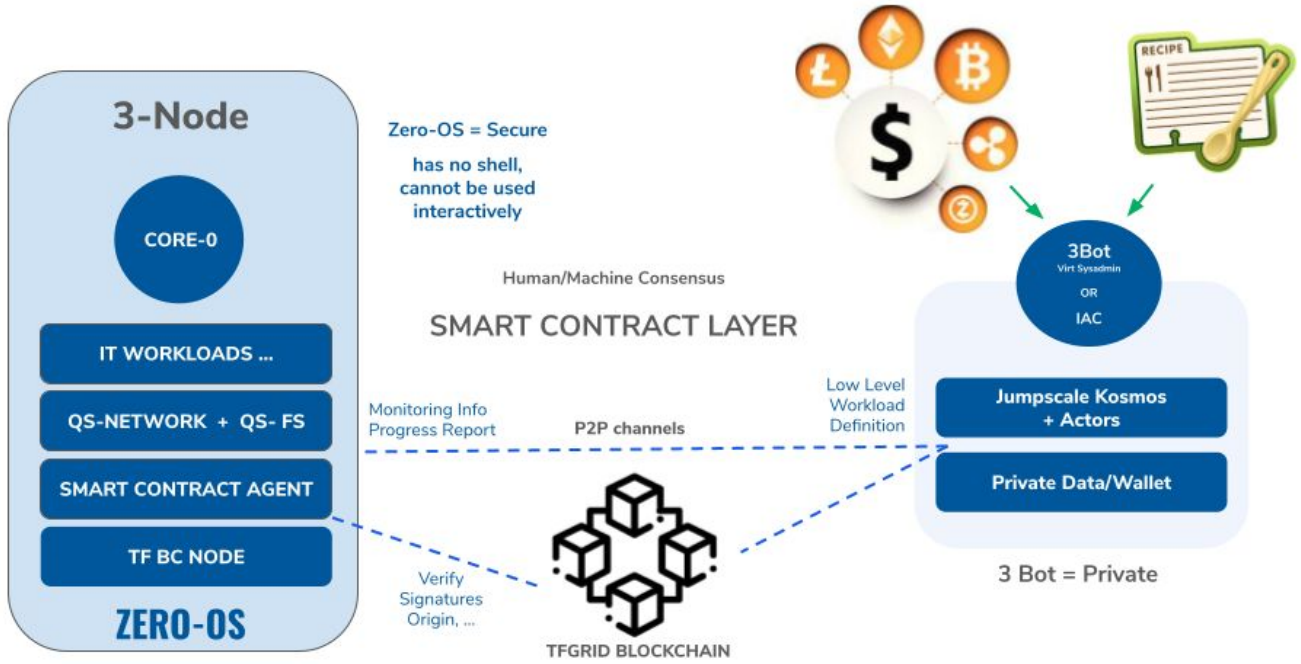
STEP 5: The capacity layer: 3 Nodes…

- thousands of 3 nodes can work together to execute and deliver the "IT Smart Contract" (if required)
- verify if consensus was reached between the business stakeholders
- verify the validity of the smart contract and download the "IT workload definition"
- download the right files to execute the smart contract and each file gets verified (signature)
- run the required processes and again signatures are checked to make sure the workload is pure.
- ensures that no person (hacker or IT person) can ever gain access or influence on the

execution process.

Architecture

Provisioning Flow



Is for our v3.x series.

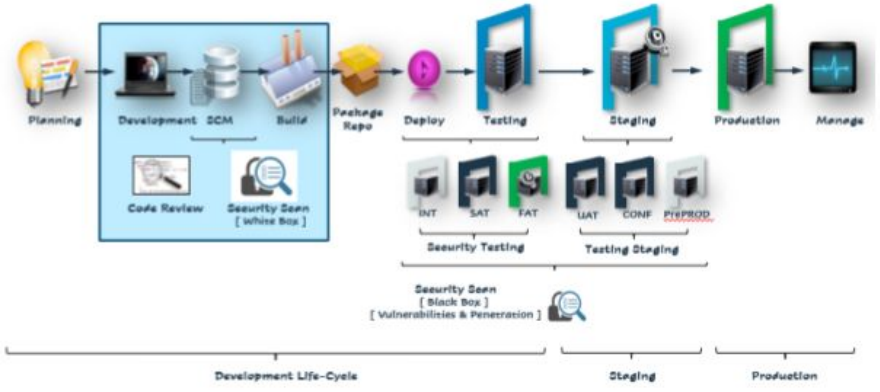
Kosmos is our scripting environment (see jumpscale) which allows you to deploy any workload in a very efficient easy manner. The Kosmos shell is very easy to use and can be installed on any platform. Kosmos and the Kosmos Actors will create a low-level “workload definition”. After verification by humans and or machines the Zero-OS will pick up the workload and do the local provisioning.

A 3bot has been educated by means of recipes and has an integrated digital wallet. These two resources are needed to be able to deploy any workload in the field.

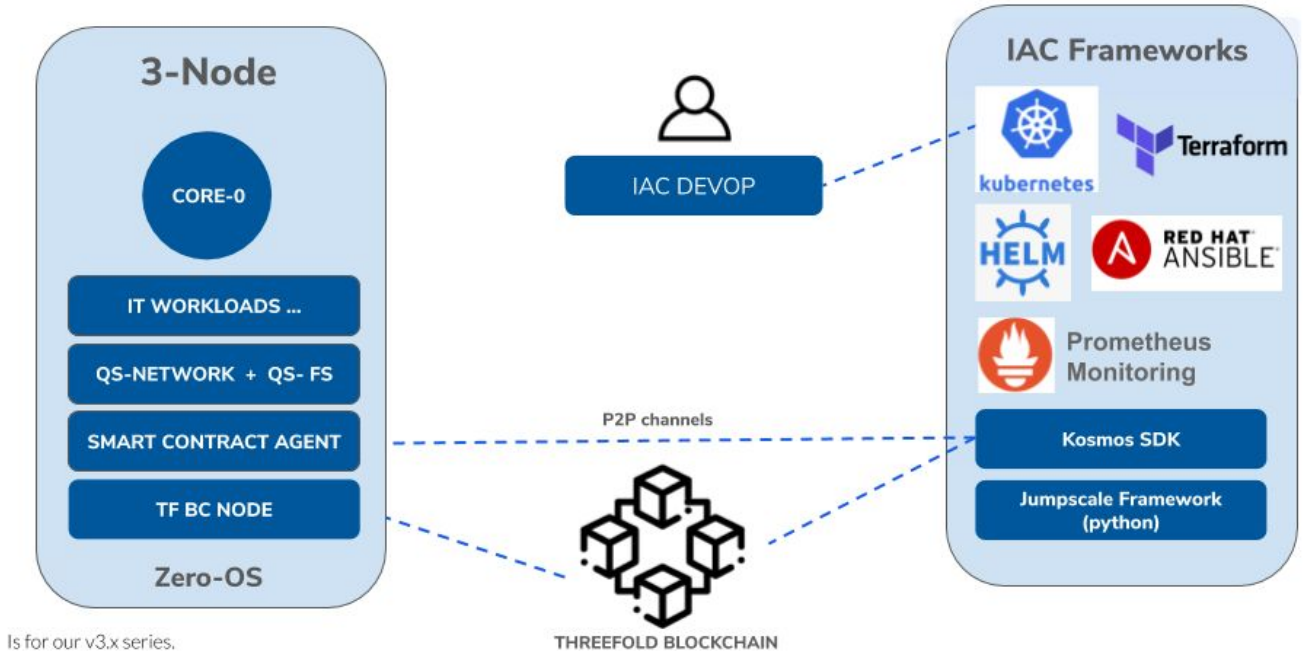
Infrastructure As Code

DevOps = the future is here today.
IAC = Infrastructure As Code

DevOps is a process framework that ensures collaboration between Development and Operations Team to deploy code to production environment faster in a repeatable and automated way. ... In simple terms, DevOps can be defined as an alignment between development and IT operations with better communication and collaboration.



THREEFOLD.TECH



Is for our v3.x series.

THREEFOLD

JumpScale Powered

Jumpscale is a cloud automation platform designed for scalability and fast development with a low memory footprint. It does that by providing very easy-to-use SALs (System Abstraction Layers) language which makes development cross-platform very efficient and unified, as well as easily adaptable to technology changes by exposing an abstract interface.

The main components of the jumpscale automation suite are:

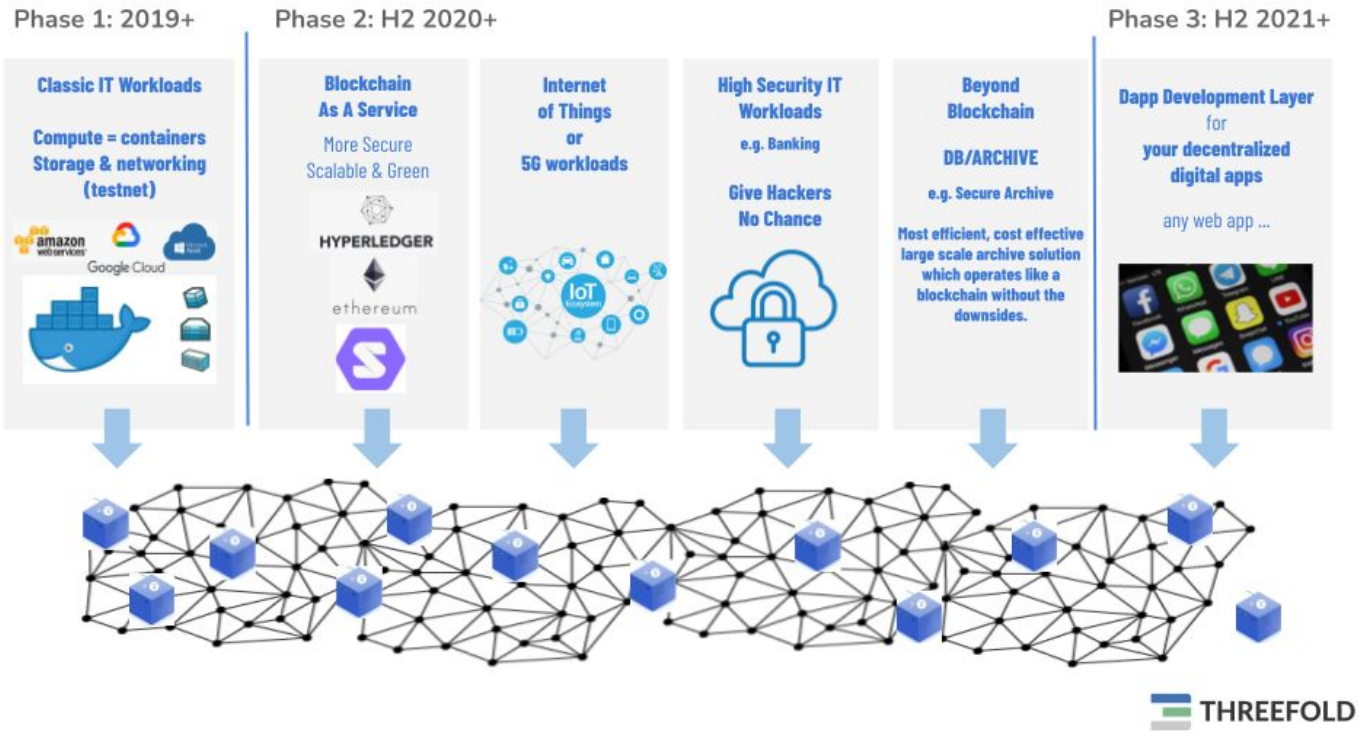
- **Clients:** User-friendly clients to connect to a multitude of systems (e.g. SSH, GitHub, FTP).
- **SAL:** System Abstraction Layer – it is a DSL (domain-specific language) for talking to a system. This allows the developer to use a user-friendly human-looking language when developing the life cycle management actors in Kosmos. This is the main bulk of jumpscale.
- **Kosmos Actors**: Executing the intelligence required to get to full self-healing & self-driving. Each actor is responsible for managing one or more services and it is the only location where configuration information resides. Think about them as virtual employees being responsible for a certain part of the universe which needs to be automated.
- **Kosmos Shell:** Shell to interact with the actors.
- **Config Manager:** A secure way to manage configuration instances. Anything saved to the file system is NaCl encrypted and only decrypted on the fly when accessed.
- **Executors**: Jumpscale comes with its own executors that abstract working locally or remotely. Including:
    - SSH Executor (for remote execution)
    - Local Executor (for local execution)
    - Docker Executor (for executing on dockers)
    - Z-Container Executor (happens over SSH & CoreX)
- **Startup Managers**: Manage running processes in corex, tmux, etc.
- **Builders:** A set of tools to perform the common tasks required when creating container images. We call these images a “Flist” for file list.
- Lots of **tools** to automate your daily life as an expert IT person.
- **Schema & Data management** layers

Configmanager
The config manager in JSX provides an immediately-available security layer for configurations. It manages sensitive data, only decrypting during runtime when needed. It also has a recovery system built-in – built to facilitate creating, retrieving, deleting, importing, and exporting configurations.

Kosmos
Kosmos is a new way to interface with all of JSX’s tools, SALs, clients, servers and builders. It gives full control over both the functionality and the instances created seamlessly, making building a deployment world and walking around all its components very simple.

Builders
A big part of deployments is managing built solutions or components, which JSX makes very simple. Using JSX builders, a developer can build, install, and sandbox their solution in no time. They can then create flists, which are lightweight metadata descriptions to be mounted in the filesystem.

Use Cases



Cloud Workloads

Phase 1 and 2 are active today:

- Docker
- Kubernetes
- Quantum Safe Storage System
- Web gateway = http(s) proxy to any group of containers at the back
- ZDB service = our own storage service on SSD and HD

+90% of all workloads which can run on Google cloud, Amazon, etc, can be made to run on the ThreeFold Grid.

Example Packaging

Min configuration for 2U = 10cm height
32 core CPU (Intel /AMD)
256GB of RAM
1.9 GB of SSD
96TB of HDD

(Example on the left is for 4U)

Example Hardware Specs
Intel i7 CPU
32 GB memory
2 TB storage
128GB high speed SSD

Threefold Tech is hardware vendor-agnostic as long as it meets the minimum requirements specified above (small box). However, Threefold has partnered with HPE to offer to our farmers certified hardware that runs our technology to generate highly secure certified capacity and is fully supported by HPE.

Why can we be so efficient?

30 years ago:
Commodore 64

	1987	2015	improvement
memory capacity	1	8,192 MB	8,192
harddisk capacity	5	262,144 MB	52,429
cpu	7	11,059 MHZ	1,580
perceived performance	1	5	5
graphics (horizontal pixels)	640	2560	4

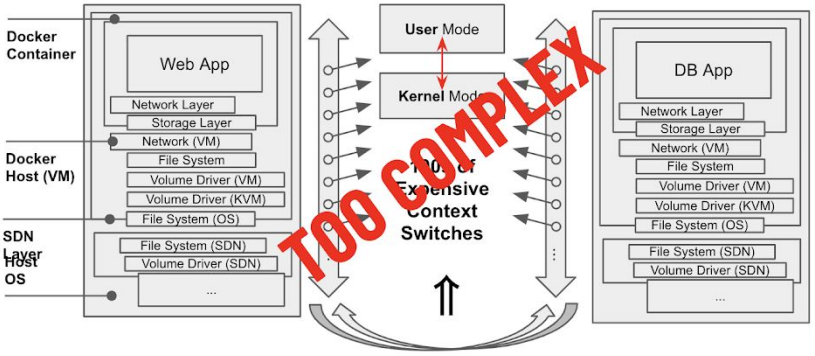
Today:
Macbook Pro

Hardware improved 10,000 times

But, user features and performance improvement improved < 10 times

Why?
- Education: Too influenced by IT vendors, too many abstraction layers
- Large IT companies: Acquisition strategy forces them to integrate = complex layers
- IT startups: Painkiller approach pays off
- It is easier to build layers on top of other layers instead of rewriting

Too many abstraction layers
MEANS
loss in efficiency &
performance
MEANS
increased cost of
management and scalability
of technology.



Current information systems are very complicated and the result of years of continuous growth and expansion to meet the ever-growing demand. Organic growth of IT systems results in adding layers of software integrating the old and the new. These added layers of software integrating old and new come at a price - they consume compute and storage capacity without adding additional end-user capabilities. It adds to the overhead of running an IT system creating a lot of context switches for the processors.

Context switching is the process of storing and restoring the execution state (context) of a process so that execution can be resumed from the same point at a later time. Each layer of abstraction involves multiple expensive user and kernel-mode context switches. Even a simple network interaction between two applications results easily in hundreds of context switches.

Inevitably this increases the inefficiency of IT architectures just like what happens with us, people, are disturbed every 5 minutes by a request to do an additional task.

Eliminating layers and using other protocols and patterns to avoid context switches has a huge benefit. This is not the only trick we do – we also work on algorithm layer (e.g. for the storage, see the autonomous whitepaper).

White Paper Takeaways

1. To deliver the required compute and storage capabilities for the exponential growth in digital services, Information Technology needs to be substantially overhauled so that it can exist everywhere – in datacenters, in homes, in cars, in lamp posts, in mobile masts, without having local or remote administration needs (autonomous).
2. Efficiency gains are mandatory to deliver the exponentially-growing needs. Hardware innovation does not follow Moore's law anymore and we cannot rely on hardware to fuel the increased needs.
3. A new technology paradigm is needed separating utility (compute and storage) operations from complex application deployments. Compute and Storage need to become the "electricity" that fuels the information age.

More Information

see https://wiki2.threefold.io/#/whitepapers