Hyperledger & Smart Contracts

Manoj S P

Open Source Solution's Specialist

manojs@sg.ibm.com

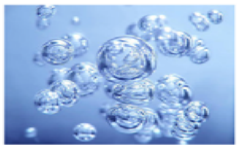
Agenda for Discussion ..

- Hyperledger – Introduction, Architecture and Ecosystem
- Blockchain Internals
- Smart Contracts & Containers

The Buzz ..Every one are talking about them

Microservices - Not A Free Lunch!

TUESDAY, APRIL 8, 2014 AT 8:54AM



This is a guest post by Benjamin Wootton, CTO of Contino, a London based consultancy specialising in applying DevOps and Continuous Delivery to software delivery projects.

Microservices are a style of software architecture that involves delivering systems as a set of very small, granular, independent collaborating services.

Though they aren't a particularly new idea, Microservices seem to have exploded in popularity this year, with articles, conference tracks, and Twitter streams waxing lyrical about the benefits of building software systems in this style.

Adopting Microservices at Netflix: Lessons for Architectural Design

February 19, 2015

TONY MAURO

Categorized

Tech

Related Posts

Popular Posts

Announcing NGINX Plus Release 6 with Enhanced Load Balancing, High Availability, and Monitoring Features

Socket Sharding in NGINX Release 1.9.1

Adopting Microservices Lessons

In some recent blog posts, we've explained why we believe it's crucial to adopt a four-tier application architecture in which applications are developed and deployed as sets of microservices. It's becoming increasingly clear that if you keep using development processes and application architectures that worked just fine ten years ago, you simply can't move fast enough to capture and hold the interest of mobile users who can choose from an ever-growing number of apps.

Switching to a microservices architecture creates exciting opportunities in the marketplace for companies. For system architects and developers, it promises an unprecedented level of control and speed as they deliver innovative new web experiences to customers. But at such a breathless pace, it can feel like there's not a lot of room for error. In the real world, you can't stop developing and deploying your apps as you retool the processes for doing so. You know that your future success depends on transitioning to a microservices architecture, but how do you actually do it?

Fortunately for us, several early adopters of microservices are now generously sharing their expertise in the spirit of open source, not only in the form of published code but in conference presentations and blog posts. Netflix is a leading example. As the Director of Web Engineering and then Cloud Architect, Adrian Cockcroft oversaw the company's transition from a traditional development model with 100 engineers producing a monolithic DVD-rental application to a microservices architecture with many small teams

NETFLIX

Stephan Kraus @donmumpleo · 8h
Start reading part 1 of how we're building a #Microservices architecture @ otto.de - wp.me/p3fb2R-u6

Stephan Kraus

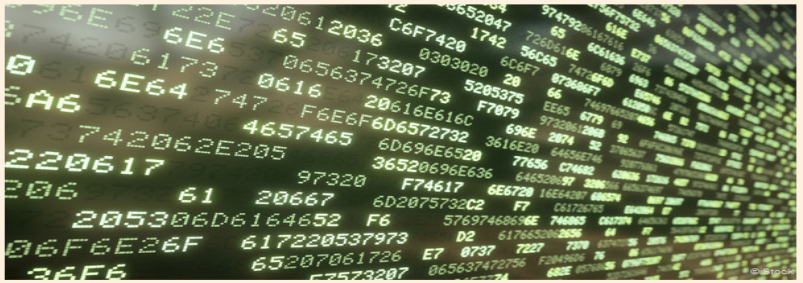
Microservices: CI with LambdaCD - The underlying infrastructure (1/3)
By Stephan Kraus @donmumpleo
Abstract: In the last two months, we started our journey towards a new microservices architecture. Among other things, we found that our existing CD tools were not ready to scale with those new requ...
View on web

2 4

Banks claim blockchain breakthrough in money transfer

Ben McLannahan in New York

Share Author alerts Print Clip Comments



A group of seven banks including Santander, CIBC and UniCredit is claiming a breakthrough, ranking among the first financial institutions in the world to move real money across borders using blockchain-based technology.

IBM Bridges Blockchain, AI With New Business Unit

Michael del Castillo (@DelRayMan) | Published on August 29, 2016 at 22:30 BST

NEWS

529 4

IBM is reorganizing its internal blockchain team into a business unit that encompasses its artificial intelligence and cloud computing efforts.

Called 'Industry Platforms', the division will be led by Bridget van Kralingen, IBM's former senior vice president of Global Business Services. Further, as part of the launch, IBM's entire blockchain leadership will make the transition to the unit, first announced last September.

In addition to the work on blockchain tech, the business unit will lead IBM's efforts to bridge its financial services work with its Watson artificial intelligence initiative.

IBM CEO and chairman Ginni Rometty said of the move:

At the heart of decentralized systems such as Bitcoin is a revolutionary platform – the “blockchain”

Traditional banks are built on private, centralized systems:

Account owners

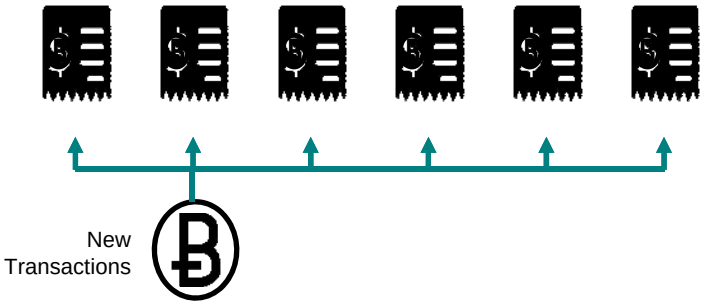
Bank balances

Transaction records

There is one central ledger for accounts, identities, and transactions.

In Bitcoin, the central functions are distributed to all the participants in the system:

Every user has access to their own copy of the transaction ledger in a long ledger called the **BLOCK CHAIN**



CRYPTOGRAPHY is used to verify transactions and keep information private

New currency is issued to users as a REWARD for doing the computation “work” involved in verifying transactions.

Introducing Hyperledger

A collaborative effort created to **advance blockchain** technology by identifying and addressing important features for a **cross-industry open standard** for distributed ledgers that can transform the way **business transactions** are conducted globally.

Hyperledger Project Members

Open source participation is important to the IBM Strategy

A Dramatic Commitment to Open Technology !

5 IBMers contributing to Linux & Apache Projects

1999

2016

62k+ IBMers contributing to 150+ Open Orgs

Founder & Platinum Sponsor

IBM is a top-3 contributor in the past 4 releases

IBM has 450+ developers & 20 core reviewers

Founder & Platinum Sponsor

IBM has the largest Cloud Foundry deployment

IBM has 10 dedicated core contributors

IBM a founding member of the newly established foundation

2.1b npm downloads in the past month

Leading dev of nightly builds for x86, Power & Mainframe systems

#2 in Commits in 2015 behind Docker

Donation of IBM SystemML machine

IBM pledged 3,500 devs to Spark across 12 labs

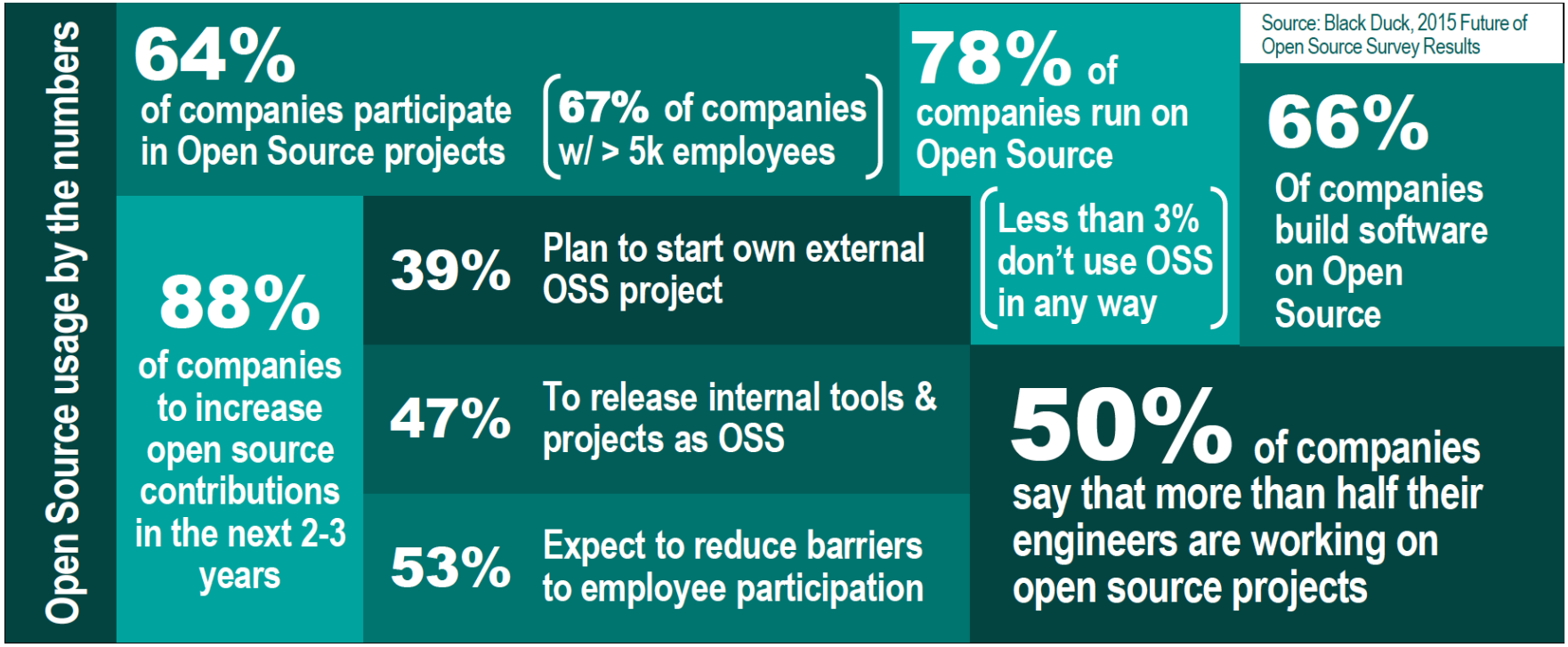
Education for 1m + data scientists & engineers on Spark.

IBM is a founding member

500+ organizations participating

Provides customer-led guidance to cloud standards-defining orgs

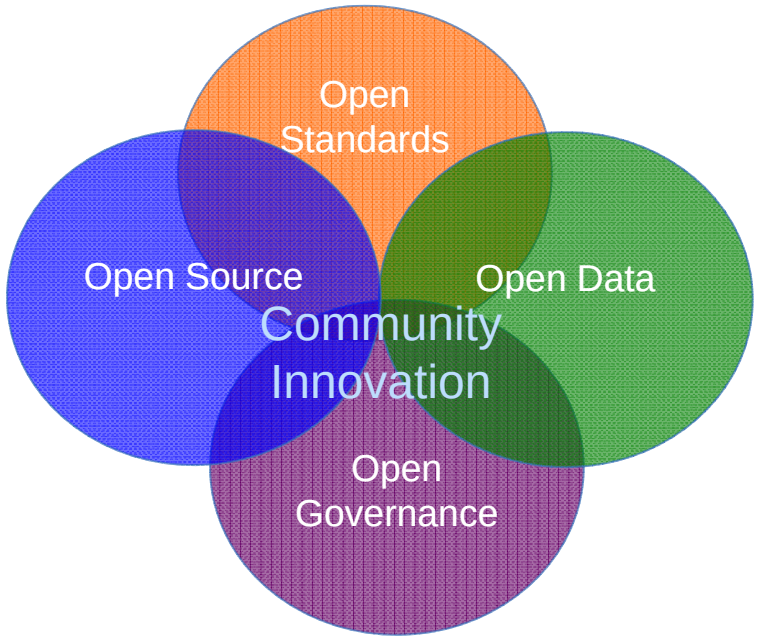
Open Source in the Enterprise



https://www.blackducksoftware.com/future-of-open-source

Open Explained

- OPEN is **enticing:**
  - Done **correctly**: reduce cost base prevent vendor lock in, leverage large community
  - Done **incorrectly**: expensive / difficult to maintain critical systems, impossible to control, legal liabilities
- OPEN is often **miss-understood** – e.g. open source software confused with open standards; open source = free!
- IBM **leader** in Open since late 1990s – Linux, Eclipse, Open Cloud. Open Source embedded in our Software.
- **Intelligent balance (Open – Commercial)** essential, system engineering led, based on total cost of ownership underpinned by Open Standards



Join the movement

As with Java, Linux, Open Stack, Node and Spark, industry can advance Hyperledger (open blockchain) technology and focus it on the requirements of industrial use cases by working together through an open source foundation

What?

- Enterprise grade specification
  - Functional & non-functional
- Help build open source fabric
- Licensing (Apache / OSS)

How?

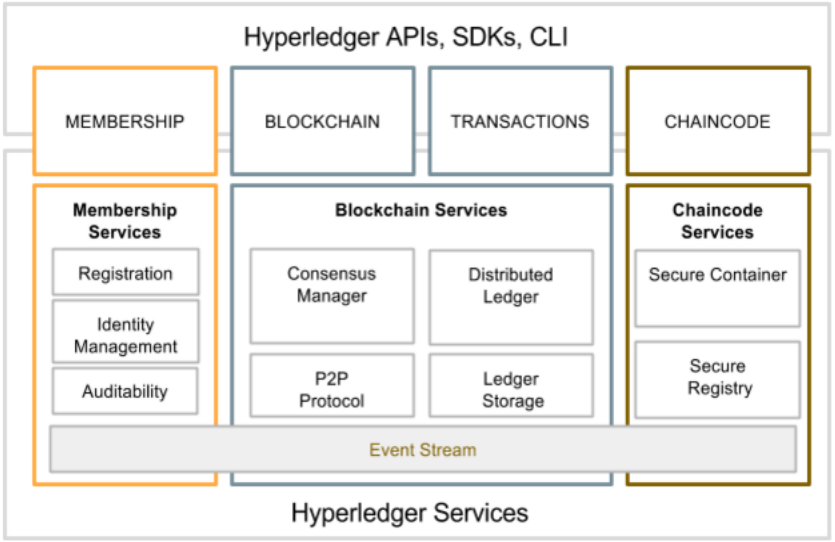
- Community led
- Open Governance
- Promote use and support
- Advisory board

Hyperledger Fabric

Hyperledger Interface is REST APIs

Managing identity, privacy, confidentiality and auditability

PKI-based infrastructure to enable a permissioned Blockchain

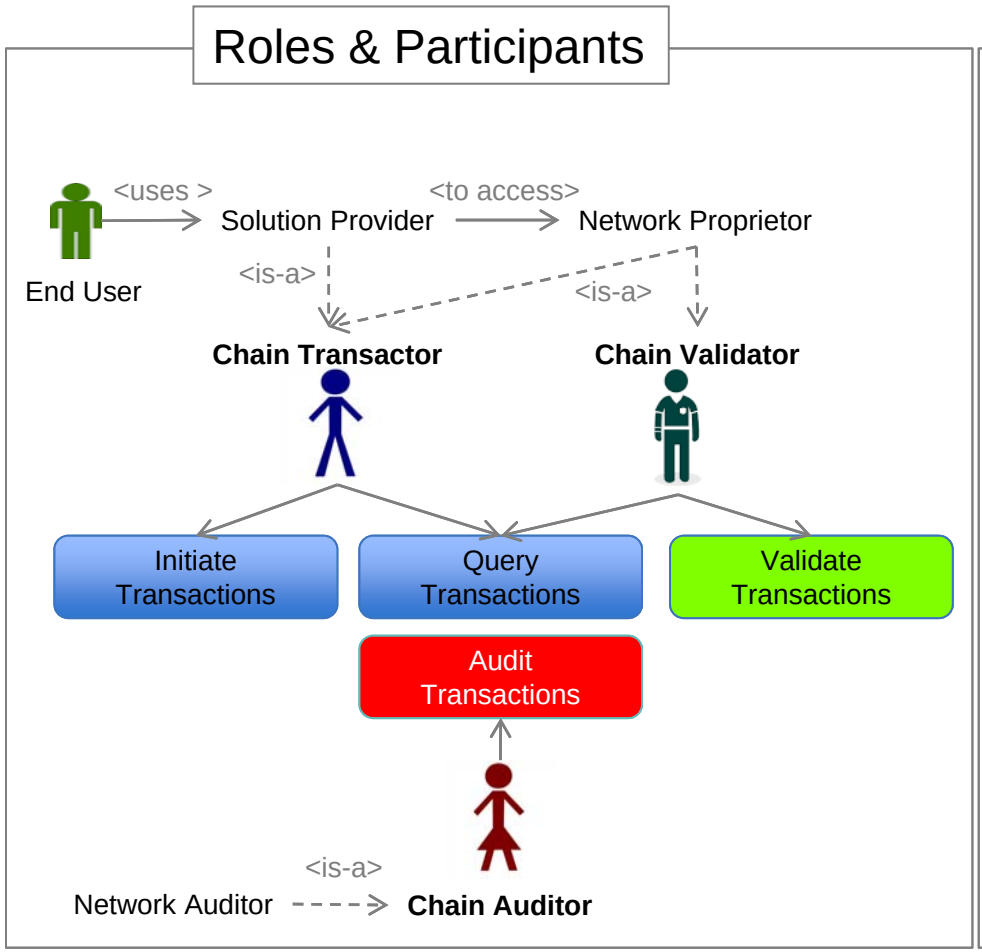


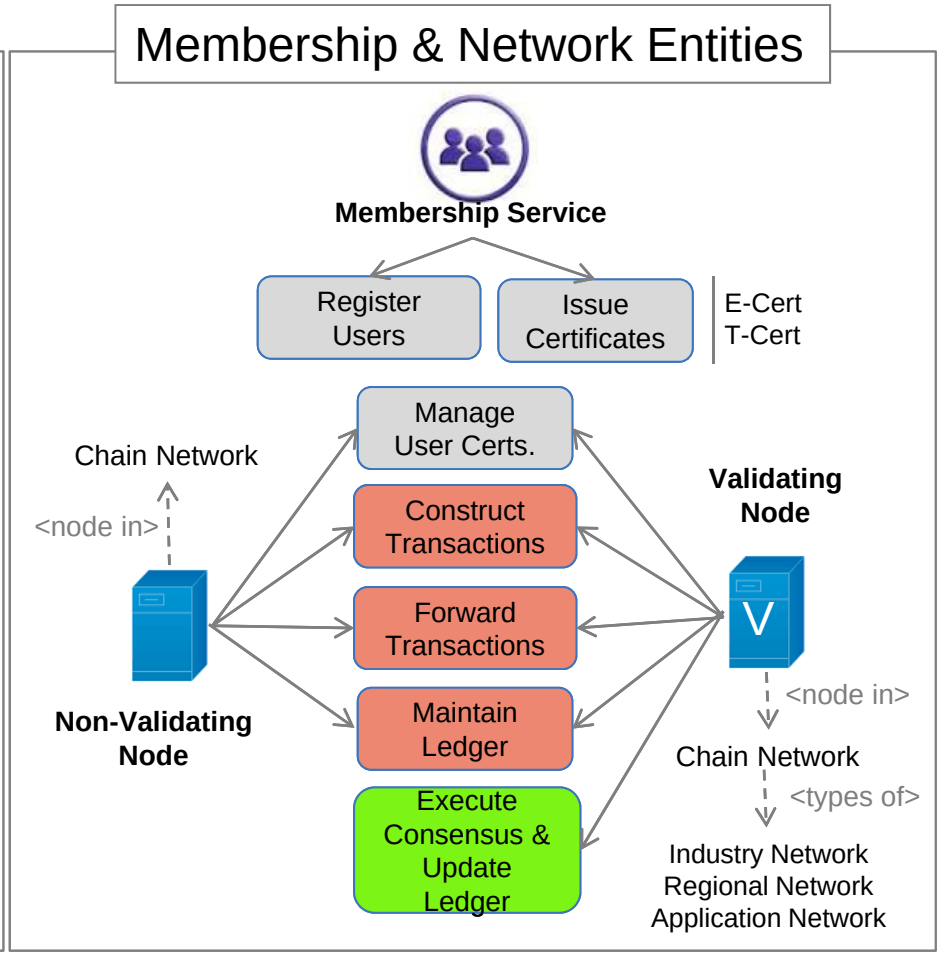
Secured and lightweight way to sandbox the "Smart Contract" execution on the validating nodes.

SDK with support for Go, Java and Node.js

Manage the distributed ledger through a peer-to-peer protocol, built on HTTP/2.
Pluggable consensus algorithm.
Default consensus based on PBFT/Sieve

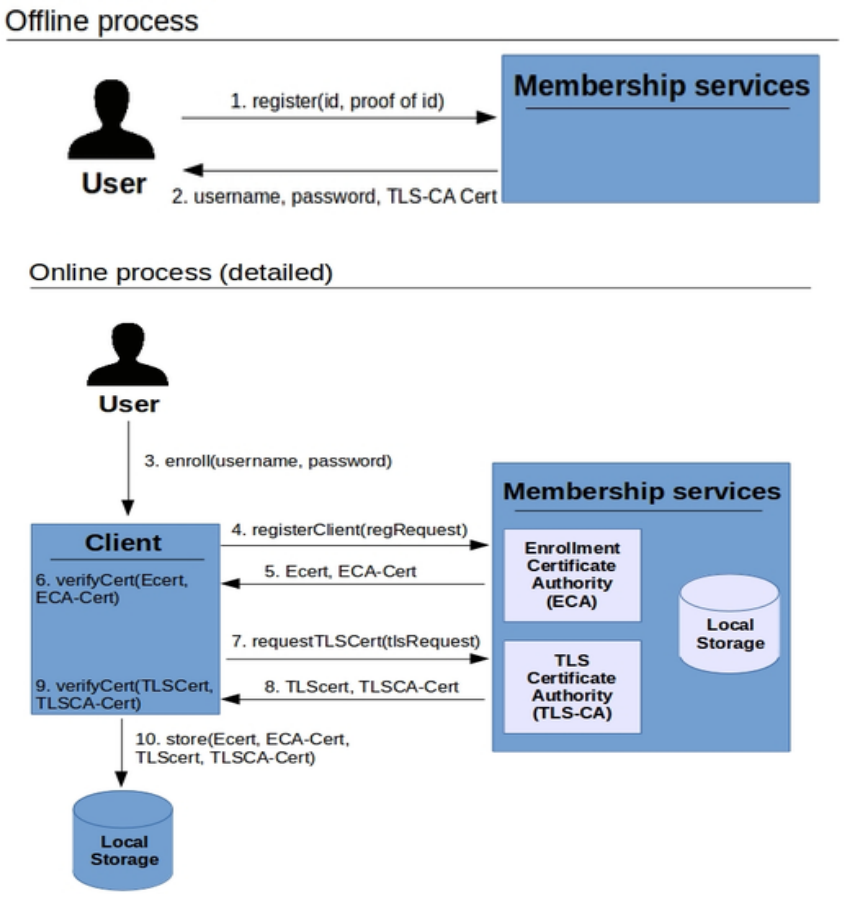
Hyperledger Overview: System Context

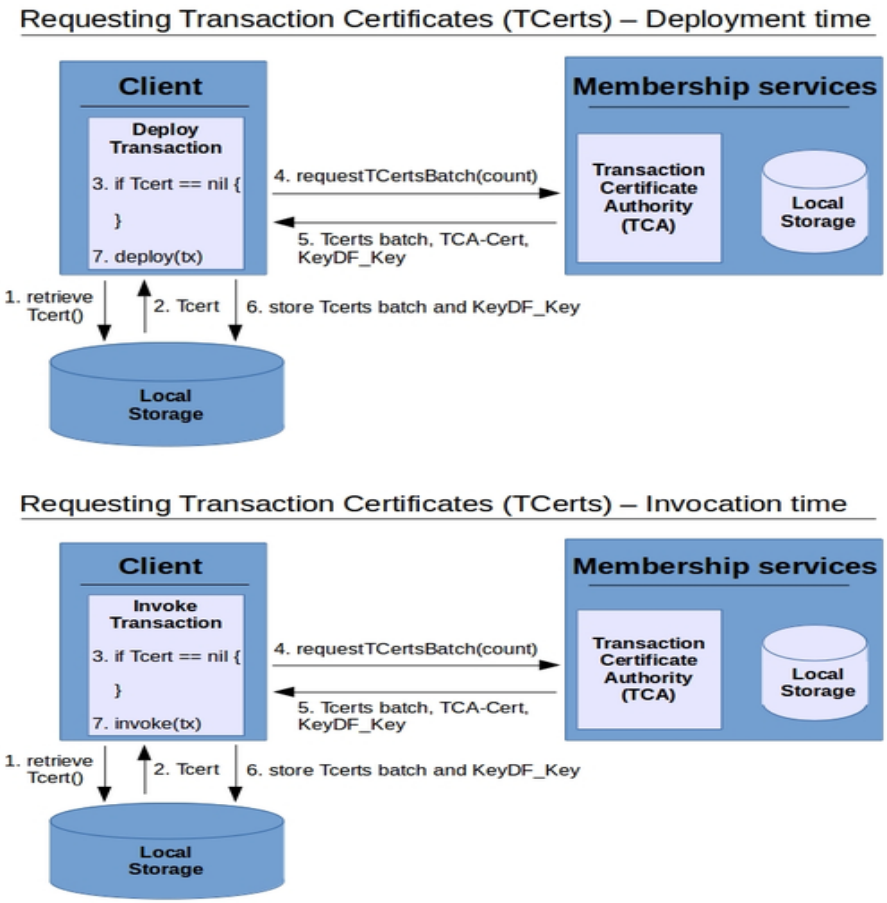




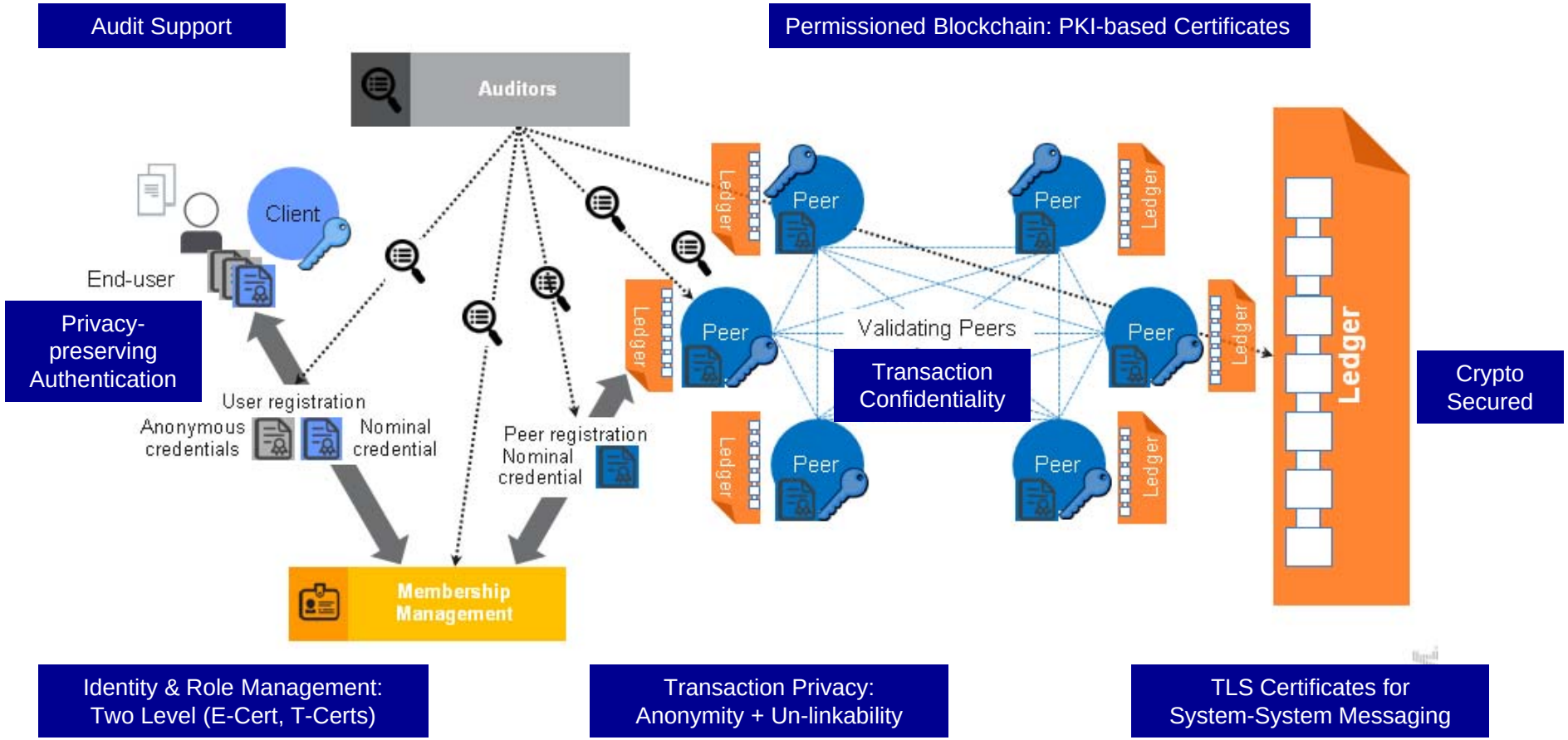
Hyperledger Membership Services

Note: Deployment Transaction: Transactions that deploy chaincode to a chain
Invocation Transaction: Transactions that invoke a function on chaincode





Hyperledger Security Overview



Hyperledger REST APIs

- Block
  - GET /chain/blocks/{block-id}
- Blockchain
  - GET /chain
- Chaincode
  - POST /chaincode
- Network
  - GET /network/peers
- Registrar
  - POST /registrar
  - GET /registrar/{enrollmentID}
  - DELETE /registrar/{enrollmentID}
  - GET /registrar/{enrollmentID}/ecert
- Transactions
  - GET /transactions/{UUID}

Example:

Blockchain Retrieval Request:

```
GET host:port/chain
```

```
message BlockchainInfo {
    uint64 height = 1;
    bytes currentBlockHash = 2;
    bytes previousBlockHash = 3;
}
```

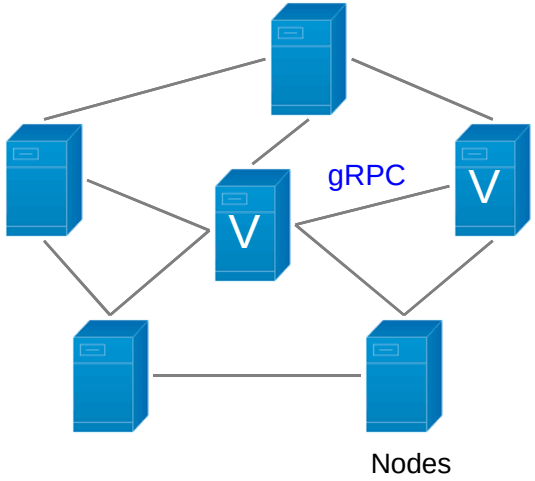
Blockchain Retrieval Response:

Number of Blocks in the Blockchain

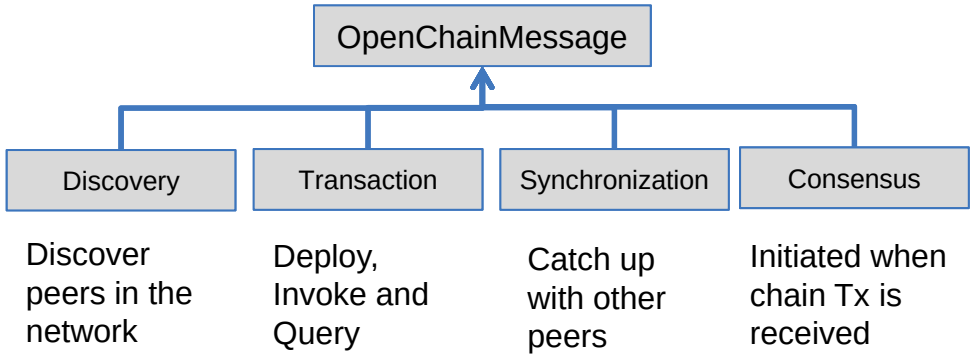

```
{
   "height": 174,
   "currentBlockHash": "lIfbDax2NZMU3rG3cDR11OGicPLp1yebIkia33Zte9AnfqvffK6tsHRyKwsw0hZFZkCGIa9wHVkOGyFTcFxM5w==
   "previousBlockHash": "Vlz6Dv5OSy0OZpJvijrU1cmY2cNS5Ar3xX5DxAi/seaHHRPdssrljDeppDLzGx6ZVyayt8Ru6jO+E68IwMrXLQ=
}
```

Hyperledger Protocol

Open Blockchain peer-to-peer communication is built on gRPC that allows bi-directional stream messaging



Message passed between nodes encapsulated by OpenChainMessage proto structure.



OBC data structures, messages, and services are described using proto3.

Protocol Buffers serialize data structures for data transfer between peers.

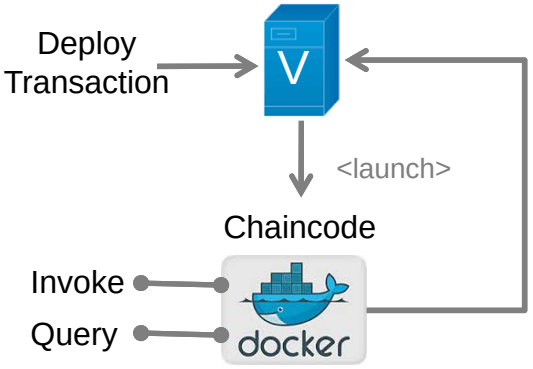
Message payloads are opaque byte arrays containing either the Transaction object or Response.

Transaction is always associated with a chaincode spec that defines the chaincode and the execution environment.

Hyperledger Chaincode implements Smart Contract

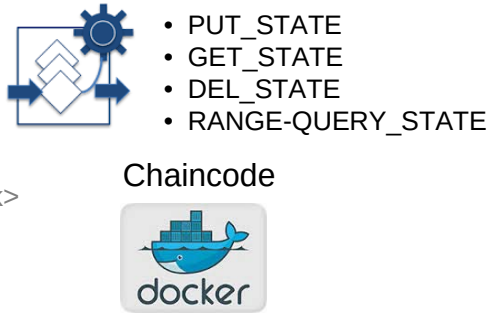
Chaincode is application code deployed as a transaction to be distributed in the network, managed by validating nodes, and implemented as Docker containers. Chaincode implemented in Go language.

Deploying Chaincode



- Register with Validating Node using ChainCodeID
- Call Invoke on Chaincode Interface to initialize

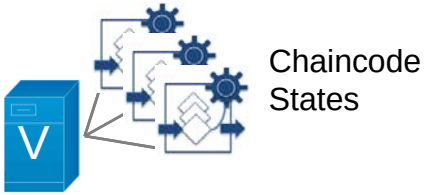
Chaincode State



- PUT_STATE
- GET_STATE
- DEL_STATE
- RANGE-QUERY_STATE

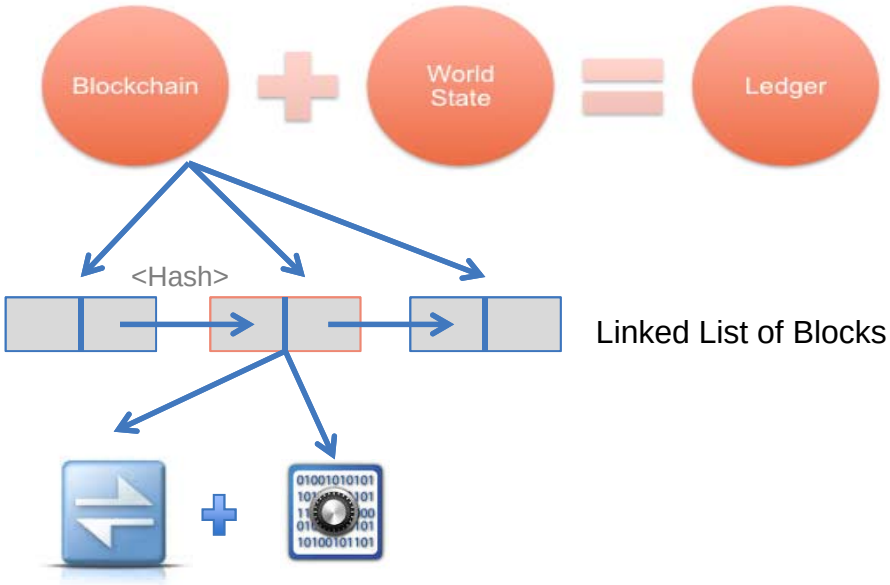
- Each Chaincode can define its own persistent state variables (key-value)
- Chaincode can update the state based on Invoke Tx

World State



- World state refers to collection of states of all deployed chaincode
- Organized as a bucket-tree to enable efficient crypto-hash

Hyperledger Ledger



Linked List of Blocks

Message Block{

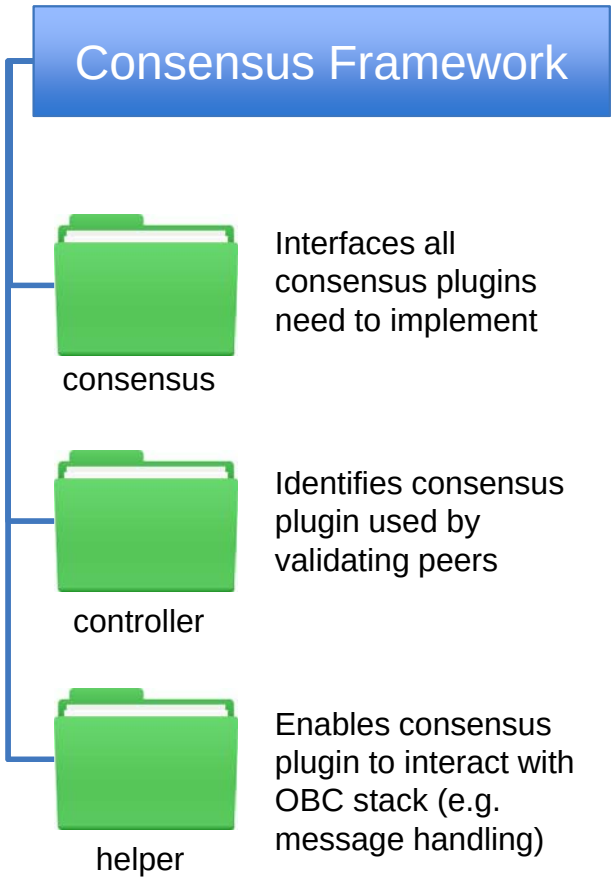
- `version` - Version used to track any protocol changes.
- `timestamp` - The timestamp to be filled in by the block proposer.
- `transactionsHash` - The merkle root hash of the block's transactions.
- `stateHash` - The merkle root hash of the world state.
- `previousBlockHash` - The hash of the previous block.
- `consensusMetadata` - Optional metadata that the consensus may include in a block.
- `nonHashData` - A `NonHashData` message that is set to nil before computing the hash of the block, but stored as part of the block in the database. }

Message BlockTransactions{

- `BlockTransactions.transactions` - An array of Transaction messages. Transactions are not included in the block directly due to their size. }

Hash of the Block based on FIPS 202

Hyperledger Pluggable Consensus Framework



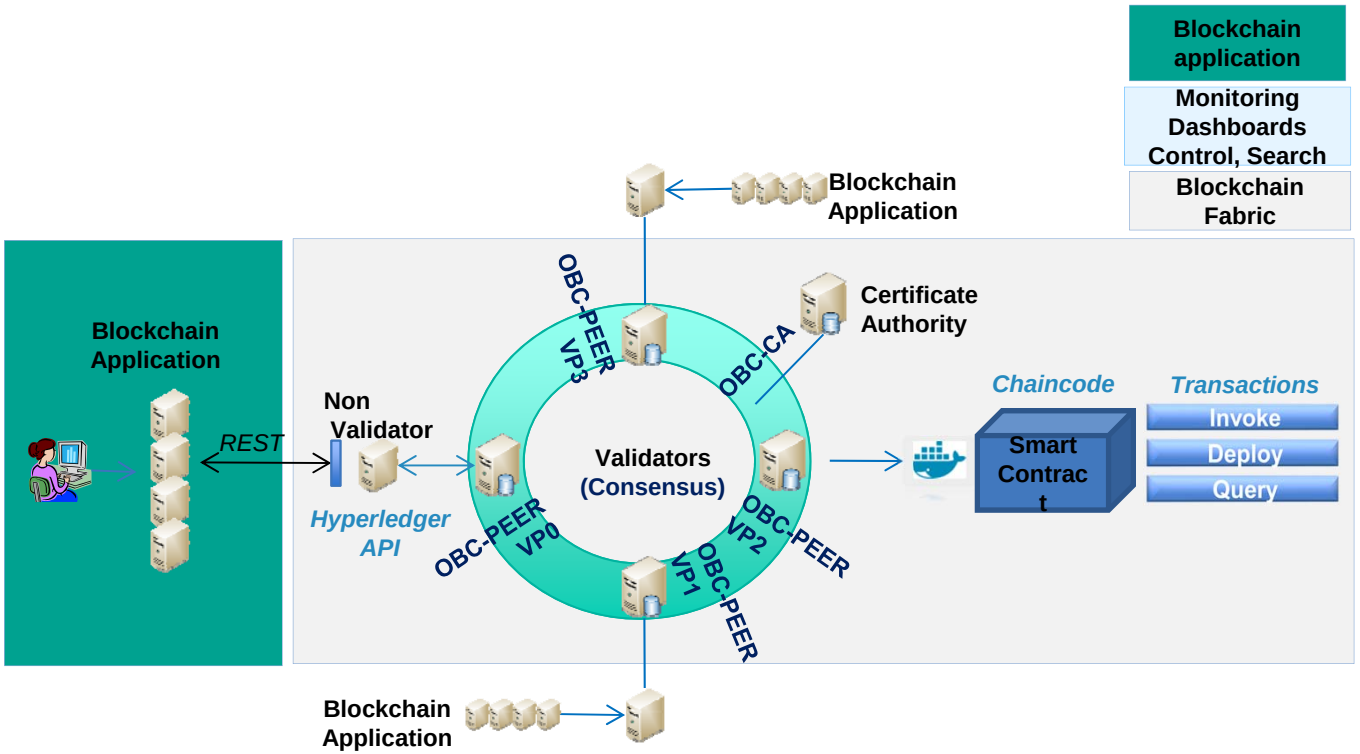
Practical Byzantine Fault Tolerance (PBFT)

If there are f failures then need 3f+1 replicas in an asynchronous network to ensure data integrity

SIEVE

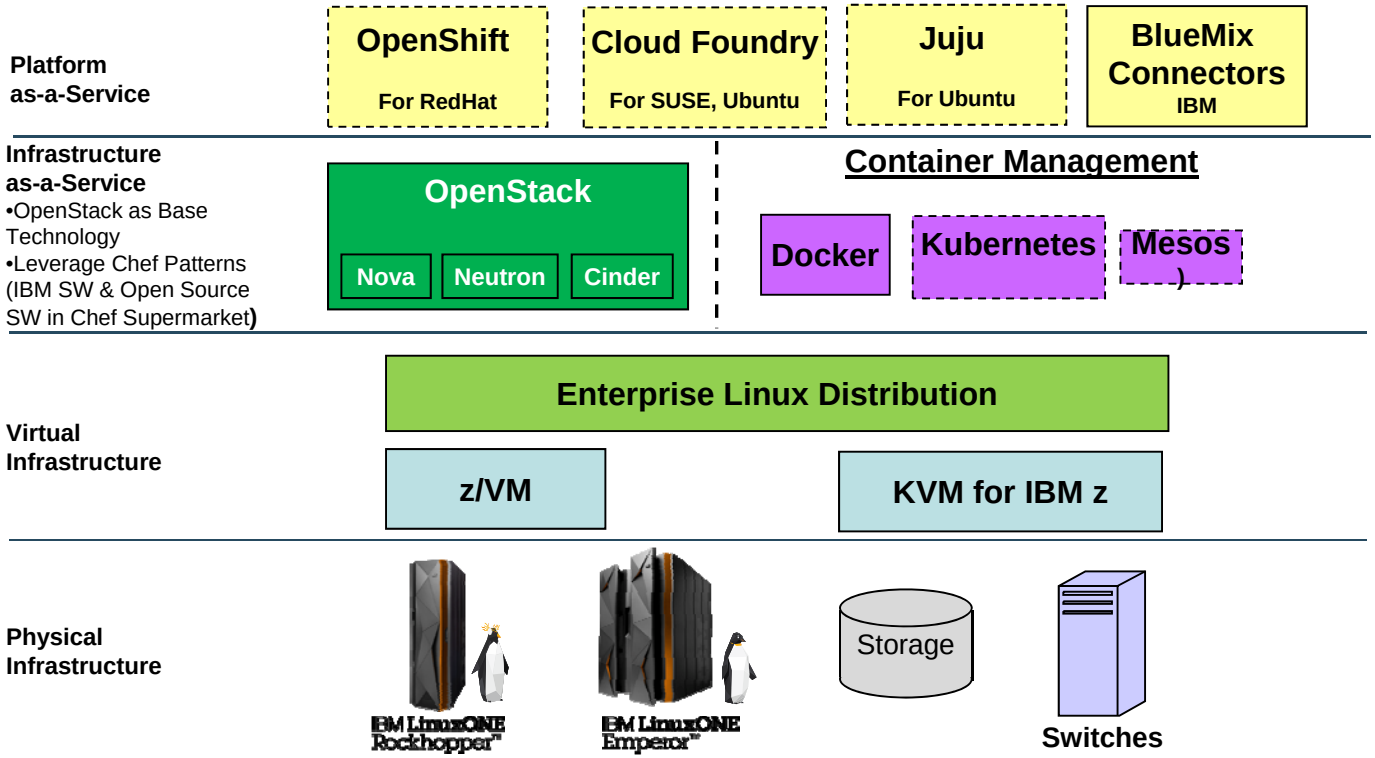
Extends PBFT to handle non-deterministic transactions by leveraging Execute-Verify (EVE) replication mechanism.

Hyperledger Fabric – Sample Application Architecture



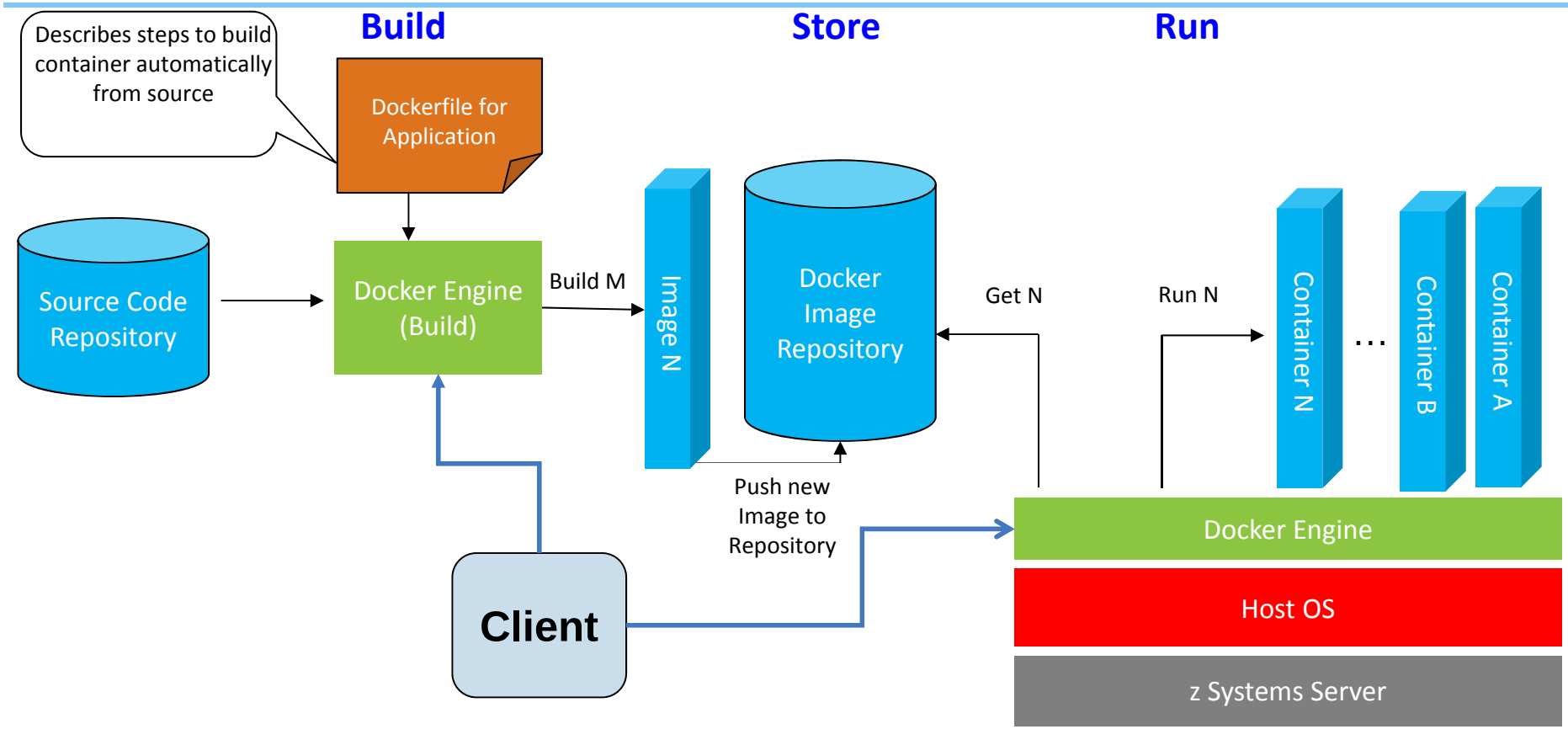
Smart Contracts & Containers

LinuxONE Blockchain : Leveraging Open Source

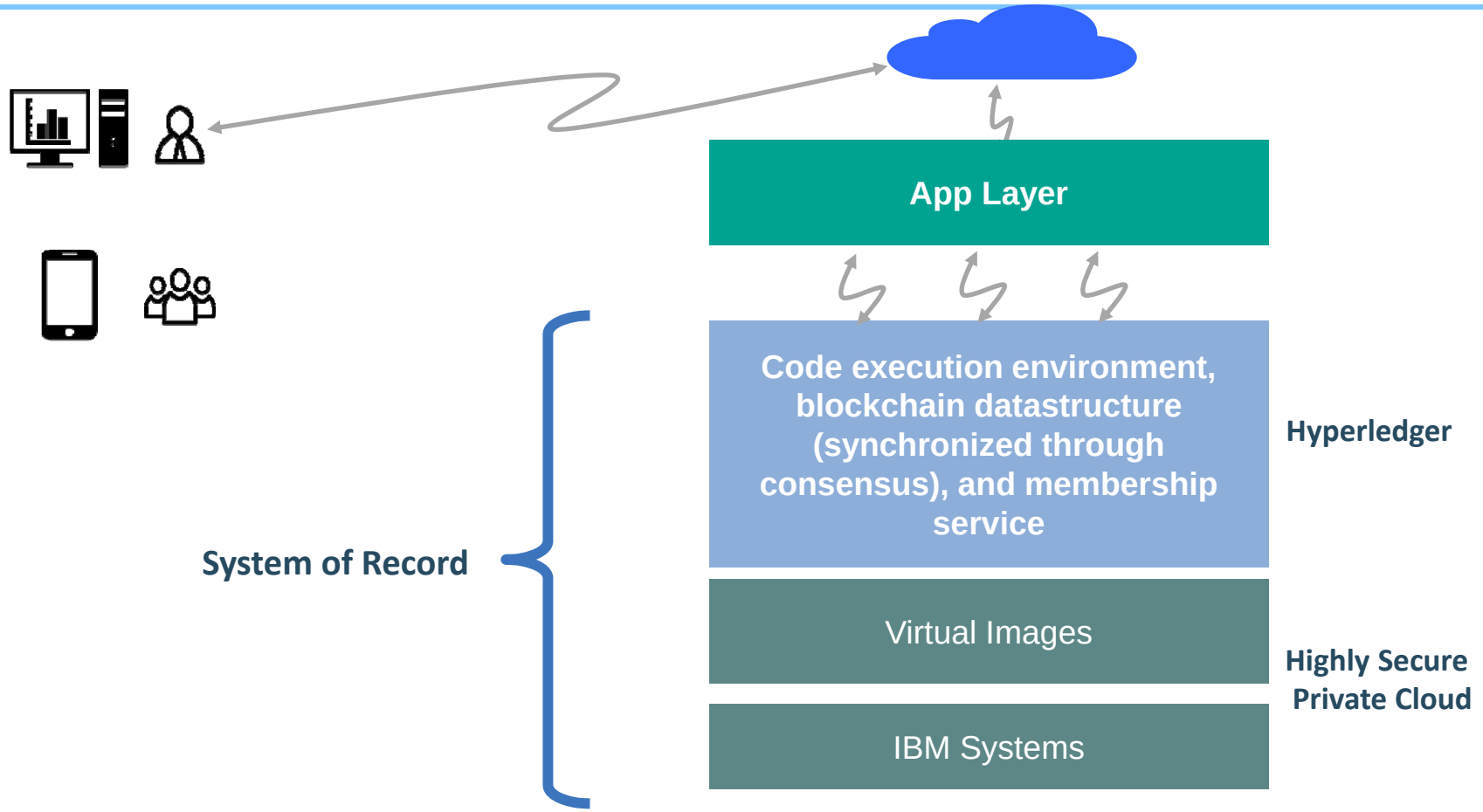


Each Distro (SUSE, RedHat, Ubuntu) will have its own flavor of a cloud stack

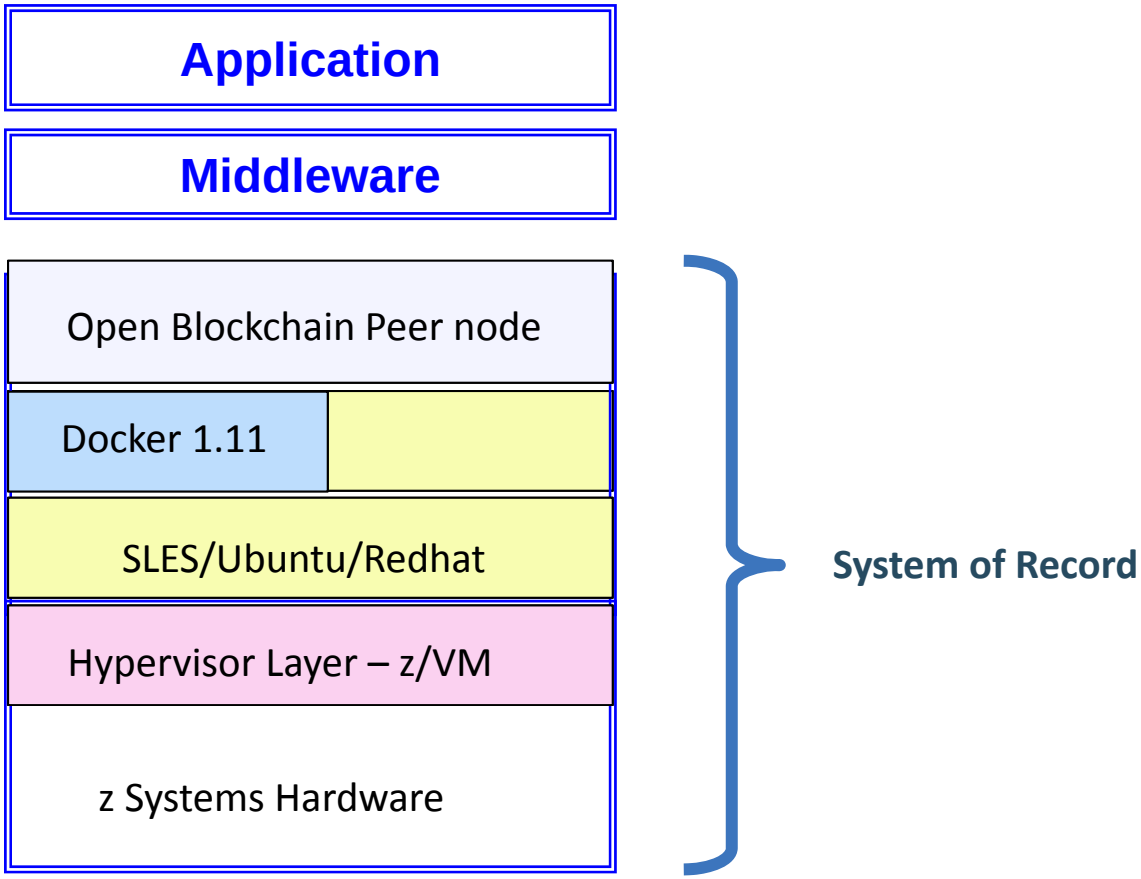
What are Docker basic functions?



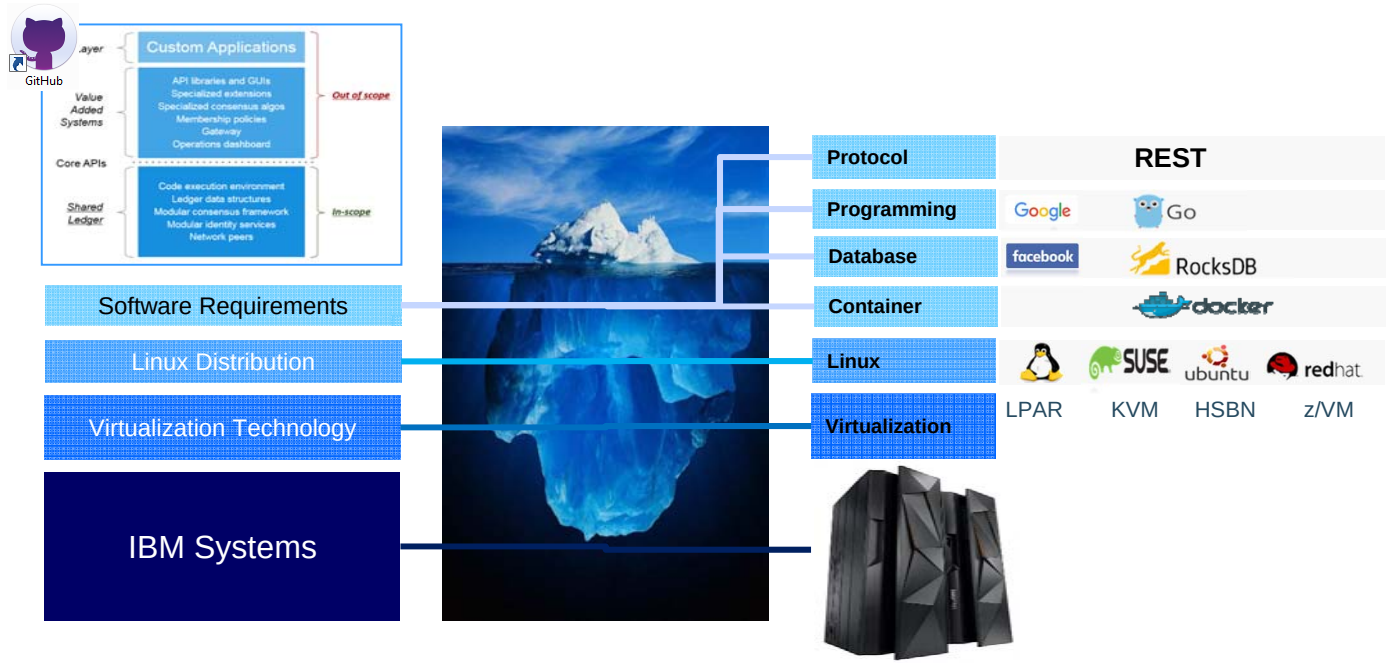
High Level View of the Infrastructure



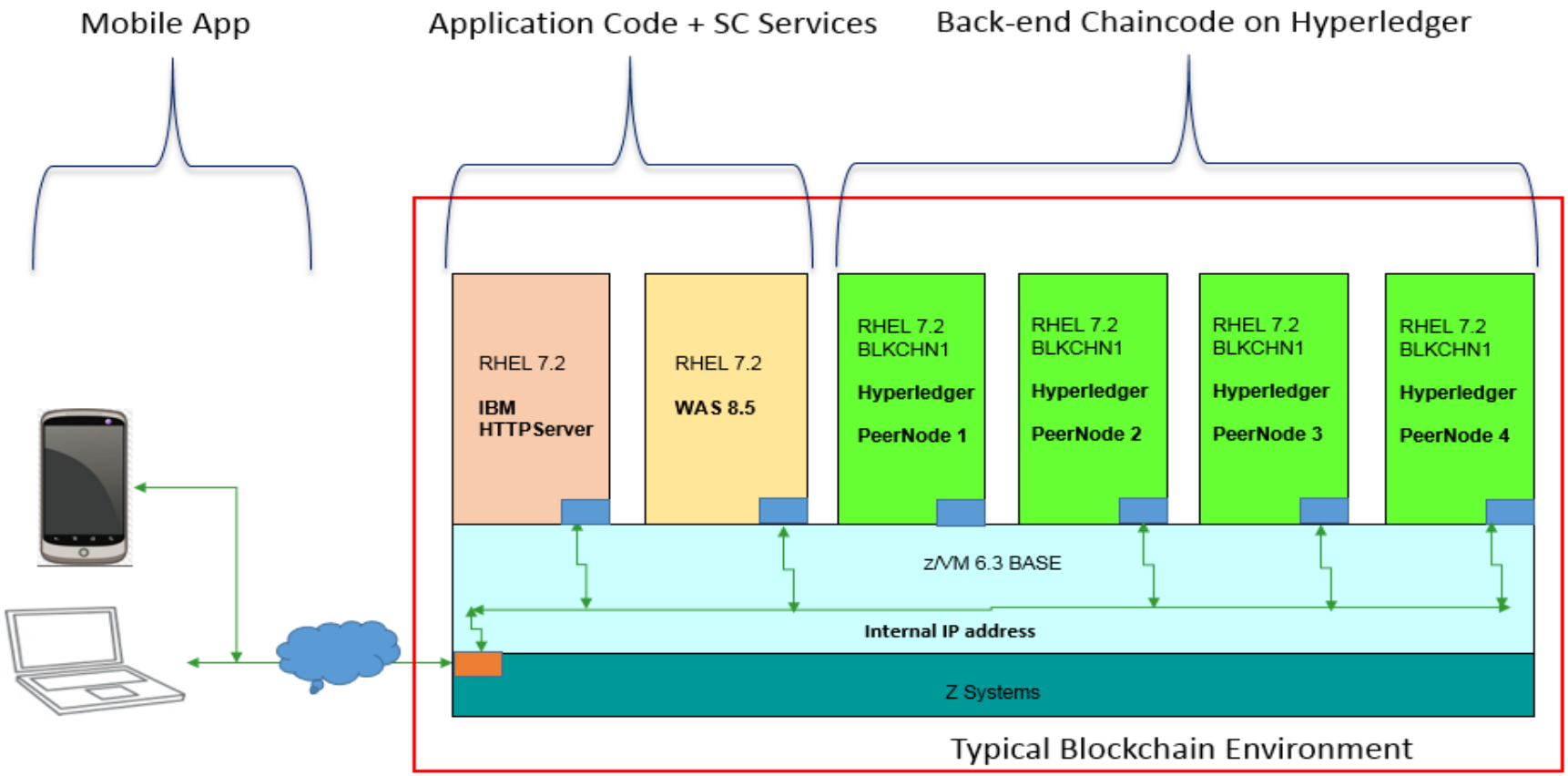
High Level Solution Components



Hyperledger Solutions Architecture…



High level Infrastructure View



Use of Containers in Blockchain

- **Chain codes run in Docker containers**
  - **Isolated from one another**
  - **Isolated from the *peer* control code**
- **Entire peer instance**
  - **Can run in a Docker container**
  - **Run in a VM**
- **Packaging, signing, install, execute as an appliance**

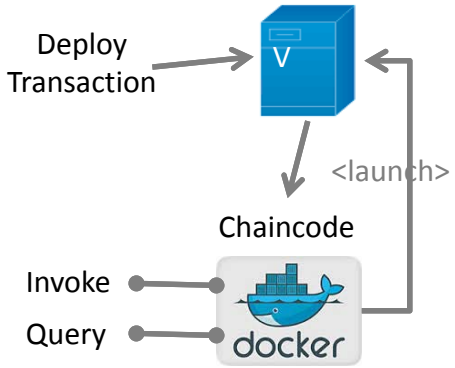
Hyperledger Chaincode container

- ✓ We can build blockchain chain code as Docker images that hold your business logic and automation code.
- ✓ Docker containers can be created from those Docker images to run your chain codes.
- ✓ Consortium can share those chain code docker images via private registry

Hyperledger Chaincode implements Smart Contract

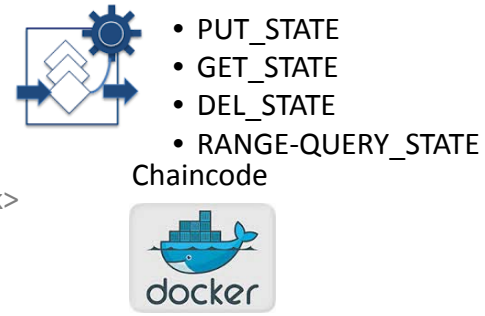
Chaincode is application code deployed as a transaction to be distributed in the network, managed by validating nodes, and implemented as Docker containers. Chaincode implemented in Go language.

Deploying Chaincode



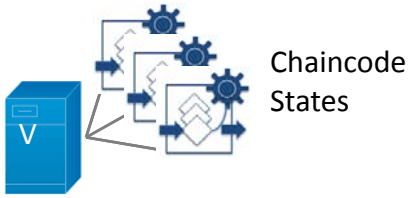
- Register with Validating Node using ChainCodeID
- Call Invoke on Chaincode Interface to initialize

Chaincode State



- Each Chaincode can define its own persistent state variables (key-value)
- Chaincode can update the state based on Invoke Tx

World State



- World state refers to collection of states of all deployed chaincode
- Organized as a bucket-tree to enable efficient crypto-hash

Hyperledger Chaincode implements Smart Contract – SWIFT MT700

Deploying Chaincode/Smart contract :

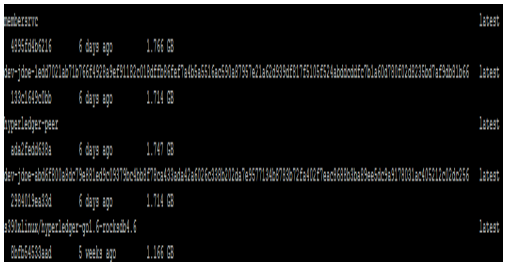
Example : Create SWIFT MT700 table Chaincode developed in go lang and packaged as a Docker image.

```
37
38      // Create SWIFTMT700 table.
39      // Column names are MT700 field names
40      // In addition to MT700 fields the ContractID and STATE of the L/C are stored.
41      // Column names MT700Fieldx to be replaced with actual MT700 field names.
42      err = stub.CreateTable("SWIFTMT700", []*shim.ColumnDefinition{
43              &shim.ColumnDefinition{"ContractID", shim.ColumnDefinition_STRING, true},
44              &shim.ColumnDefinition{"MT700Field1", shim.ColumnDefinition_STRING, false},
45              &shim.ColumnDefinition{"MT700Field2", shim.ColumnDefinition_STRING, false},
46              &shim.ColumnDefinition{"MT700Field3", shim.ColumnDefinition_STRING, false},
47              &shim.ColumnDefinition{"STATE", shim.ColumnDefinition_STRING, false},
48      })
49      if err != nil {
50              return nil, errors.New("Failed creating SWIFTMT700 table.")
51      }
52
53      return nil, nil
54 }
```

Build

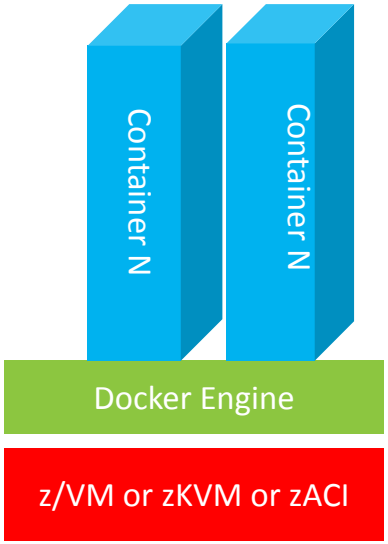
Chaincode Images Stored on Master Peer:

Example : Create SWIFT MT700 table Chaincode developed in go lang and packaged as a Docker image.



Store

Application Invokes Event



Why Blockchain Dockerization on z Systems

Docker is available to use on z Systems platforms

- Same code and open source model as used in the industry today
- Exact Same Usability and Experience as on other platforms for developers
- Growing ecosystem of dockerized applications for z Systems and increasing community engagement

Docker is Better on z Systems

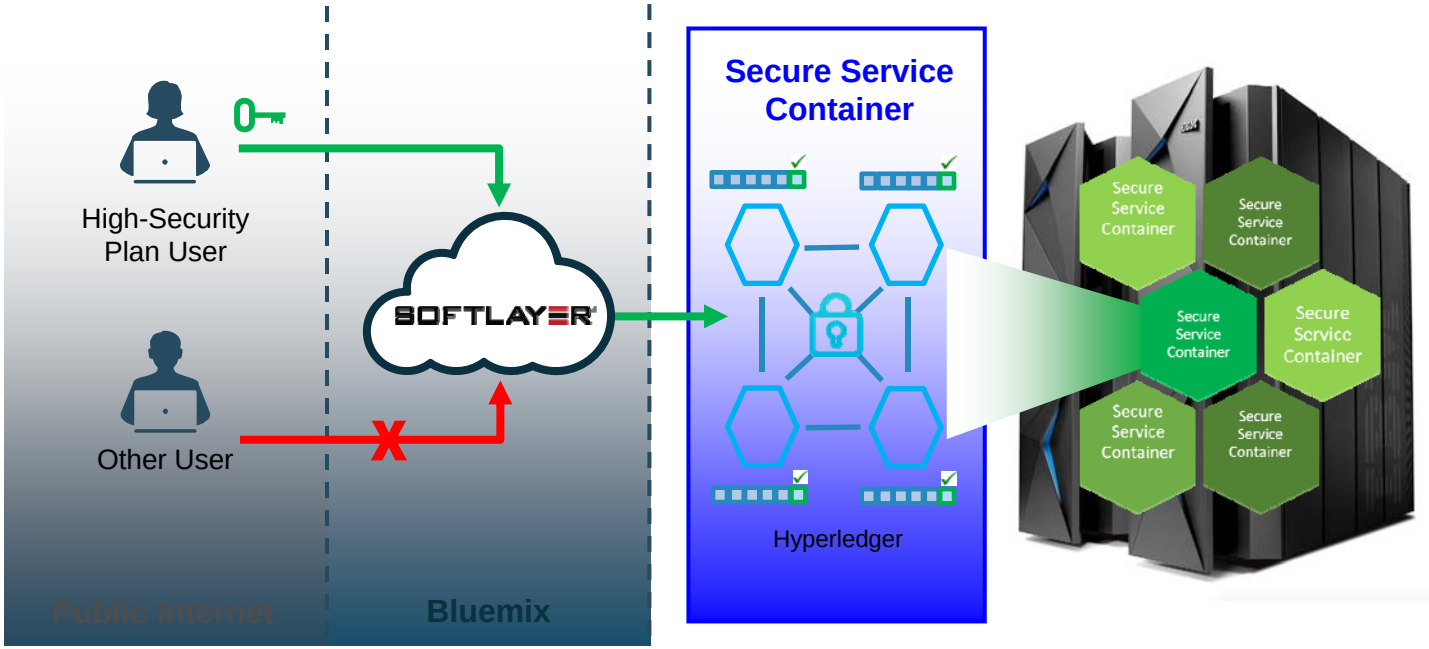
- Greater System Capacity to run Typical Cloud Native Docker Workloads
- More containers per system lowers cost of operations for service providers
- With its Huge IO Bandwidth, z Systems excels at data oriented workloads running in containers

z Systems Platform are built for workload consolidation – Docker enables it

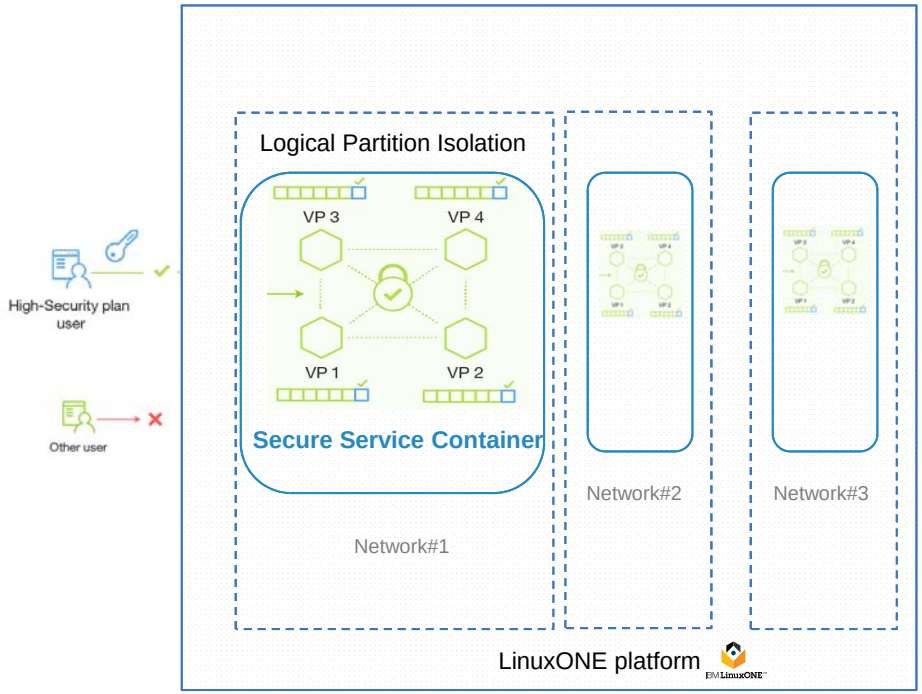
High Security Business Network (HSBN) On LinuxONE

HSBN Architecture – Overview

High Security Business Network



High Security Business Network Architecture – High Level



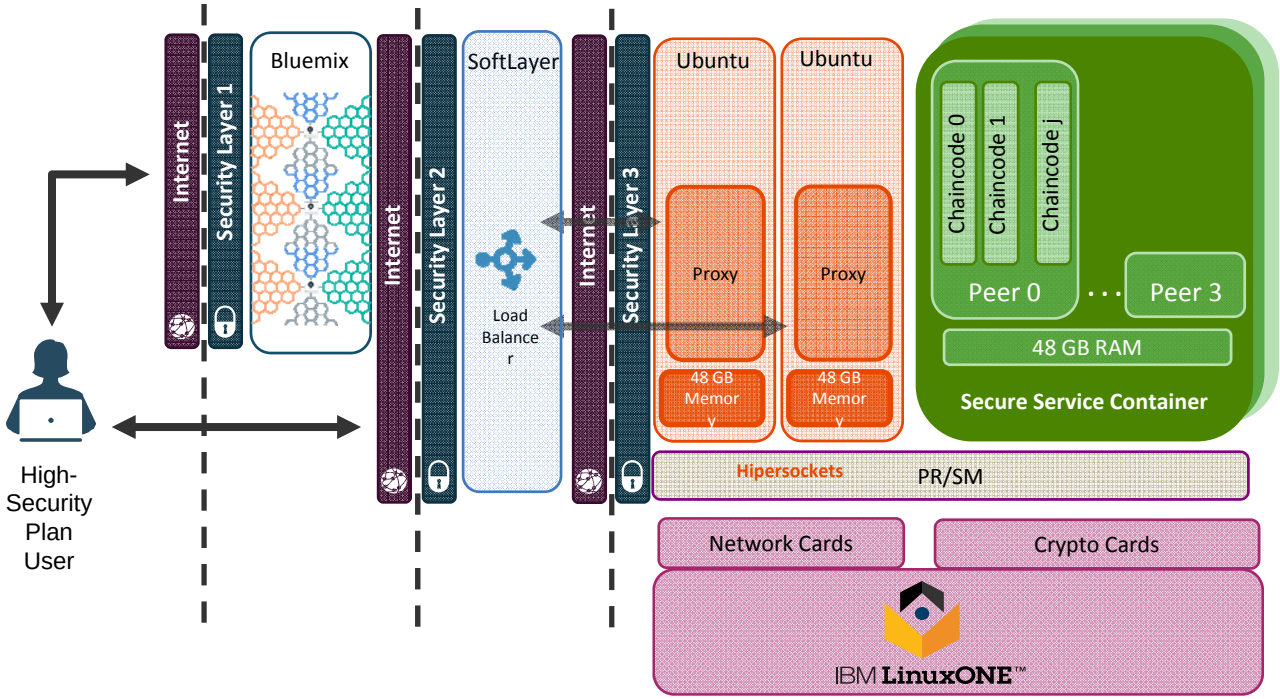
High Security Business Network

The high security business network is deployed as an appliance into a Secure Service Container, which provides the base infrastructure for hosting blockchain services. The appliance combines operating systems, Docker, middleware, and software components that work autonomously to provide core services and infrastructure with optimized security.

Overview: https://console.ng.bluemix.net/docs/services/blockchain/etn_ssc.html

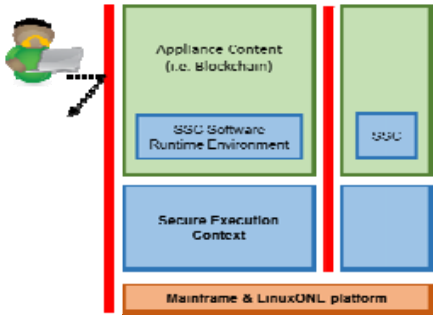
HSBN on LinuxONE : Reference Architecture

High Security Business Network



High Secure Blockchain Container Network

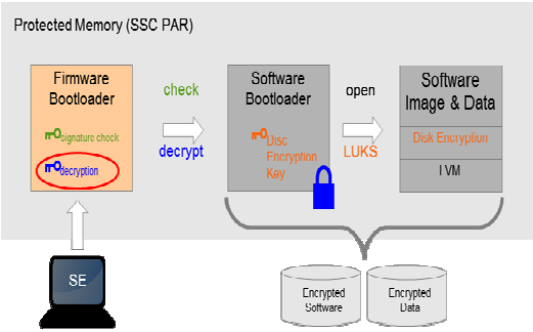
Secure Service Container ensures…



No system admin access, ever

- Once the appliance image is built, OS access (ssh) is not possible
- Only Remote APIs available
- Memory access disabled
- Encrypted disk
- Debug data (dumps) encrypted

How the Secure Service Container boot sequence works…



Boot sequence

1. Firmware bootloader is loaded in memory
2. Firmware loads the software bootloader from disk
   i. Check integrity of software bootloader
   ii. Decrypt software bootloader
3. Software bootloader activate encrypted disks
   i. Key stored in software bootloader (encrypted)
   ii. Encryption/decryption done on the flight when accessing appliance code and data

High Security Business Network (HSBN) – Hyperledger Container

Security	Performance	Compliance	Simplicity
• **Protection against misuse of privileged user credentials:** Blockchain operating environments and data are protected by secured service containers against access and abuse by root users, system administrator credentials and other privileged user access. These Blockchain instances are locked so they must deployed to system models configured to our high security settings. • **Malware protection:** Blockchain data and software is protected from malware being installed. • **Protection of peers from one another:** Blockchain peers are able to run in protected, isolated environments to prevent deliberate or unintentional leakage of information from one party's environment to another. • **Key safety:** Identity, communications, and data privacy are safeguarded by having all keys in a secure services container. For our general-availability release, enrollment key security will be further enhanced by implementing “secure key” using our tamper-resistant crypto-card.	• **Hardware accelerators:** Crypto optimization supports an environment that moves hashing and symmetric encryption to accelerators and optimizes digital signatures to reduce drain on CPU performance.	• **Highly auditable operating environment:** Hardware and firmware audit logs provide information about any critical actions done to system such as replacing hardware or changing configurations. This allows such changes to be audited, including verification of unauthorized actions.	• Open-source Hyperledger code along with a **single, integrated stack**.

Additional security and privacy benefits

Benefit	Value	How
Prevent Edward Snowden-type attack	**Protection against misuse of privileged user credentials**	*Our differentiation is that IBM Secure Service Containers prevent system admins with access to the hardware from disabling the restrictions as it is possible on other environments:* • *No access to the data store* • *No ability to modify any of the code in the container* • *All data leaving the container is encrypted* HOW: We do this by encrypting all data on the disk; only the machine hardware has the keys—there are no keys accessible to privileged users. Only authorized APIs are available (not the underlying software). For example on other systems, system admins can disable SELinux on the Redhat Enterprise Linux, then get full access to the system.
Data Privacy	**Participants in a business network can't see each other's private data**	*Because each peer in the network has a copy of all data from all parties, we do not want the owner of each peer to be able to look at the data stored in the peer. The container prevents the machine owner from peeking /viewing the raw data. The only thing a peer owner can do is start or stop a peer.* HOW: All peer data and code is encrypted all the time. The peer owner does not have the keys.

Thank You

Manoj S P

Open Source Solution's Specialist

manojs@sg.ibm.com