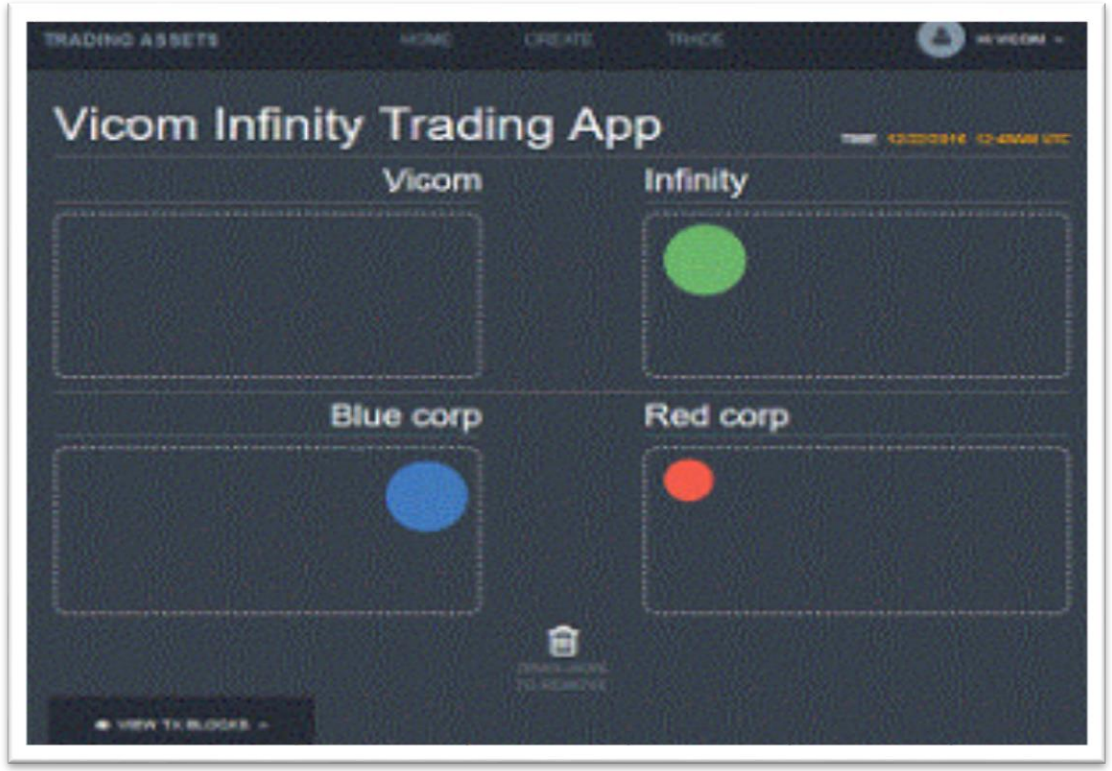
Lessons Learned from running Hyperledger Demos on z/VM Linux

Yongkook(Alex) Kim
Vicom Infinity
June 23rd 2017 VMWorkshop@ Ohio State University

Before we begin…let's try this first



http://blockchain.infinite-blue.com:3000

What is blockchain and why so much noise?

- You can deploy blockchain demo app yourself on IBM Bluemix in less than an hour without knowing anything about it!
- From zero knowledge on blockchain to running (pre-built) demo apps on local z Linux in 4 months – but almost impossible to learn about everything and updates that come out every day (Hyperledger is the fastest growing open source project on earth)
- There are three major blockchain/Distributed ledger technologies competing but still none of them are on production for enterprises
- There is a reason for public cloud providers servicing blockchain on their infrastructure → your data will be on somebody else's datacenter anyway!

What is Hyperledger?

- **Hyperledger** is an open source collaborative effort created to advance **cross-industry blockchain technologies**. It is a global collaboration, hosted by The Linux Foundation, including leaders in finance, banking, IoT, supply chain, manufacturing and technology.

(from Linux Foundation)

Traditional Contract – when things go wrong

- You are a landlord of a condo for a lease, your tenant A signed a paper contract
- On the last month of the lease, tenant A paid a check as a rent
- That check got bounced, and the property's window got broken
- Now you have to file a law-suit to get your money back

A Smart Contract – make your life/business easy

- You are a landlord of a condo for a lease, your tenant A signed a smart contract based on Hyperledger blockchain
- Money(deposit/rent) gets paid by internet payment
- On the last month of the lease, the property's IoT devices will check the condition of the condo
- Based on a business rule defined on the smart contract, deposit will be refunded to the tenant A

Enough with high level overview – what's under the hood?

Hyperledger deploys following open source technologies

- GoLang, Java and Node.js - to provide runtime for Chaincode(business logic)
- Docker/Vagrant – to encapsulate chaincode in a secure manner
- gRPC – enables peer-to-peer network
- RocksDB to CouchDB - Persistent state database using a key-value store interface
- PKI/TLS – for certificate authority and secure transmission/authentication

-- clear architecture description on Hyperledger with references:
https://www.zurich.ibm.com/dccl/papers/cachin_dccl.pdf

http://hyperledger-fabric.readthedocs.io/en/latest/

Birth of blockchain and revision by Hyperledger

- Very good YouTube video showing Bitcoin's blockchain concept/demo

→https://www.youtube.com/watch?v=_160oMzblY8

→https://anders.com/blockchain/blockchain.html

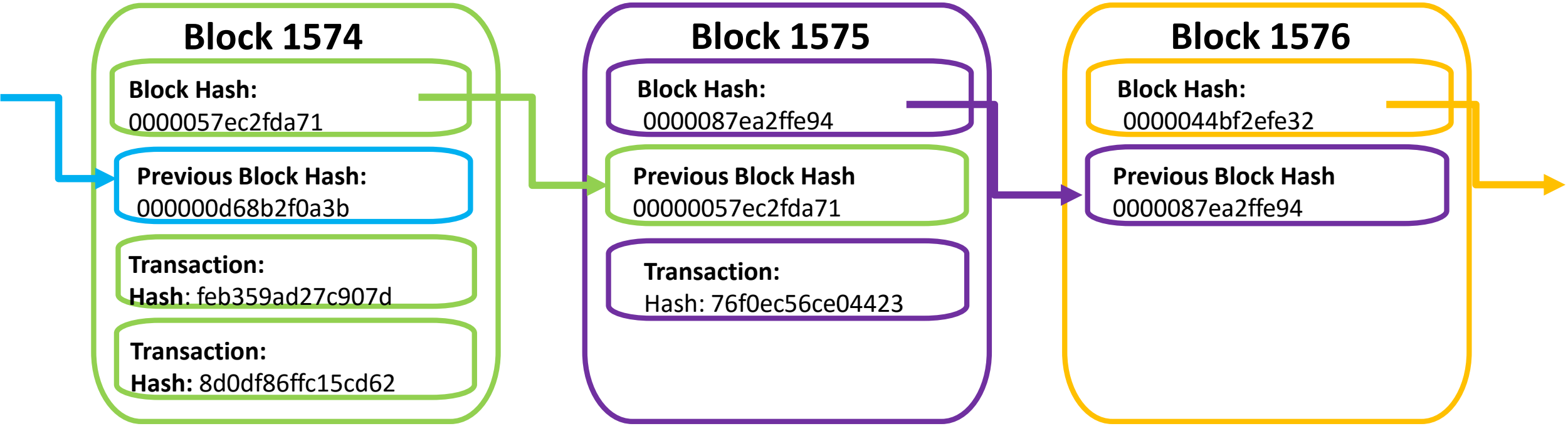
- It was powerful technology and all good but…..

Enterprises also wanted to have:

- Permissioned/private network
- Highly available and maximum security
- High throughput for business applications

- Therefore new and improved blockchain was introduced with accompanying validation methodologies and application framework – the birth of Hyperledger

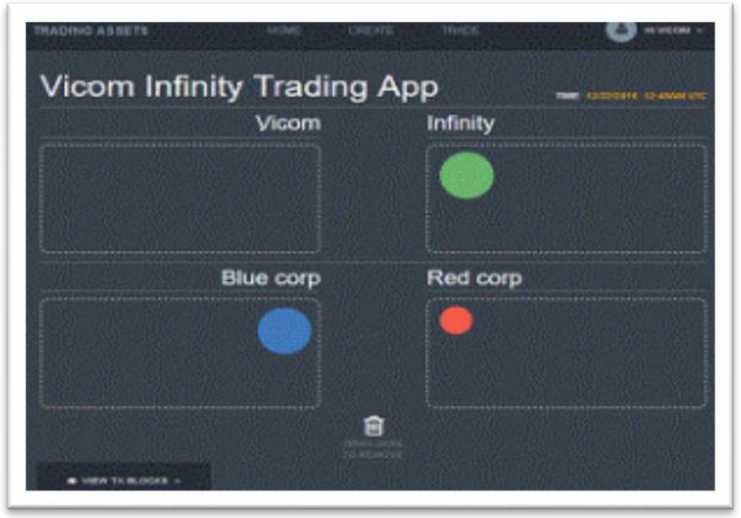
This is quick representation of Blockchain



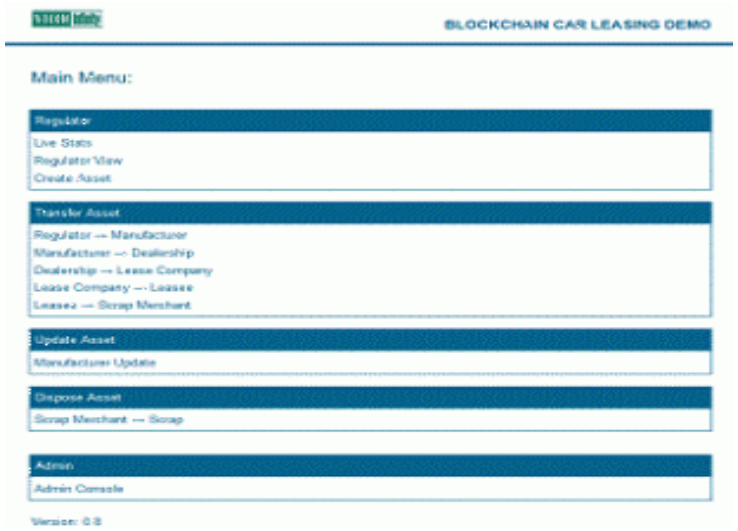
Made up of a series of blocks added in chronological order

Three demo apps I deployed on z Systems

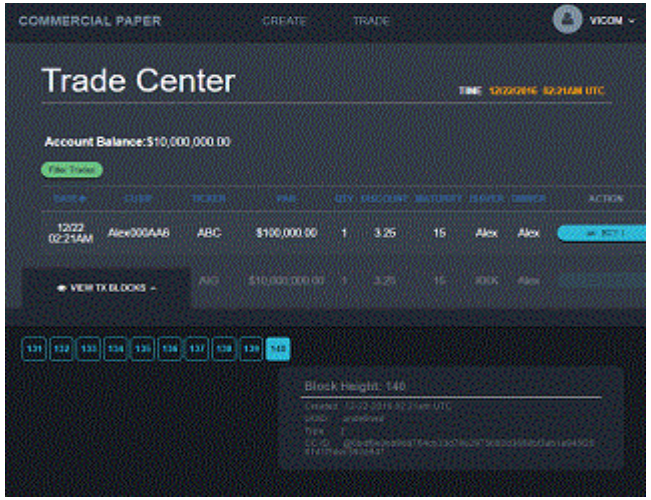
- **Marbles App** – create& trade marbles and watch how blockchains are created and added on each transaction.



- **Car Lease App** – a showcase for a car lease industry that how blockchain can provide security and efficiency using it's multi-party distributed ledger.



- **Commercial Paper App** – a simple demonstration of how a commercial paper trading network might be working with blockchain technology.



Visit http://blockchain.infinite-blue.com try all of these demo apps and get more information

Lesson #1 – Try Bluemix Service and demo first then use docker-compose to launch local fabric

- All of these apps made to run on IBM Bluemix, which contains many predefined Hyperledger fabric definitions
- It's a lot easier to try first with Bluemix as it gives you higher chance to build it successfully
- Once Bluemix model is built, you now have a golden model to compare with when you having issues while building it locally
- If you want to deploy both demo app and Hyperledger fabric on local, try port app first using Bluemix fabric
- Follow instruction on dockerhub to create/run fabric using docker-composer : https://hub.docker.com/r/ibmblockchain/

Lesson #2 – Be a good friend with Docker commands

- Launching Hyperledger fabric with docker-compose is quick and easy, but understanding detailed requirements of the fabric and connections can take time
- Start with single peer model and extend to two/four-peer if required
- Make sure docker images are fresh(not corrupted) and remove any cached images using ‘docker rm –f $(docker ps –a –q)’ when re-launching fabric
- Examine fabric console logs with ‘docker logs -f <containerID>’
- More details on : https://hub.docker.com/r/ibmblockchain/fabric-peer/

Lesson #3 – Understand how local networks are set up

- When deploying Hyperledger fabric and demo apps on local systems, it is key to understand what ports are limited on your network
- If there are any firewalls/proxy between servers and your workstations – make sure what ports/protocols can be passed. Some of the demo apps utilize WebSocket and some firewalls/proxy may not work well with it. When it happens, try to make SSH tunneling or make HTTPS for the app server
- Test REST API calls for Hyperledger using utilities like PostMan app
- For v1.0 Hyperledger fabric, use SDK tools like composer-cli, composer-playground and REST Server etc.

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
```

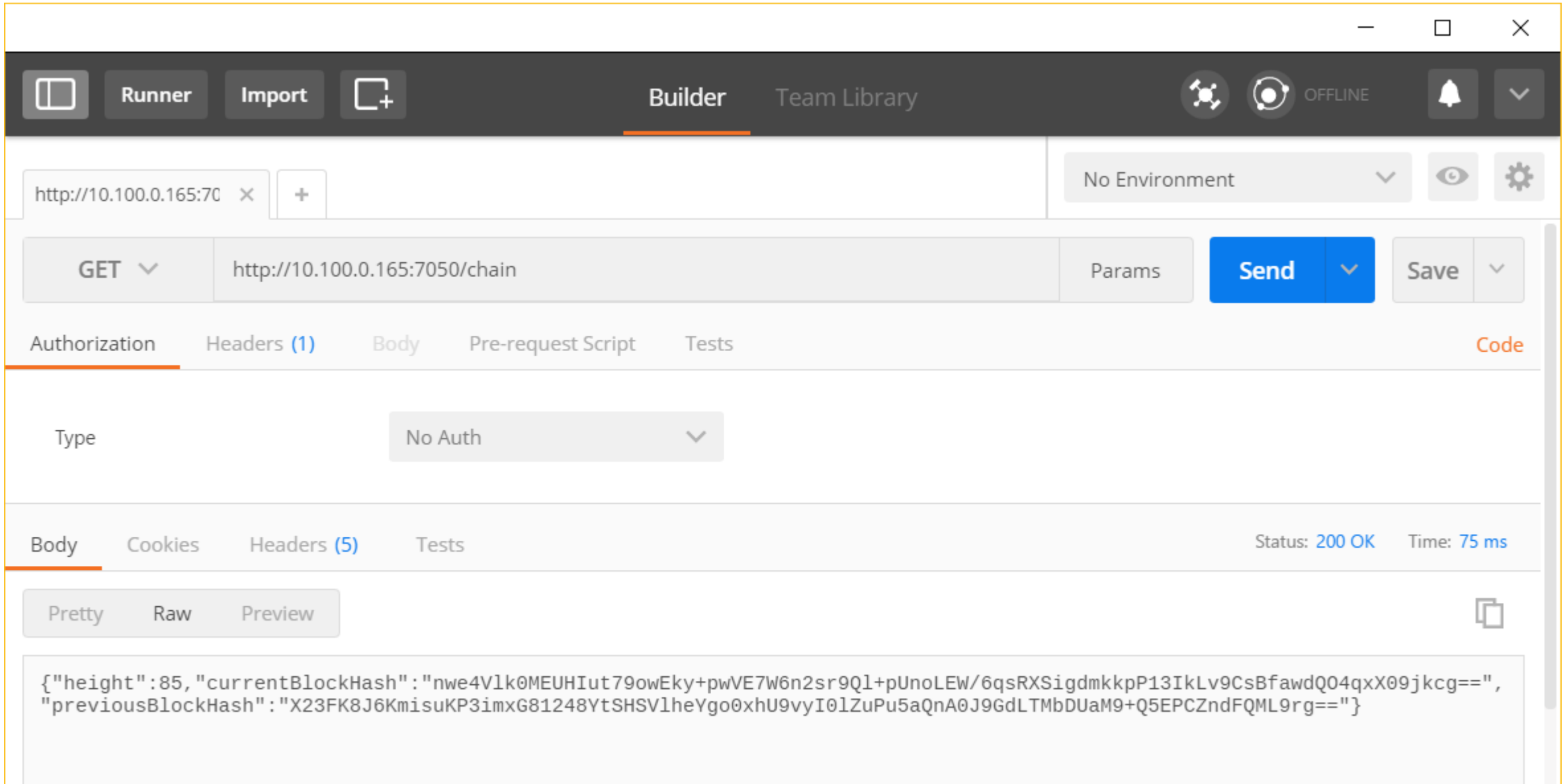
```
message BlockchainInfo {
    uint64 height = 1;
    bytes currentBlockHash = 2;
    bytes previousBlockHash = 3;
}
```

Blockchain Retrieval Response:

Number of Blocks in the Blockchain


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

PostMan example(Chrome App)



Lesson #4 – prepare enough storage space if you want to explore longer

- Docker images for the fabric as well as chaincodes will require a good amount of storage(disk) space – plan for expansion in case needed and/or change default /var/lib/docker image storage to larger amount storage volume mount

- Monitor frequently to see how much memory/storage is consumed for your blockchain apps

- For development environment, encourage using Bluemix or local on laptop. For performance test for real application, HSBN on Bluemix is recommended until on-prem solution for zSystems become available

Lesson #5 – connect with who already worked on it

- I got so much help from colleagues at IBM and customers
- Using Slack/github/RocketChat I could get quick responses from many people - https://chat.hyperledger.org
- Special thanks to (Neale Fergurson)@SineNomine, (Volodymyr Paprotski, John Harrison, Barry Silliman, Dave Huffman)@IBM, (Sam D'Angelo, Vincent Terrone)@AIG

What's Next?

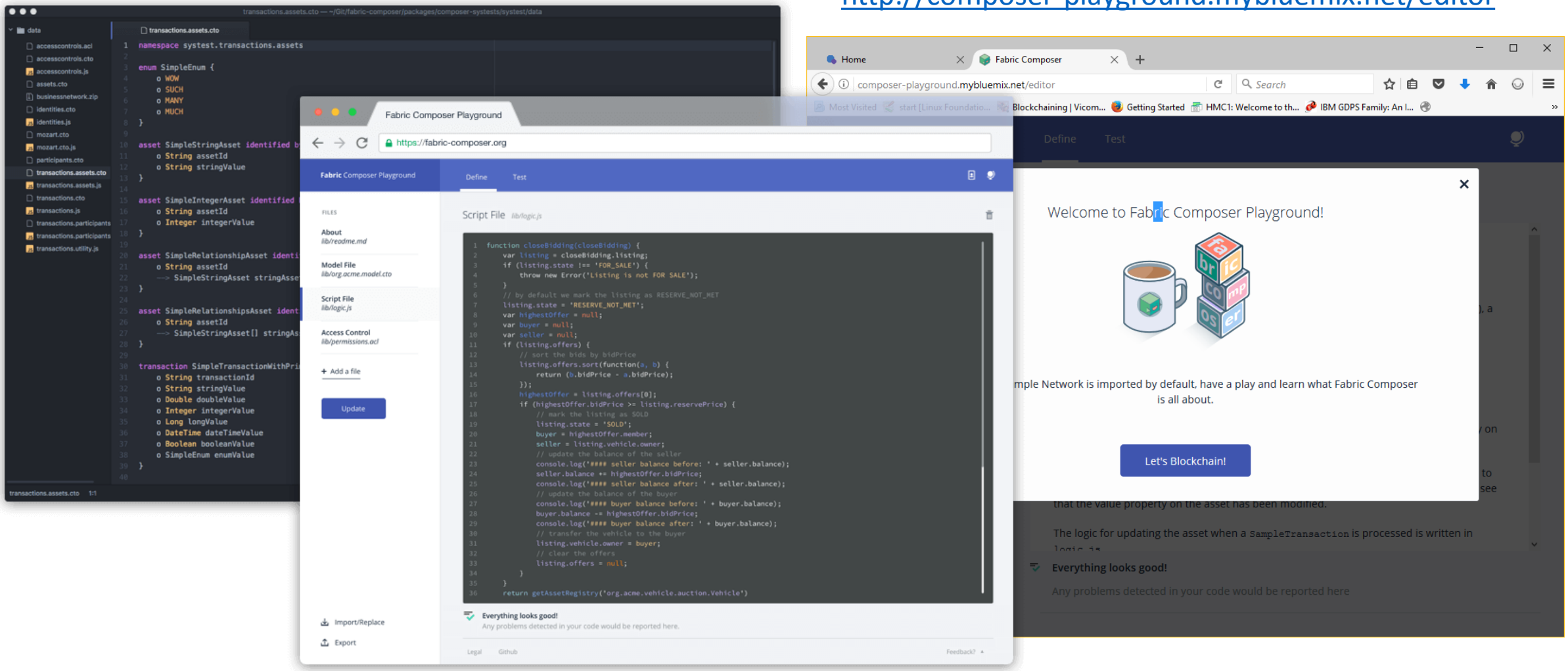
- Hyperledger™ Fabric v1.0beta is announced and available on Bluemix

  (limited open beta for HSBN vNext is available now)
- Fabric-composer, composer-playground can be very helpful to create an app to deploy business logics
- More from zSysetms to be highlighted supporting Hyperledger™
- A full day workshop on Hyperledger™ on August 6 @SHARE Academy

  http://www.share.org/p/cm/ld/fid=1245

Fabric Composer - https://fabric-composer.github.io/

http://composer-playground.mybluemix.net/editor



Pricing Plans

Monthly prices shown are for country or region: United States

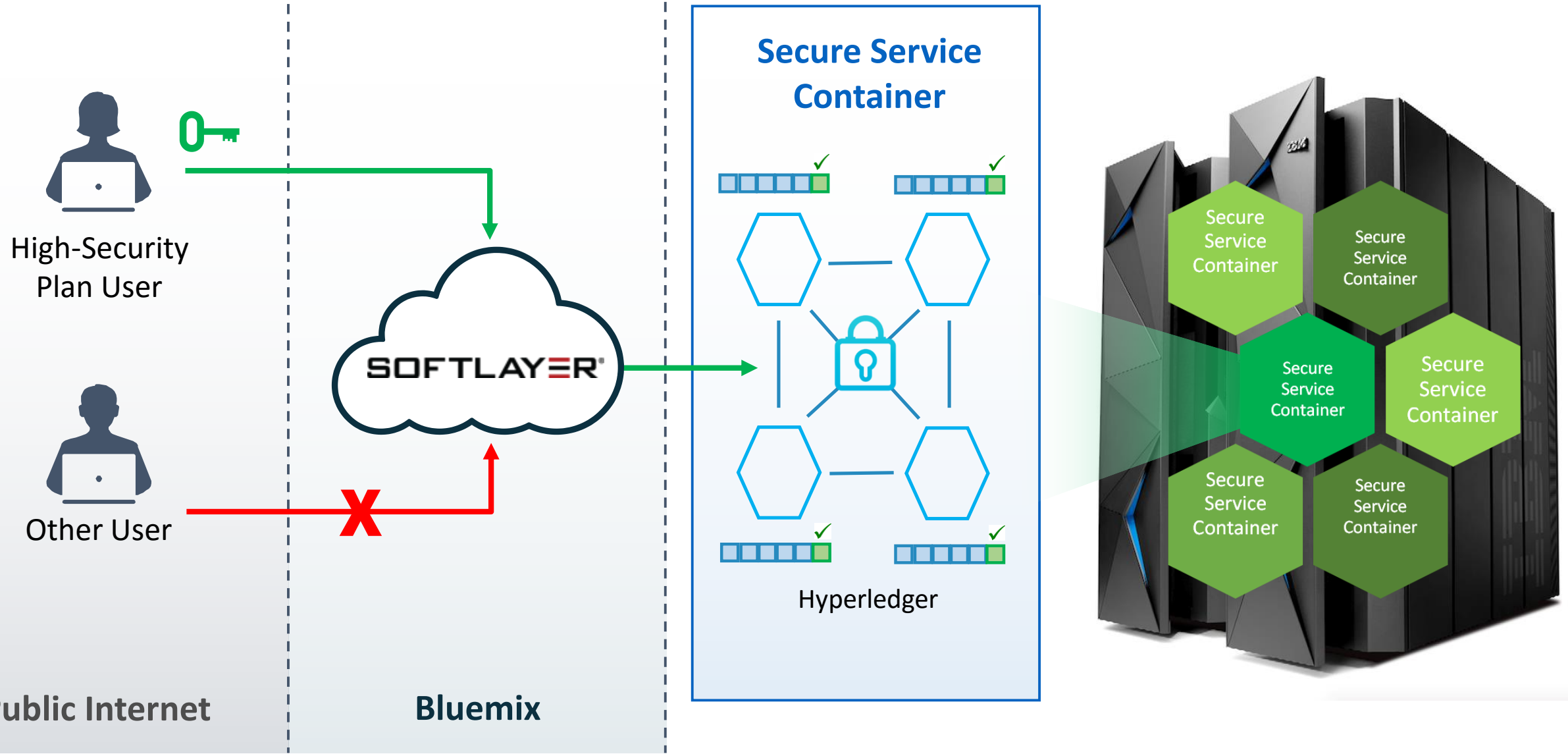
	PLAN	FEATURES	PRICING
✓	**Starter Developer plan (beta)**	**- 4 peers and a Cert Authority** - Based on Hyperledger Fabric v0.6 architecture - Dashboard with logs, controls, and APIs - Sample apps with source code	Free
		Get started using IBM Blockchain! Monitor your network and view health status. Leverage the REST API to deploy and invoke chaincode transactions.	
	High Security Business Network vNext(Limited Beta)	**- Bootstrap dynamic blockchain networks** - Uses the latest Hyperledger Fabric v1.0 architecture - Members provision their own resources - Create multiple communication channels	Free
	High Security Business Network plan	**- 4 peers and a Cert Authority on IBM LinuxOne™** - Based on Hyperledger Fabric v0.6 architecture - Isolated environment on dedicated compute - Optimized performance and high speed network - Advanced Security: HSM, Secure Service Container and more	$10,000.00 USD/MONTHLY

More details - http://bit.ly/2nAtIQN

BACK UP Charts

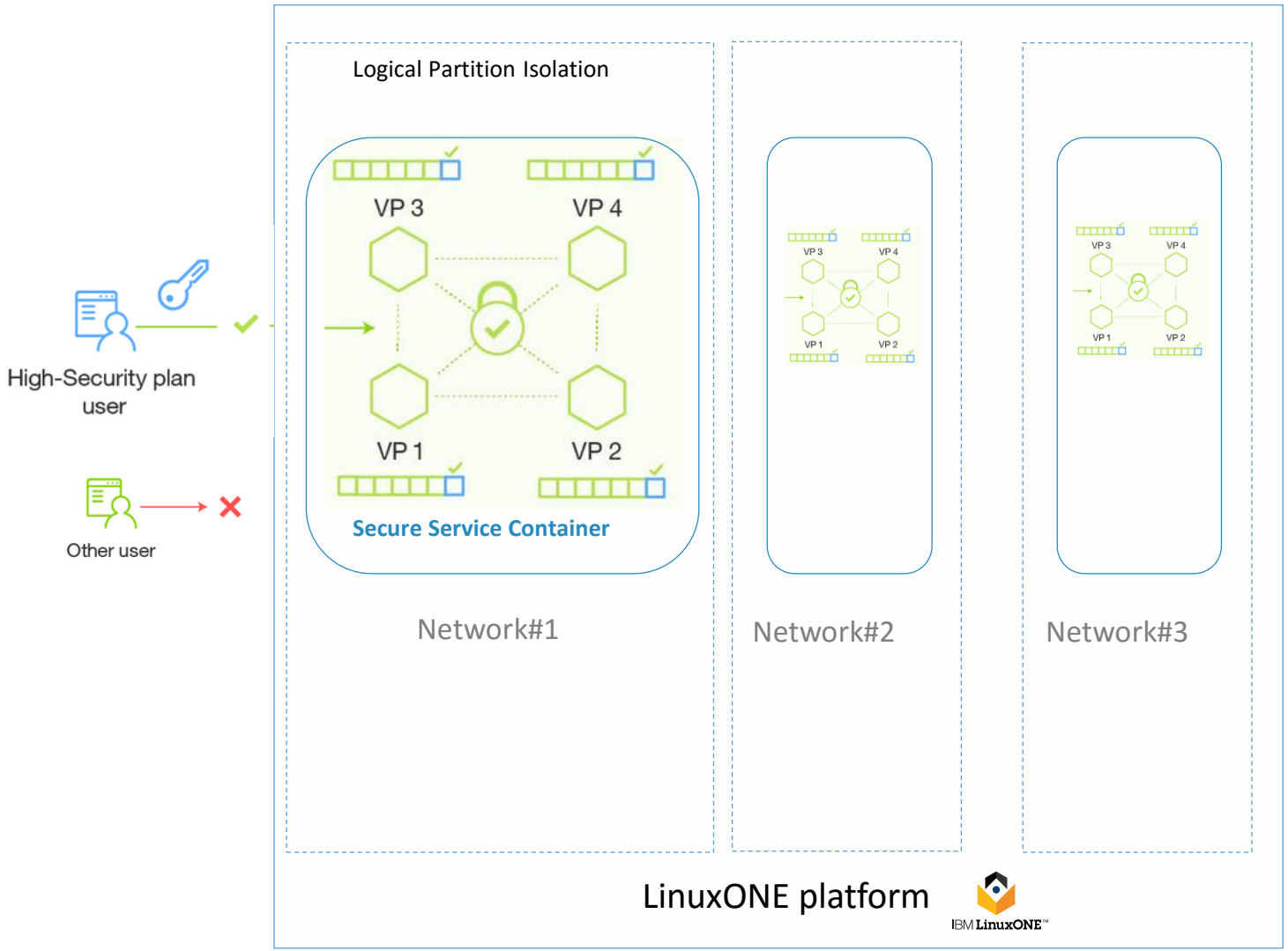
Architecture – Overview

High Security Business Network



Architecture – High Level

High Security Business Network

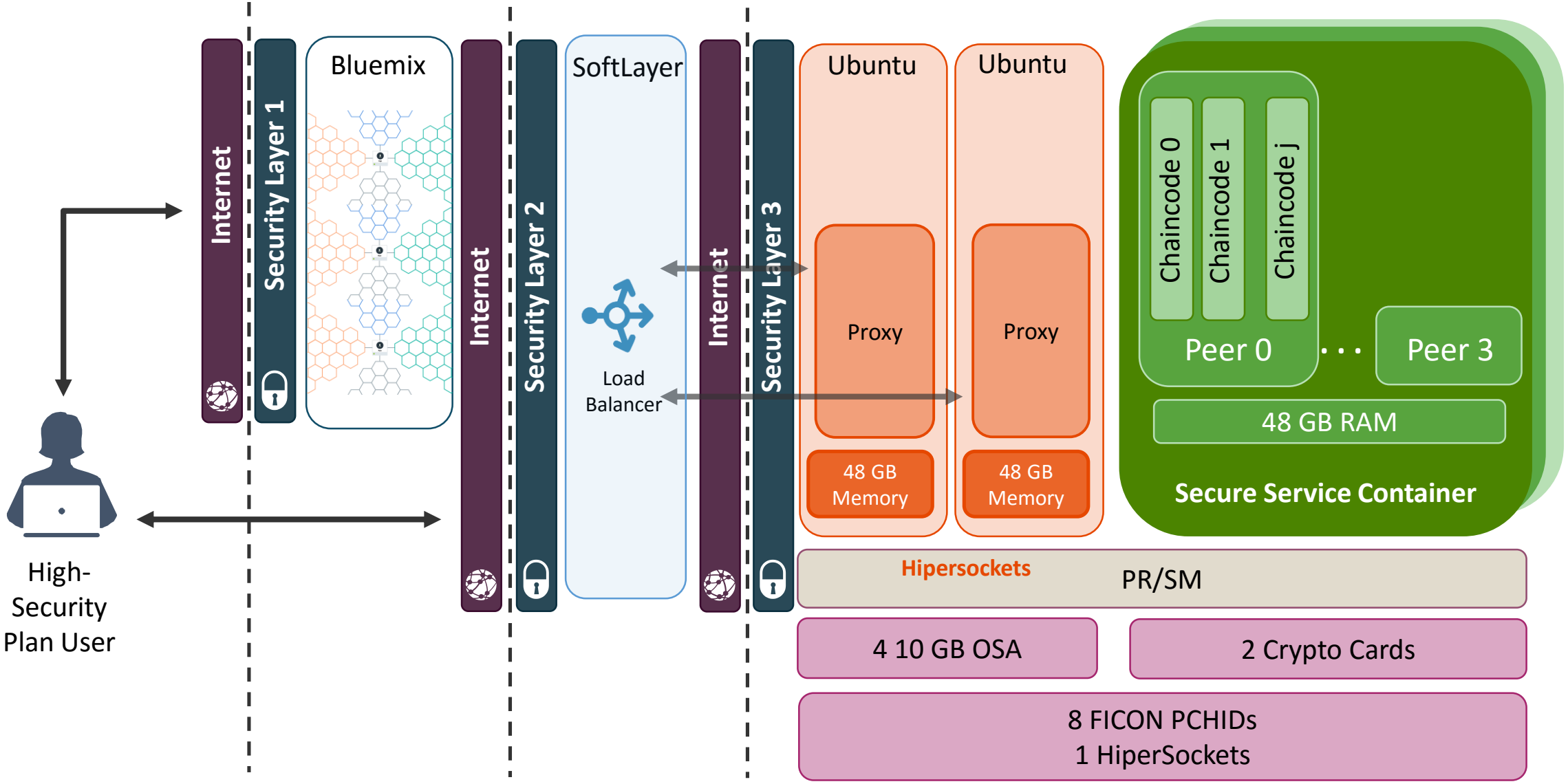


The high security business network is deployed as an appliance into a Secure Service Container, which provides the base infrastructure for hosting blockchain services. The appliance combines operating systems, Docker, middleware, and software components that work autonomously to provide core services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

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

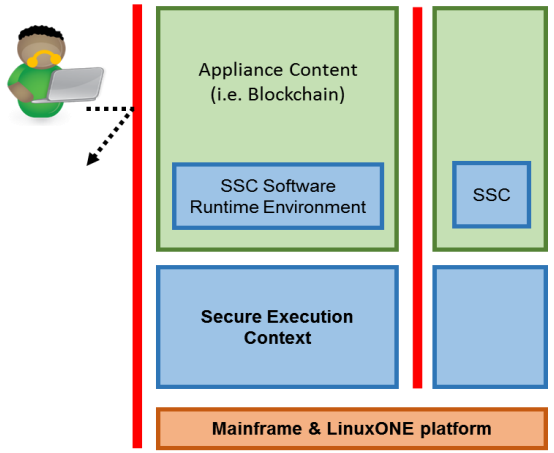
Reference Architecture

High Security Business Network



Secure Service Container

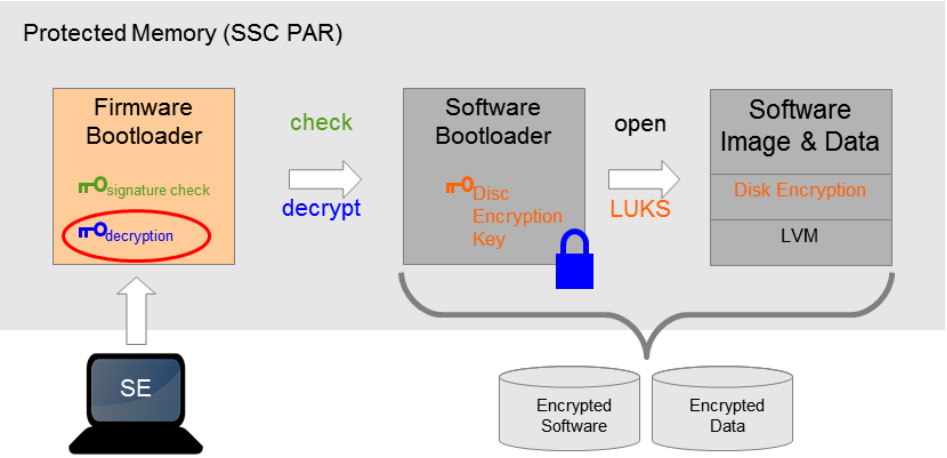
Secure Service Container ensures…



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

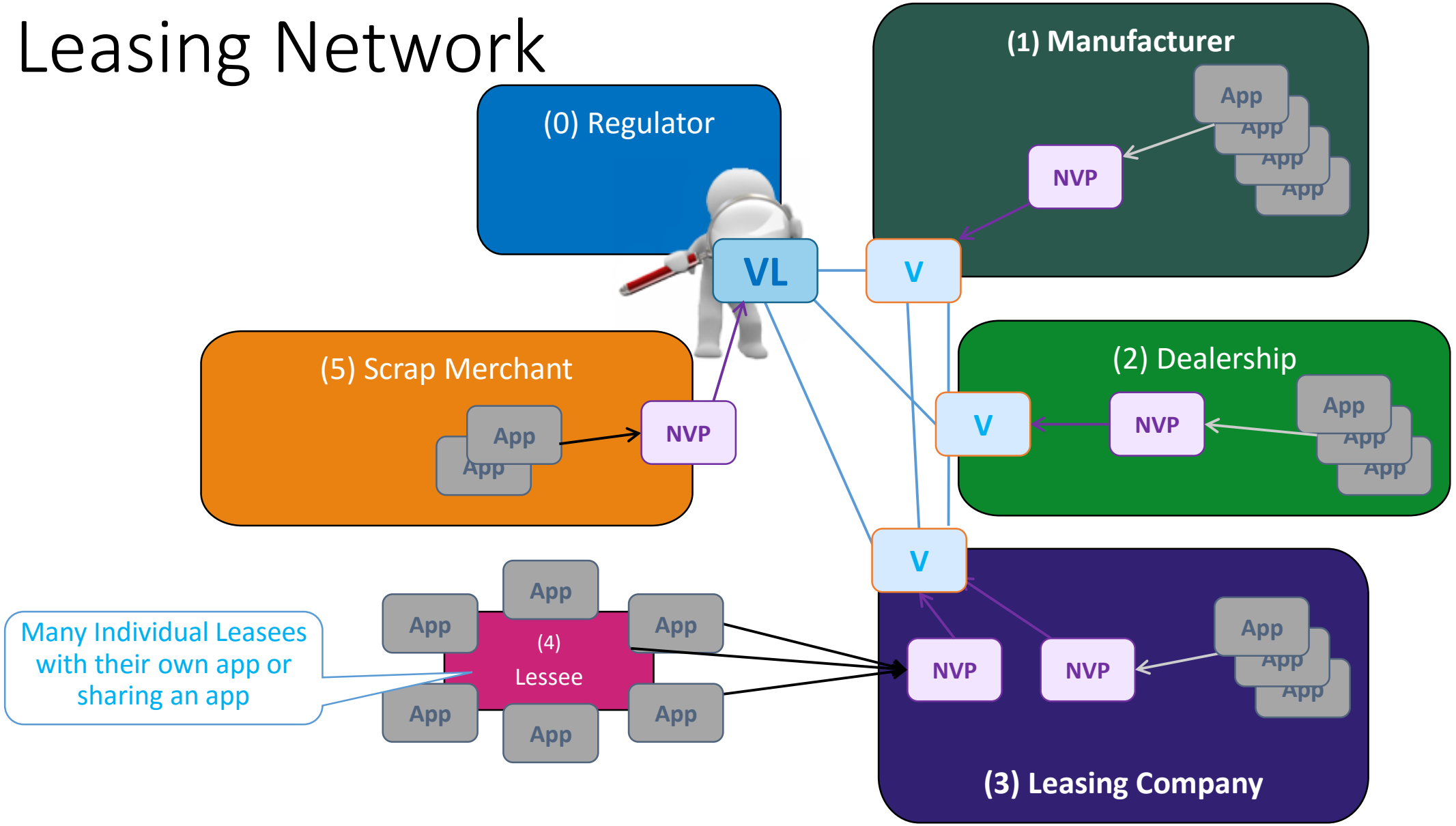
How the Secure Service Container boot sequence works…



Boot sequence

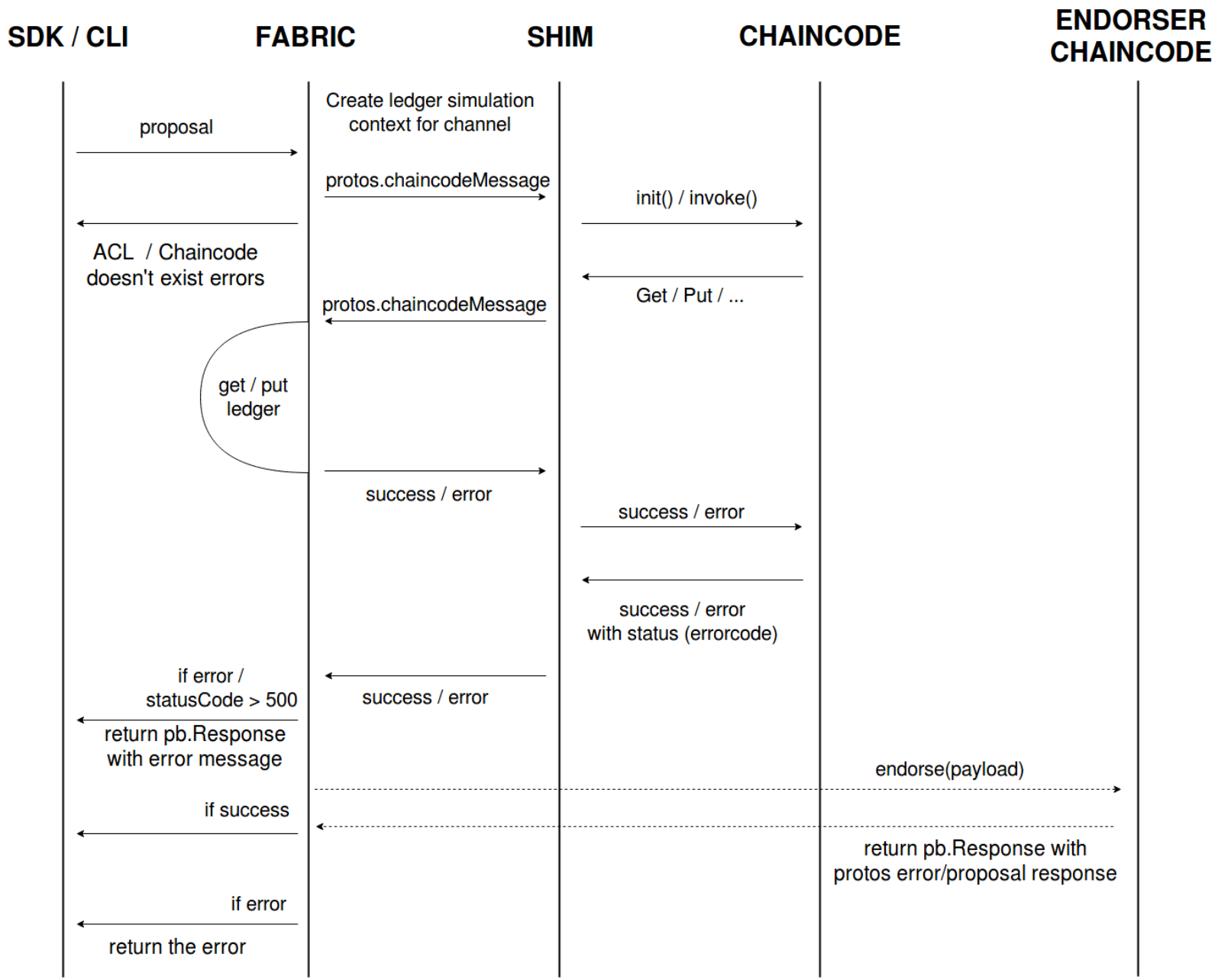
1. Firmware bootloader is loaded in memory
2. Firmware loads the software bootloader from disk
   i. Check integrity of software bootloader
   ii. Decrypt software bootloader
3. Software bootloader activate encrypted disks
   i. Key stored in software bootloader (encrypted)
   ii. Encryption/decryption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

Car Leasing Network



How is Hyperledger Fabric different from other blockchain implementations?

	Bitcoin	Ethereum	Hyperledger
Cryptocurrency required	bitcoin	ether, user-created cryptocurrencies	none
Network	public	public or permissioned	permissioned
Transactions	anonymous	anonymous or private	public or confidential
Consensus	proof of work	proof of work	PBFT
Smart contracts (business logic)	none	yes (Solidity, Serpent, LLL)	yes (chaincode)
Language	C++	Golang, C++, Python	Golang, Java


SDK / CLI
FABRIC
SHIM
CHAINCODE
ENDORSER CHAINCODE
Create ledger simulation context for channel
proposal
protos.chaincodeMessage
init() / invoke()
ACL / Chaincode doesn't exist errors
Get / Put / ...
protos.chaincodeMessage
get / put ledger
success / error
success / error
success / error with status (errorcode)
if error / statusCode > 500
success / error
return pb.Response with error message
endorse(payload)
if success
return pb.Response with protos error/proposal response
if error
return the error