Housekeeping

- Thank you to our venue sponsor - AWS
- Volunteer
- Slides and recording on meetup.com after the event
- Hyperledger Hackfest Q1 2019 in Hong Kong
Poll – Hyperledger APAC Hackfest in Q1 2019

<table>
<thead>
<tr>
<th>19 participants</th>
<th>Week of January 14th</th>
<th>Week of February 25th</th>
<th>Week of March 4th</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edmund To</td>
<td>☑️ 10</td>
<td>☐️ 2</td>
<td>☑️ 14</td>
</tr>
<tr>
<td>davidZ</td>
<td>☐️</td>
<td>☑️</td>
<td>☑️</td>
</tr>
<tr>
<td>Jacky Ting</td>
<td>☑️</td>
<td>☑️</td>
<td>☑️</td>
</tr>
<tr>
<td>Zhujun He</td>
<td>☐️</td>
<td>☑️</td>
<td>☑️</td>
</tr>
<tr>
<td>Tong Li</td>
<td>☑️</td>
<td>☑️</td>
<td>☑️</td>
</tr>
<tr>
<td>Scott Long</td>
<td>☑️</td>
<td>☑️</td>
<td>☑️</td>
</tr>
</tbody>
</table>
About our speaker - Duncan Johnston-Watt

- CEO & Co-founder
- Blockchain Technology Partners
  - Corporate Hyperledger Member
- Hyperledger Speaker Bureau member
- Hyperledger Scotland Meetup organizer
- Twitter/WeChat: @duncanjw
Deploying and managing Hyperledger Sawtooth

Hyperledger HK Meetup – August 2018
Introducing BTP
Blockchain Primer
Hyperledger Sawtooth
BTP Platform
Demo
Vision

- Radically simplify enterprise adoption of blockchain technologies
- Reduce cost & complexity of doing business through decentralization
- Ensure trust, transparency and accountability in a distributed world

Bring the benefits of blockchain to enterprise
Focus

- We provide our customers with permissioned blockchain platforms built from open source components Hyperledger Sawtooth and Kubernetes (CNCF) backed by the Linux Foundation.

- We work with our customers to co-create blockchain-based solutions that differentiate their business from their competitors.

- We support our customers and in collaboration with them create an operational model that meets their needs now and in the future.
Co-creation is a management initiative, or form of economic strategy, that brings different parties together (for instance, a company and a group of customers), in order to jointly produce a mutually valued outcome.

https://en.wikipedia.org/wiki/Co-creation
What is a blockchain?

- At its core an expanding list of cryptographically signed, irrevocable transactional records shared by all participants in a network.
- Each record contains a time stamp and reference links to previous transactions.
- With this information, anyone with access rights can trace back a transactional event, at any point in its history, belonging to any participant.

Blockchain-Based Transformation: A Gartner Trend Insight Report, March 2018
What are some of its capabilities?

- Represent assets digitally
- Enable new forms of value exchange e.g. through tokens
- Interact/transact without a central authority or a middleman
- Ensure distributed copies of identical records that are immutable and traceable
- Enable management, governance and execution of partnerships and contracts across entities

Blockchain-Based Transformation: A Gartner Trend Insight Report, March 2018
## How is it implemented?

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Permissionless public ledgers</strong> e.g. Bitcoin blockchain and Ethereum network</td>
<td>Open to anyone to participate in the network; there is no external control to govern the participation or limit access. It enables some anonymity, a property that is not usually desirable in regulated organizations. In a permissionless deployment model, transactions are fully viewable, but the participants are identified only through a cryptographically derived account address.</td>
</tr>
<tr>
<td><strong>Permissioned consortium ledgers</strong> e.g. R3 Corda, Hyperledger Sawtooth/Fabric</td>
<td>Operate on behalf of a community of interest, but ecosystem and network access is limited to authenticated and approved participants. These systems have trusted transaction validators and can serve single or multiple industries.</td>
</tr>
<tr>
<td><strong>Permissioned private ledgers</strong> e.g. Hyperledger Sawtooth/Fabric, Chain, Bankchain or SETL</td>
<td>Operate exclusively within a defined community of known/trusted participants such as financial institutions and government agencies, and the community (or designated authority) controls access and contribution to the ledger. These systems have single or multiple trusted transaction validators.</td>
</tr>
</tbody>
</table>
Hyperledger Sawtooth

- Modular platform for building, deploying, and running distributed ledgers
- Includes a novel consensus algorithm, Proof of Elapsed Time (PoET), which targets large distributed validator populations with minimal resource consumption
- Provides clear separation between the application level and the core system level enabling developers to focus on implementing business use case
Hyperledger Sawtooth

- Sawtooth provides a smart contract abstraction that lets application developers write contract logic in a language of their choice.
- An application can be a native business logic or a smart contract virtual machine providing Ethereum contract compatibility.
- Both types of applications can co-exist on the same blockchain.
- Sawtooth enables these design decisions in the transaction-processing layer, which lets multiple types of applications coexist in the same instance of the blockchain network.
- Sawtooth’s own configuration is implemented on chain using a set of transaction-processors.
Hyperledger Sawtooth

Source: https://sawtooth.hyperledger.org/docs/core/releases/1.0/app_developers_guide/aws.html#overview-of-sawtooth-components
Agenda

Sawtooth Design Motivations

Sawtooth 1.0 Features

Distributed App Development

- Code: [https://github.com/hyperledger/sawtooth-core](https://github.com/hyperledger/sawtooth-core)
- Demos: [https://sawtooth.hyperledger.org/examples/](https://sawtooth.hyperledger.org/examples/)
- Docs: [https://sawtooth.hyperledger.org/docs/](https://sawtooth.hyperledger.org/docs/)
Announcing Hyperledger Sawtooth 1.0 !!!

Hyperledger Sawtooth is an open source distributed ledger framework and one of the nine business blockchain and distributed ledger technologies hosted by The Linux Foundation. Hyperledger Sawtooth delivers unique capabilities. A few examples are included below:

- **On-chain governance** – Utilize smart contracts to vote on blockchain configuration settings such as the allowed participants and smart contracts.
- **Advanced transaction execution engine** – Process transactions in parallel to accelerate block creation and validation.
- **Support for Ethereum** – Run solidity smart contracts and integrate with Ethereum tooling.
- **Dynamic consensus** – Upgrade or swap the blockchain consensus protocol on the fly as your network grows, enabling the integration of more scalable algorithms as they are available.
- **Broad language support** – Program smart contracts in your preferred language, with support including Go, JavaScript, Python and more.

The efforts around Hyperledger Sawtooth have grown significantly; from the initial code contribution in April 2016, to **Active status graduation** in May 2017, to today’s version 1.0 availability. Hyperledger Sawtooth is supported by an active community: organizations including Amazon Web Services, Active Ticketing, Bitwise.io, Cloudsoft, Context Labs, Dot BC Media, Ericsson, Hacera, Huawei, IBM, Intel, Microsoft Azure, Monax, Open Music Initiative, PokitDok, R3, T-Mobile, Wind River and more than 50 engineers have contributed to the project. Additionally, Proof of Concepts (PoC’s) have been deployed to support multiple business cases including music and media content rights attribution, recording healthcare transactions, Know Your Customer (KYC) in financial services and others.
Sawtooth Design Philosophy

- Design for Scale
- Keep Distributed Ledgers Distributed
- Ease-of-use for Development & Deployment
- Make Smart Contracts Safe
v1.0 Highlighted New Features

• Advanced Transaction Execution
  • Parallel Execution
  • Multi-Language Support
    • Build apps in your language of choice

• On-chain Governance
  • Dynamic Consensus
  • Proof of Elapsed Time (PoET)
  • New Permissioning Features

• Seth
  • Sawtooth + Ethereum
  • Run Solidity on Sawtooth

• Distributed Applications
  • Marketplace
    • Asset management
  • Supply Chain
    • Provenance of goods
    • Telemetry / tracking
Basic Concept

Clients:
- Submits transactions; Queries the database

Validator Process:
- Mediates access to the database
- State

Transaction Processors:
- Business logic; Validates transactions

Sawtooth Hosts
A Couple More Pieces

Sawtooth Host

Clients

REST Service

Validator Process

State

Transaction Processor(s)

Other Validators
High-level Sawtooth Architecture

Sawtooth Host

Clients

REST Service

Validator

Interconnect

Block Management

Transaction Handling

Consensus

State

P2P Network

Transaction Processor(s)

Other Validators
v1.0 Features: Parallel Execution

Multiple transactions per block can affect the same state

Parallel scheduling

Multi-process “smart contracts”
v1.0 Features: Multi-Language Support

Sawtooth Host

- REST Service
- Validator
  - Interconnect
  - Block Management
  - Transaction Handling
  - Consensus
  - State
  - P2P Network

Clients

Choose from:
- Python
- JavaScript
- Go
- Rust*

Transaction Processor(s)

Choose from:
- Go
- JavaScript
- Python
- C++*, Java*, Rust*

*No all features are available in these languages in v1.0.1
v1.0 Features: Seth

- Seth client
- REST Service
- Seth RPC
- PoET
- State
- Identity
- Settings
- BlockInfo
- Seth Transaction Processor
v1.0 Features: On-chain Governance

Control the blockchain on the blockchain

Settings Transaction Family enables participants to agree on network policies.

For example, vote on changing consensus parameters using registered public keys of consortia members.

Settings are extensible – they can be added after genesis.

<table>
<thead>
<tr>
<th>Setting (Examples)</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>sawtooth.poet.target_wait_time</td>
<td>5</td>
</tr>
<tr>
<td>sawtooth.validator.max_transactions_per_block</td>
<td>100000</td>
</tr>
<tr>
<td>sawtooth.validator.transaction_families</td>
<td>[{</td>
</tr>
<tr>
<td></td>
<td>&quot;family&quot;: &quot;intkey&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;version&quot;: &quot;1.0&quot;</td>
</tr>
<tr>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;family&quot;: &quot;xo&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;version&quot;: &quot;1.0&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
v1.0 Features: Dynamic Consensus

Set Consensus := PoET

Consensus = DevMode

Abstract Consensus

DevMode  PoET  PoET-SGX  Raft*  PBFT*

*Future consensus options

Sawtooth Host

client

REST Service

Settings

Permissioning

Other Smart Contracts
Hyperledger Sawtooth 1.0
Application Development

1.0 Released January 2018
Application Development

Transaction Family

Client
- Transaction Creation

Validator Process
- State
  - Data Model

Transaction Processor
- Business Logic
Transaction Processor ≈ Smart Contracts

Transaction Families **encapsulate business logic** on Sawtooth

A Transaction Family can be as simple as a single transaction format, with associated validity and state update logic…

…or as complex as a VM with opcode accounting and bytecode stored in state -- ‘smart contracts’

The *choice* is up to the developer

Sawtooth allows these concepts to **coexist** in the same instance of the blockchain -- same blocks, same global state
Transaction Families: The Transaction Processor

All validators in the network run every authorized transaction processor.

On receipt of a transaction the validator will call the TP’s Apply() method.

Business logic simply goes in Apply() and gets and sets state as needed.
Transaction Families: The Client

Clients can be browser apps, CLIs, etc.

Main job is to package and sign transactions & batches

Clients can post batches through the Rest API or connect to the validator directly
Transactions are wrapped in batches which provide an atomic unit of commit for multiple transactions (which can span transaction families).

Transactions declare input and output addresses (including wildcards) to allow for state access isolation calculations (topological sort on DAG) in the scheduler.

These inputs and outputs are enforced by the Context Manager on the context established for the transaction.

This allows parallel validation and state delta aggregation across a potentially large number of transactions (and across blocks).
Transaction Families: The Data Model

Both Client and Transaction Processor must use the same...

- Data model
- Serialization / encoding
- Addressing scheme
Example Applications & Code

Supply Chain: https://github.com/hyperledger/sawtooth-supply-chain

Marketplace: https://github.com/hyperledger/sawtooth-marketplace

RBAC: https://github.com/hyperledger/sawtooth-next-directory

Dev Guide: https://sawtooth.hyperledger.org/docs/core/releases/latest/app_developers_guide.html
Check it out

Give Sawtooth a try
- Work through the tutorials
- Build your own transaction family to explore use cases

Become a contributor
- Join the community
- Help with docs, code, examples
- Become an expert and help others on chat

Links
- Code: https://github.com/hyperledger/sawtooth-core
- Docs: https://sawtooth.hyperledger.org/docs/
- Chat: https://chat.hyperledger.org/channel/sawtooth
BTP Platform

- BTP Platform
  - Hyperledger Sawtooth distribution
  - Branded, curated and hardened
  - Provides unified user experience

- BTP Platform Editions
  - Starter
  - Professional
  - Enterprise

- AWS Marketplace (Q4 18)
  - Starter & Professional Editions
  - Easy to fire up and get started
  - PAYG pricing

- Future Targets
  - Azure Marketplace
  - Alibaba Cloud Marketplace
  - Google Cloud Platform Marketplace
BTP Reference Architecture

Process Nodes

Step 1 → Step 2 → Step 3 → Step 4 → Step 5

Distributed Ledger

BTP Platform

Gateway Nodes

Regulator → Smart City

Workshop - August 2018
BTP Reference Architecture

- **Process Nodes**
  - Implement key workflows

- **Distributed Ledger**
  - Ensures integrity of transactions

- **Gateway Nodes**
  - Key integrations e.g.
    - Regulators
    - Smart Cities
    - Downstream services
Create Kubernetes Cluster

- Target is AWS
- Standard tools: kops & kubectl

- [https://github.com/kubernetes/kops/releases/tag/1.10.0](https://github.com/kubernetes/kops/releases/tag/1.10.0)
- [https://github.com/kubernetes/kops/blob/master/docs/aws.md](https://github.com/kubernetes/kops/blob/master/docs/aws.md)
- [https://kubernetes.io/docs/setup/custom-cloud/kops/](https://kubernetes.io/docs/setup/custom-cloud/kops/)
Create Kubernetes Cluster

export KOPS_STATE_STORE=s3://clusters.dev.catenasys.com

cd ~/git/catenasys/kubernetes-manifests/clusters/

kops create -f duncanjw.yaml

kops create secret --name duncanjw.dev.catenasys.com \ 
sshpublickey admin -i ~/.ssh/kubernetes-duncanjw.pub

kops update cluster duncanjw.dev.catenasys.com --yes

kops validate cluster
Sawtooth Overview

package clusters

```
clusters {
    clusters {
        cluster {
            service {
                sawtooth-next
                type = "None"
            }
            service {
                sawtooth-rest-api
                type = "LoadBalancer"
            }
            service {
                sawtooth-validator
                type = "None"
            }
            service {
                influxdb
                type = "None"
            }
            service {
                grafana
                type = "LoadBalancer"
            }
            statefulset {
                sawtooth-next
            }
            statefulset {
                sawtooth-validator
            }
            statefulset {
                sawtooth-monitoring
            }
        }
    }
}
```
Sawtooth Next Directory
Sawtooth Monitoring

```yaml
package sets [ monitoring.yaml ]

- service
  - grafana
    { type = "LoadBalancer" }
  - influxdb
    { type = "None" }

- statefulset
  - sawtooth-monitoring

- pod
  - sawtooth-monitoring
    - container
      - grafana
    - container
      - influxdb
```
Sawtooth Validator

[Diagram of Sawtooth Validator architecture]

Workshop - August 2018
Create Sawtooth Network

- Target is our new cluster `duncanjw.dev.catenasys.com`
- We are going to use a sample Sawtooth PoET manifest
- At this stage we have a vanilla cluster

```bash
kubectl get all -o wide
cd ../../../manifests/
kubectl apply -f sawtooth-demo.yaml
watch -n 5 kubectl get all -o wide
```
Explore Sawtooth Network

- First we look at some logs

```bash
kubect1 logs sawtooth-validator-0 sawtooth-validator
```

- Next we fire up Grafana

- To do this we need the EXTERNAL-IP advertised by `service/grafana`

```bash
kubect1 get all -o wide
```
Let’s play XO (tic-tac-toe)!

- **REST API**
  - [https://sawtooth.hyperledger.org/docs/core/releases/1.0/architecture/rest_api.html](https://sawtooth.hyperledger.org/docs/core/releases/1.0/architecture/rest_api.html)

- **Overview**
  - [https://sawtooth.hyperledger.org/docs/core/releases/1.0/app_developers_guide/intro_xo_transaction_family.html](https://sawtooth.hyperledger.org/docs/core/releases/1.0/app_developers_guide/intro_xo_transaction_family.html)

- **Specification**
  - [https://sawtooth.hyperledger.org/docs/core/releases/1.0/transaction_family_specifications/xo_transaction_family.html](https://sawtooth.hyperledger.org/docs/core/releases/1.0/transaction_family_specifications/xo_transaction_family.html)
Let’s play XO (tic-tac-toe)!

- First we fire up a `sawtooth-shell`

```
docker run -it hyperledger/sawtooth-shell
```

- Then do some housekeeping

```
export URL="--url http://<rest-api-external-ip>:8080"
```
```
sawtooth keygen jack
```
```
sawtooth keygen jill
```
Let’s play XO (tic-tac-toe)!

- Now we can create & play a game using the xo-tp client

```bash
xo create $URL game1 --username jack ; xo list $URL
xo take $URL game1 5 --username jack ; xo list $URL
# username jack is now PLAYER 1
xo take $URL game1 1 --username jill ; xo list $URL
# username jill is now PLAYER 2
xo show $URL game1
```
Now we add another validator

```
kubectl scale statefulsets sawtooth-validator --replicas=4
watch -n 5 kubectl get all -o wide
```

Once this is up we can check its log and see it in Grafana

```
kubectl logs sawtooth-validator-3 sawtooth-validator
```
We tear down the Sawtooth network starting with the stateful sets

```bash
kubectl delete statefulset.apps/sawtooth-validator \
statefulset.apps/sawtooth-next \
statefulset.apps/sawtooth-monitoring
kubectl get all -o wide
```
Delete Sawtooth Network

- Then free up the AWS load balancers & persistent volume claims

```sh
kubectl delete service/sawtooth-rest-api service/grafana

kubectl get pvc
kubectl delete pvc sawtooth-sawtooth-validator-0
sawtooth-sawtooth-validator-1
sawtooth-sawtooth-validator-2
sawtooth-sawtooth-validator-3
```
Delete Kubernetes Cluster

- Once we are sure everything has been garbage collected we can delete the cluster itself

```
kops delete cluster --name duncanjw.dev.catenasys.com --yes
```
Hyperledger Member Summit, Montreal, Oct 1 & 2
  ▪ https://events.linuxfoundation.org/events/hyperledger-member-summit/

Open Source Summit Europe, Oct 22-24
  ▪ https://events.linuxfoundation.org/events/open-source-summit-europe-2018/

Hyperledger Scotland Meetup, Oct 23
  ▪ https://www.meetup.com/Hyperledger-Scotland/events/253702788/

Hyperledger Global Forum, Basel, Dec 12-15
  ▪ https://events.linuxfoundation.org/events/hyperledger-global-forum-2018/
“BTP has a clear value proposition — bringing the benefits of blockchain to business — and its leadership team has the necessary expertise to bring together the components to deliver”

Csilla Zsigri & William Fellows, 451 Research Impact Report
“The US (29%), China (18%), Australia (7%) are perceived as the most advanced currently in developing blockchain projects. However within three to five years, respondents believe China will be have overtaken the US (30%), shifting the early centre of influence and activity from the US and Europe”

Blockchain is here. What’s your next move?, PwC’s Global Blockchain Survey 2018