Project Work

S.Gopinath

National Informatics Centre

Title: Trustable Logs using Blockchain

Advisor: Prof. S.Revathi

Trustable Logs using Blockchain

S.Gopinath

BSA Crescent Institute of Science & Technology

Jul 2020

Problem Statement I

- Events are recorded as *logs*.
- Logs are used for Debugging, Capacity Planning, Accounting etc.
- Regulatory Authorities and Investigating agencies needs *logs!*
- Legacy log systems are mostly *tamperable!*
- WORM devices can address the issue (reasonably trustable) but it is costlier!
- Solution - Blockchain ? *Immutability*

Overview I

- Work $\Rightarrow$ {Phase-1, Phase-2}
- Two Styles of Blockchain Implementation. Style-1 and Style-2
- Blockchain Platform - Hyperledger Sawtooth Platform on Ubuntu
- Consensus Algorithm - PBFT
- Programming Language - Rust

Overview II

- Implementation - 4 Nodes Blockchain Cluster in AWS (for each Style)
- Style-1 - Single State - Multiple Data (*Transaction digging*
- Style-1 - Multiple States (*No Transaction digging*
- Off-Chaining - Logs are taken as *file-blocks* - Size-1024000 bytes $\Rightarrow$ SHA256
- *Partial Trust Establishment*

Literature Survey I

- Benedikt Putz,.., "A secure and auditable logging infrastructure based on a permissioned blockchain" *Elsevier* Nov 2019.

- Dr.Manish Kumar,.., "Secure Log Storage Using Blockchain and Cloud Infrastructure", *9th ICCCNT-2018*, IISc Bangalore

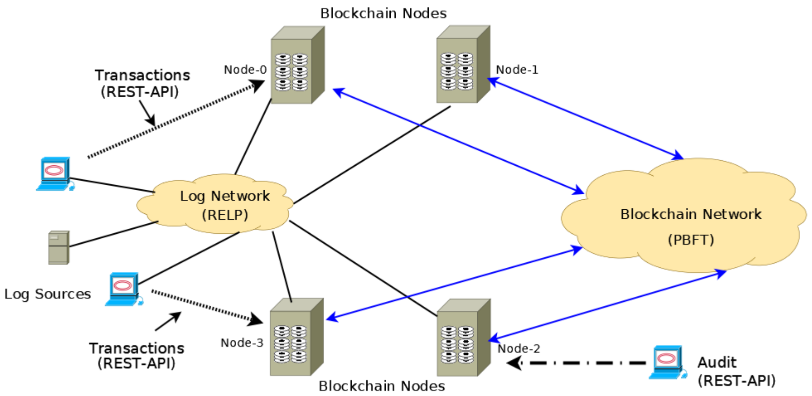
- Miguel Castro and Barbara Liskov, "Practical Byzantine Fault Tolerance"

- Vitalik Buterin "White Paper - Next Generation Smart Contracts and Decentralized Application Platform"

Literature Survey II

- Logan Seeley "Introduction to Sawtooth PBFT" *Blog Hyperledger Sawtooth*,Feb 2019
- Online Sites, Videos, Hyperledger and Rust forums, emails etc.

System Architecture I



Base Architecture

Drawn using dia

System Architecture I

- Logs to be delivered to all Blockchain nodes preferably in real time.
- Transport Example-**rsyslogd** with Reliable Event Log Protocol
- Transactions are generated by the clients associated with the log source. They sent to the Blockchain Network at fixed time (may be periodically).
- Each transaction represents a fixed size *file-blocks* in the log. The transaction payload comprises of file-block boundaries and its hash

System Architecture II

- *Partial Trust Establishment*
- After block generation and its addition to blockchain, the logs in the blockchain nodes can be removed.

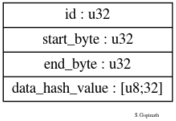
Software Modules I

- Top Level Modules - Client Module, Transaction Processor Module, Verification Module.
- Client - On behalf of log source.
  Log stream $\Rightarrow$ "Trans. Batches" $\Rightarrow$ Sawtooth Validator.
- Transaction Processor (TP)-Business Logic - runs in every node.
- Transaction Processor - Semantic Validation of Transactions.

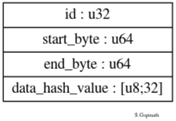
Software Modules II

- Semantic Validation - Hashes in Transactions are valid ?. Computes independently.
- PBFT ensures that the committed blocks are valid
- Verification Module - Given log - Integrity Checks - Partial Trust
- Programming Language - Rust

Client-Payload I



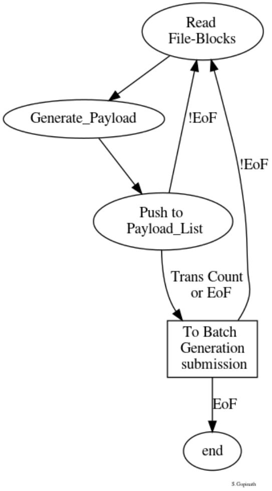
id : u32
start_byte : u32
end_byte : u32
data_hash_value : [u8;32]



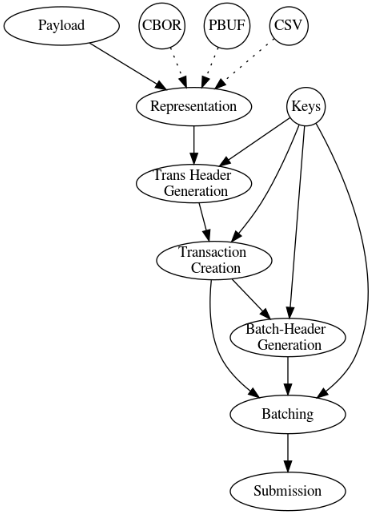
id : u32
start_byte : u64
end_byte : u64
data_hash_value : [u8;32]

- Payload is serialized using CBOR

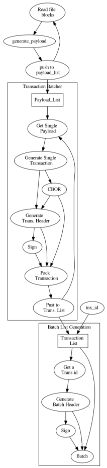
Client-Payload Generation



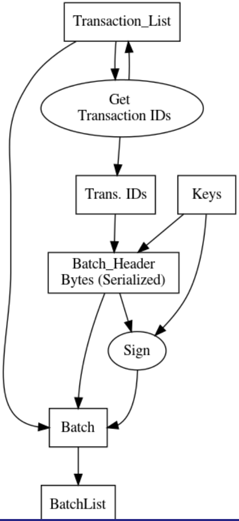
Client-Overall Flow-1



Client-Overall Flow-2



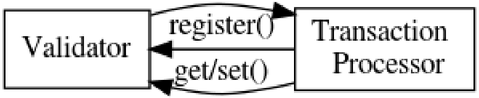
Batch Generation Flow



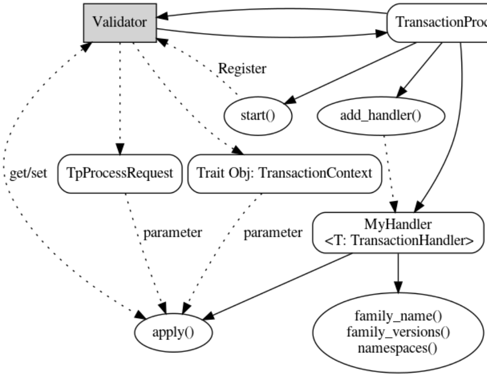
Transaction Processor Module

- Runs in every Sawtooth Node.
- Multiple Transaction Processor can run at a same time.
- get/set methods. set method is used to "set" the State with payload.
- Validator calls **apply** method of the TP which it implements.

Transaction Processor - Placement



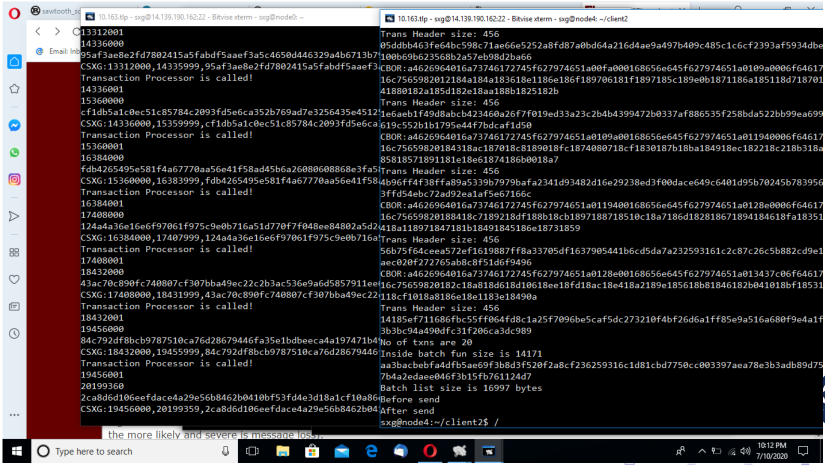
Transaction Processor- Flow


Validator
TransactionProcessor
Register
start()
add_handler()
get/set
TpProcessRequest
Trait Obj: TransactionContext
parameter
parameter
MyHandler <T: TransactionHandler>
apply()
family_name() family_versions() namespaces()

Verification Module

- Log source presents *log stream* for auditing.
- Verification program, compares the *hash* of submitted file-blocks with the payload data in the blockchain.
- Style-1 Verification - Digs Transactions from blockchain.
- Style-2 Verification - Takes payload in *States*
- REST-API for extracting blocks or States from the blockchain.

Style-1 Client (shows client and a TP)



10.163.tlp - sxg@14.139.190.162:22 - Bitvise xterm - sxg@node0: ~

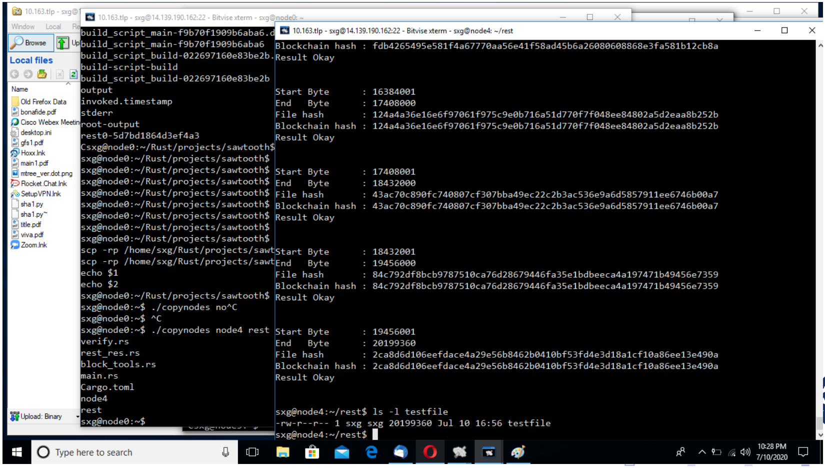
```
13312001
14336000
95af3ae8e2fd7802415a5fabdf5aaef3a5c4650d446329a4b6713b7
CSXG:13312000,14335999,95af3ae8e2fd7802415a5fabdf5aaef3
Transaction Processor is called!
14336001
15360000
cf1db5a1c0ec51c85784c2093fd5e6ca352b769ad7e3256435e4512
CSXG:14336000,15359999,cf1db5a1c0ec51c85784c2093fd5e6ca
Transaction Processor is called!
15360001
16384000
fdb4265495e581f4a67770aa56e41f58ad45b6a26080608868e3fa5
CSXG:15360000,16383999,fdb4265495e581f4a67770aa56e41f58
Transaction Processor is called!
16384001
17408000
124a4a36e16e6f97061f975c9e0b716a51d770f7f048ee84802a5d2
CSXG:16384000,17407999,124a4a36e16e6f97061f975c9e0b716a
Transaction Processor is called!
17408001
18432000
43ac70c890fc740807cf307bba49ec22c2b3ac536e9a6d5857911ee
CSXG:17408000,18431999,43ac70c890fc740807cf307bba49ec22
Transaction Processor is called!
18432001
19456000
84c792df8bcb9787510ca76d28679446fa35e1bdbeeca4a197471b4
CSXG:18432000,19455999,84c792df8bcb9787510ca76d28679446
Transaction Processor is called!
19456001
20199360
2ca8d6d106eefdace4a29e56b8462b0410bf53fd4e3d18a1cf10a86
CSXG:19456000,20199359,2ca8d6d106eefdace4a29e56b8462b04
```

10.163.tlp - sxg@14.139.190.162:22 - Bitvise xterm - sxg@node4: ~/client2

```
Trans Header size: 456
05ddbb463fe64bc598c71ae66e5252a8fd87a0bd64a216d4ae9a497b409c485c1c6cf2393af5934dbe
100b69b623568b2a57eb98d2ba66
CBOR:a4626964016a73746172745f627974651a00fa000168656e645f627974651a0109a0006f64617
16c7565982012184a184a183618e1186e186f189706181f1897185c189e0b1871186a185118d718701
41880182a185d182e18aa188b1825182b
Trans Header size: 456
1e6aeb1f49d8abcb423460a26f7f019ed33a23c2b4b4399472b0337af886535f258bda522bb99ea699
619c552b1b1795e44f7bdcaf1d50
CBOR:a4626964016a73746172745f627974651a0109a00168656e645f627974651a011940006f64617
16c75659820184318ac187018c8189018fc1874080718cf1830187b18ba184918ec182218c218b318a
85818571891181e18e61874186b0018a7
Trans Header size: 456
4b96ff4f38ffa89a5339b7979bafa2341d93482d16e29238ed3f00dace649c6401d95b70245b783956
3ffd54ebc72ad92ea1af5e67166c
CBOR:a4626964016a73746172745f627974651a0119400168656e645f627974651a0128e0006f64617
16c75659820188418c7189218df188b18cb1897188718510c18a7186d1828186718941846l8fa18351
418a118971847181b18491845186e18731859
Trans Header size: 456
56b75f64ceea572ef1619887ff8a33705df1637905441b6cd5da7a232593161c2c87c26c5b882cd9e1
aec020f272765ab8c8f51d6f9496
CBOR:a4626964016a73746172745f627974651a0128e00168656e645f627974651a013437c06f64617
16c75659820182c18a818d618d10618ee18fd18ac18e418a2189e185618b81846182b041018bf18531
118cf1018a8186e18e1183e18490a
Trans Header size: 456
14185ef711686fbc55ff064fd8c1a25f7096be5caf5dc273210f4bf26d6a1ff85e9a516a680f9e4a1f
3b3bc94a490dfc31f206ca3dc989
No of txns are 20
Inside batch fun size is 14171
aa3bacbebfa4dfb5ae69f3b8d3f520f2a8cf236259316c1d81cbd7750cc003397aea78e3b3adb89d75
7b4a2edaee046f3b15fb761124d7
Batch list size is 16997 bytes
Before send
After send
sxg@node4:~/client2$ /
```

the more likely and severe is message loss).

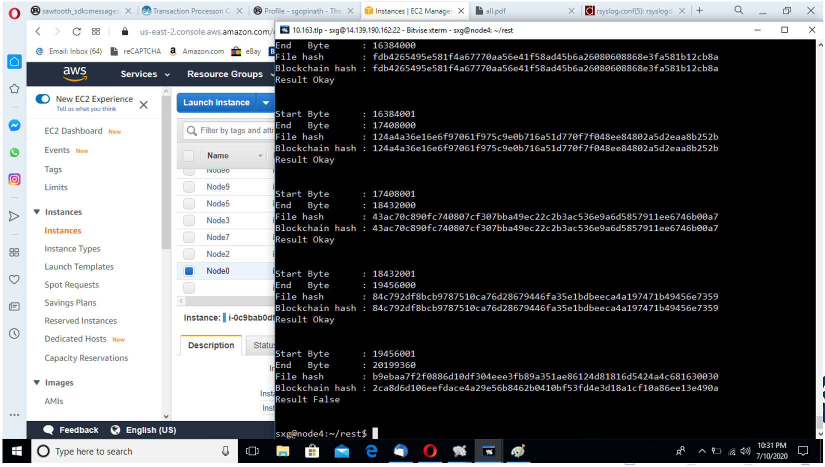
Style-1 Verification-1



```
build_script_main-f9b70f1909b6aba6.d
build_script_main-f9b70f1909b6aba6
build_script_build-022697160e83be2b.
build-script-build
build_script_build-022697160e83be2b
output
invoked.timestamp
stderr
root-output
rest0-5d7bd1864d3ef4a3
Csxg@node0:~/Rust/projects/sawtooth$
sxg@node0:~/Rust/projects/sawtooth$
sxg@node0:~/Rust/projects/sawtooth$
sxg@node0:~/Rust/projects/sawtooth$
sxg@node0:~/Rust/projects/sawtooth$
sxg@node0:~/Rust/projects/sawtooth$
sxg@node0:~/Rust/projects/sawtooth$
sxg@node0:~/Rust/projects/sawtooth$
sxg@node0:~/Rust/projects/sawtooth$
scp -rp /home/sxg/Rust/projects/sawt
scp -rp /home/sxg/Rust/projects/sawt
echo $1
echo $2
sxg@node0:~/Rust/projects/sawtooth$
sxg@node0:~$ ./copynodes no^C
sxg@node0:~$ ^C
sxg@node0:~$ ./copynodes node4 rest
verify.rs
rest_res.rs
block_tools.rs
main.rs
Cargo.toml
node4
rest
sxg@node0:~$
```

```
Blockchain hash : fdb4265495e581f4a67770aa56e41f58ad45b6a26080608868e3fa581b12cb8a
Result Okay


Start Byte      : 16384001
End   Byte      : 17408000
File hash       : 124a4a36e16e6f97061f975c9e0b716a51d770f7f048ee84802a5d2eaa8b252b
Blockchain hash : 124a4a36e16e6f97061f975c9e0b716a51d770f7f048ee84802a5d2eaa8b252b
Result Okay


Start Byte      : 17408001
End   Byte      : 18432000
File hash       : 43ac70c890fc740807cf307bba49ec22c2b3ac536e9a6d5857911ee6746b00a7
Blockchain hash : 43ac70c890fc740807cf307bba49ec22c2b3ac536e9a6d5857911ee6746b00a7
Result Okay


Start Byte      : 18432001
End   Byte      : 19456000
File hash       : 84c792df8bcb9787510ca76d28679446fa35e1bdbeeca4a197471b49456e7359
Blockchain hash : 84c792df8bcb9787510ca76d28679446fa35e1bdbeeca4a197471b49456e7359
Result Okay


Start Byte      : 19456001
End   Byte      : 20199360
File hash       : 2ca8d6d106eefdace4a29e56b8462b0410bf53fd4e3d18a1cf10a86ee13e490a
Blockchain hash : 2ca8d6d106eefdace4a29e56b8462b0410bf53fd4e3d18a1cf10a86ee13e490a
Result Okay


sxg@node4:~/rest$ ls -l testfile
-rw-r--r-- 1 sxg sxg 20199360 Jul 10 16:56 testfile
sxg@node4:~/rest$
```

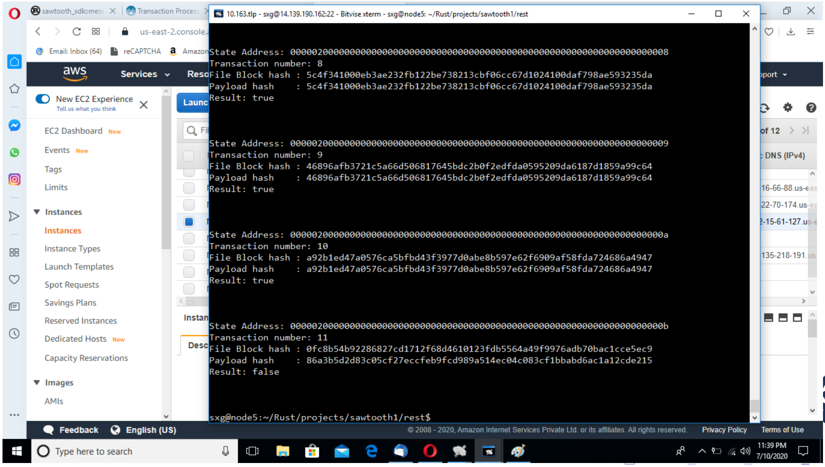
Style-1 Verification-2 (shows False)



10.163.tlp - sxg@14.139.190.162:22 - Bitvise xterm - sxg@node4: ~/rest
Launch Instance
EC2 Dashboard
Instances
Node0

```
End   Byte      : 16384000
File hash       : fdb4265495e581f4a67770aa56e41f58ad45b6a26080608868e3fa581b12cb8a
Blockchain hash : fdb4265495e581f4a67770aa56e41f58ad45b6a26080608868e3fa581b12cb8a
Result Okay


Start Byte      : 16384001
End   Byte      : 17408000
File hash       : 124a4a36e16e6f97061f975c9e0b716a51d770f7f048ee84802a5d2eaa8b252b
Blockchain hash : 124a4a36e16e6f97061f975c9e0b716a51d770f7f048ee84802a5d2eaa8b252b
Result Okay


Start Byte      : 17408001
End   Byte      : 18432000
File hash       : 43ac70c890fc740807cf307bba49ec22c2b3ac536e9a6d5857911ee6746b00a7
Blockchain hash : 43ac70c890fc740807cf307bba49ec22c2b3ac536e9a6d5857911ee6746b00a7
Result Okay


Start Byte      : 18432001
End   Byte      : 19456000
File hash       : 84c792df8bcb9787510ca76d28679446fa35e1bdbeeca4a197471b49456e7359
Blockchain hash : 84c792df8bcb9787510ca76d28679446fa35e1bdbeeca4a197471b49456e7359
Result Okay


Start Byte      : 19456001
End   Byte      : 20199360
File hash       : b9ebaa7f2f0886d10df304eee3fb89a351ae86124d81816d5424a4c681630030
Blockchain hash : 2ca8d6d106eefdace4a29e56b8462b0410bf53fd4e3d18a1cf10a86ee13e490a
Result False


sxg@node4:~/rest$
```

Style-2 Verification-1 (shows False)



```
State Address: 0000020000000000000000000000000000000000000000000000000000000000000008
Transaction number: 8
File Block hash : 5c4f341000eb3ae232fb122be738213cbf06cc67d1024100daf798ae593235da
Payload hash    : 5c4f341000eb3ae232fb122be738213cbf06cc67d1024100daf798ae593235da
Result: true



State Address: 0000020000000000000000000000000000000000000000000000000000000000000009
Transaction number: 9
File Block hash : 46896afb3721c5a66d506817645bdc2b0f2edfda0595209da6187d1859a99c64
Payload hash    : 46896afb3721c5a66d506817645bdc2b0f2edfda0595209da6187d1859a99c64
Result: true



State Address: 000002000000000000000000000000000000000000000000000000000000000000000a
Transaction number: 10
File Block hash : a92b1ed47a0576ca5bfbd43f3977d0abe8b597e62f6909af58fda724686a4947
Payload hash    : a92b1ed47a0576ca5bfbd43f3977d0abe8b597e62f6909af58fda724686a4947
Result: true



State Address: 000002000000000000000000000000000000000000000000000000000000000000000b
Transaction number: 11
File Block hash : 0fc8b54b92286827cd1712f68d4610123fdb5564a49f9976adb70bac1cce5ec9
Payload hash    : 86a3b5d2d83c05cf27eccfeb9fcd989a514ec04c083cf1bbabd6ac1a12cde215
Result: false



sxg@node5:~/Rust/projects/sawtooth1/rest$
```

Conclusion I

- The parameters (stream name, file-block size) may be passed as arguments. Run time parameterization may be implemented in future.
- Persistence System may help to update blocks in near real time.
- The use of Generics in programming may help better code organization.
- A exhaustive testing is required and VA may be done

Conclusion II

- The log *transport* mechanisms to be studied and to go beyond RELP. Other log architectures to be considered.

Challenges I

- Lack of Text material for Blockchain
- Sawtooth Programming documentation may be improved.
- Rust - *intimidating* and lot of concepts to be absorbed.

Acknowledgement I

- Mr.SM Arun, Sawtooth Developer - Hyperledger Community. Shared examples and studied. Leap forward.
- The confidence came after seeing some of his codes.
- Hyperledger Community - Notably SM-Arun.
- Rust - Users Community - Good support from them.

Acknowledgement II

- Lockdown during Covid-19 - It helped *slow learning*, micro-experimentation, *subtle* learning and all with lesser *stress*.
- Encouragement from Dr.S.Revathi, Mrs.Valli, Mrs.R.Akila and Administration.
- Online materials.

Thanks

Thanks!!!