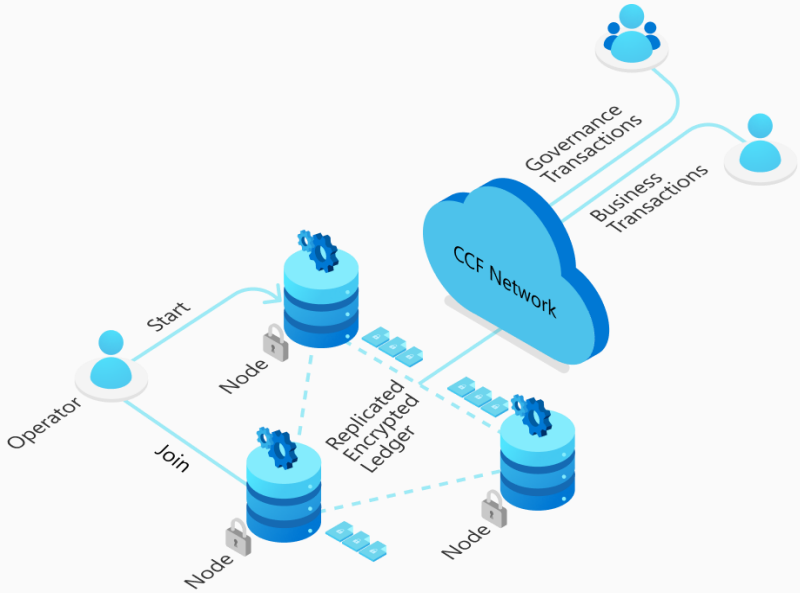
November 8th 2024
IETF 121 - Dublin

Adding SCITT to the Confidential Consortium Framework

H. Birkholz, A. Delignat-Lavaud, C. Fournet, A. Chamayou

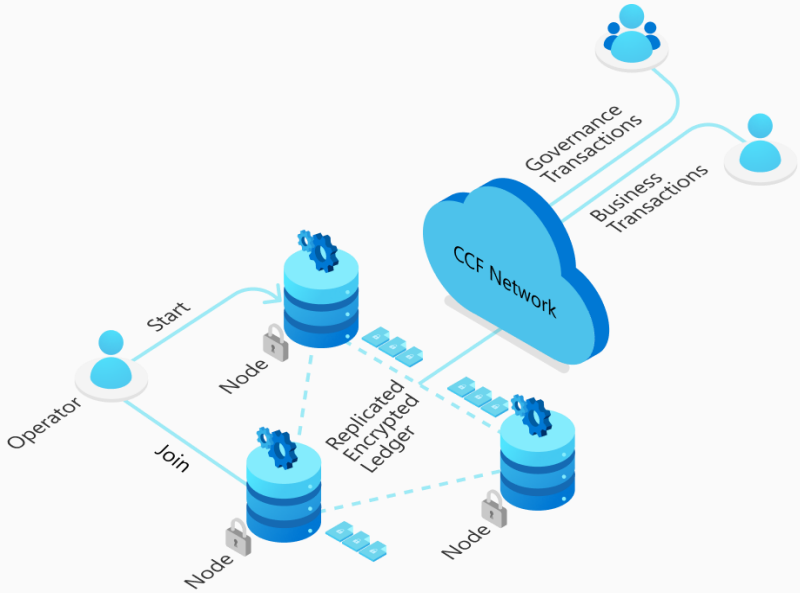
What is the Confidential Consortium Framework?

- Open-source framework built to achieve **decentralized trust** with **centralized execution**
- Combines TEEs and flexible decentralized governance to enable high-performance confidential applications, backed by an immutable ledger



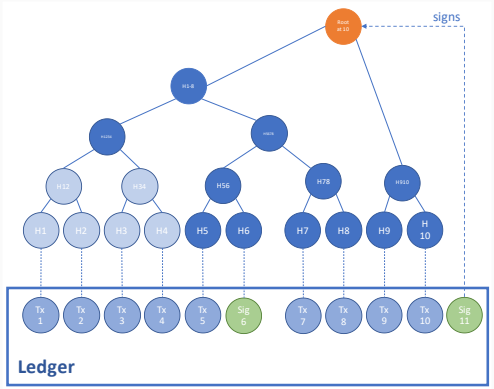
CCF: Key Features

Hardware-based TEEs & Remote Attestation	Confidentiality
Flexible Constitution & Proposal-Vote-Apply model	Customizable Governance
Immutable Append-only Ledger & Receipts	Transparency & Auditability
Crash-Fault Tolerance & Disaster Recovery	Resiliency
Common Programming Languages & Tools	Developer Friendly



COSE Receipt Profile: Important Points

- Single ledger, total order, and Merkle Tree for all transactions
  - Includes governance (code updates, membership updates, etc.)
  - Includes hardware attestations for all nodes
- Receipt claims separate from KV state
  - Flexible coupling to storage
  - Support for confidential state
  - Selective Disclosure



More Information on CCF

- Code repository on GitHub
- Technical documentation
- Research papers
  - Smart Casual Verification of the Confidential Consortium Framework (NSDI '25)
  - Confidential Consortium Framework: Secure Multiparty Applications with Confidentiality, Integrity, and High Availability (VLDB '24)

Azure Confidential Ledger

- Main commercial application of CCF so far
- General purpose ledger service, protected by TEEs
- Write receipts with selectively disclosable claims
- JSON/Azure APIs
- GA in Azure since 2022

Azure Transparency Service

- Deployed on Azure Confidential Compute, powered by CCF
- Will serve 1st party and 3rd party use cases
- Open Source [GitHub]
- User-reproducible builds
- Gated Preview in **April 2025**

Some CDDL

```
ccf-proof-element = [
  left: bool           ; position of the element
  hash: bstr .size 32 ; hash of the proof element (string of HASH_SIZE(32) bytes)
]

ccf-inclusion-proof = bstr .cbor {
  &(leaf: 1) => ccf-leaf
  &(path: 2) => [+ ccf-proof-element]
}
```

Some More CDDL

```
ccf-leaf = [
  internal-transaction-hash: bstr .size 32 ; a string of HASH_SIZE(32) bytes
  internal-evidence: tstr .size (1..1024)  ; a string of at most 1024 bytes
  data-hash: bstr .size 32                 ; a string of HASH_SIZE(32) bytes
]
```

Note: CDDL available from GitHub and automatically checked against implementation

Next Step? Adoption of 1st Stand-Alone COSE Receipt Profile

- A few additional set of eyes for review
- In-room hands call?
- Maybe a list adoption call?
- Profit!