One Network Enterprises™

Can Blockchain Revolutionize the Supply Chain?

Ranjit Notani, One Network CTO, examines Blockchain's powerful potential and a major problem for Blockchain in the supply chain

Can Blockchain Revolutionize the Supply Chain?

Ranjit Notani, One Network CTO, examines Blockchain's powerful potential and a major problem for Blockchain in the supply chain

INTRODUCTION

In this white paper we'll look into whether and how Blockchains can revolutionize the Supply Chain. Our conclusion is that the answer is a **qualified yes**.

To really answer this question, we need to understand what exactly Blockchains are and what they bring to the table. Along the way we will introduce a more nuanced way of looking at Blockchains and introduce some new concepts, ideas and taxonomy that will provide a conceptual framework to better answer this question.

Although this paper is written in the context of Supply Chains we believe the ideas are applicable to a wide range of multi-party problems.

WHAT ARE BLOCKCHAINS?

To understand Blockchains one must first start with the **cryptocurrency Bitcoin**. When Bitcoin burst onto the scene in 2008 it went on to become the first widely adopted cryptocurrency (as a third alternative to the two major classes of extant currencies, viz. Fiat currencies like the US Dollar and Commodity currencies like Gold).

The interesting thing about Bitcoin (as opposed to Fiat currencies) is that it did not require any trusted intermediaries like Banks in general or Central Banks in particular for its functioning.

The most important property of Bitcoin was its disintermediating nature. As Bitcoin took off, people started to wonder whether the principles of Bitcoin could be applied towards other problems. The general architectural principles behind Bitcoin was "extracted" as Blockchains.

The specific nomenclature of "Blockchain" was derived from some of the underlying building blocks of Bitcoin. But it is important to understand that Blockchain is more a set of principles that can be interpreted in many different ways and hence there are dozens of types and variations of Blockchains. More about this later.

So what are the key underlying ideas of Blockchain? Blockchain has been analogized to a **Shared Database or Ledger that anyone can write to and everyone can read and with no central owner**.

Now, we believe that the Shared Database analogy is somewhat misleading. This is because while it is true that anyone can write to the Blockchain, these writes are **mediated by potentially complex rules and validations** specific to *that* Blockchain. For example, in the Bitcoin Blockchain, one of the rules is that a person cannot spend money (i.e. Bitcoins) they don't have. While a typical database has some rules (called constraints) they typically don't support sophisticated business logic.

So, a **Mediated** Shared Database is a better analogy. Now, a Database also has the connotation of the "thing" that holds the data rather than the data itself. The data itself is often referred to as **State**. For the purposes of this discussion it

The interesting thing about Bitcoin is that it did not require any trusted intermediaries like Banks in general or Central Banks in particular for its functioning.

is the underlying shared state and its characteristics that will better inform the discussion. So, based on this the key characteristics of Blockchains are:

- Mediated Shared State (for write purposes)
- There is no central owner of the mediated shared state.
- "Anyone" can "write" to the Mediated Shared State
- "Anyone" can "read" the entire Shared State (no mediation required)
- Delayed Single Version of the Truth.
  - The shared state has a probabilistic single version of the truth. (The likelihood of a transaction being "reversed" decreases exponentially with time).

The lack of a central owner implies that both the shared state (data) and the mediation (i.e. business logic) themselves must be distributed.

CAN MEDIATED SHARED STATE REVOLUTIONIZE THE SUPPLY CHAIN?

The answer to this question is an **unqualified yes**. Currently Supply Chains and related business processes are highly fragmented across thousands of trading partners. Business processes can be dramatically improved by moving from highly fragmented state to mediated shared state.

The supply chain can move to **single version of the truth** which in turn allows **near-real-time sense-and-respond** which in turn leads to **dramatically better business metrics** like lower inventory levels, higher service levels, lower total landed costs etc.

So, what's the catch? The catch is that Blockchain is not just about (non-owned) mediated shared state, but also comes along with another (for many use cases) less welcome characteristic, viz.

- "Anyone" can "read" the entire shared state (no mediation required)

Most supply chain participants do not want their information shared with their competitors, much less "anyone".

Another way of stating this is that there are only a relatively minor subset of supply chain problems where this is an acceptable characteristic.

We'll refer to this as the **"everyone-can-read-everything" problem**.

COMMUNITY BLOCKCHAINS

Over time as people tried to solve various multi-party business problems with Blockchains, they kept bumping up against the "everyone-can-read-everything" problem. And so the idea of the **Permissioned Blockchain** was born. These are also often referred to as **Private Blockchains**.

The idea behind Permissioned Blockchains is that some central authority would vet "who" could participate in the Blockchain.

There is a raging debate in the tech community as to whether these are truly Blockchains or merely fancy distributed databases. One difference being in how these distributed systems reach consensus. But that debate is not particularly pertinent to the *benefits* of this approach.

So, what are Permissioned Blockchains?

In the original Blockchain (now called **Non-Permissioned** or **Public Blockchains** to differentiate) "anyone" could initiate a mediated write to the shared state and "anyone" could read the entire shared state.

The idea behind Permissioned Blockchains is that some central authority would vet "who" could participate in the Blockchain. So, instead of "anyone" being able to read or write to the Blockchain, now only vetted community members could. For this reason, we prefer the term **Community Blockchains**.

So the characteristics of Community Blockchains are:

- Mediated Shared State
- **There is a central owner that determines who can participate in the Community Blockchain.**
- "Any **Community Member**" can "write" to the mediated shared state
- No central owner of the mediated shared state
- "Any **Community Member**" can "read" the shared state (without mediation)
- *Delayed* Single Version of the Truth.

So, do Community Blockchains solve the problems of Public Blockchains? Unfortunately, for the most part, no. Once again the sticking point is this characteristic:

- "Any Community Member" can "read" the shared state (without mediation)

The problem is how does one come up with the "community". For example, if we take the example of Global Trade, maybe the community would be made up of Buyers, Vendors, Ocean Carriers, Land Carriers, 3PL's, 4PL's, Ports, Customs, Freight Forwarders etc.

Even in a community like this, Buyers (for example) would not want (say) other Buyers to see information about their (say) Orders.

MICRO-COMMUNITY BLOCKCHAINS TO THE RESCUE?

One seeming solution to the problem of "every-member-can-read-everything" is the idea of **Micro-Community** Blockchains. Here we have micro-communities where the membership is so small that the fact that every-member can read everything is no longer a problem. This is the approach taken by projects such as *HyperLedger*.

There are two types of Micro-communities that seem plausible at first glance.

1. *Pairwise* Micro-community Blockchains

In this approach **every pair of trading partners** has a pairwise micro-community. While an advance over point-to-point messaging such as EDI, this a massive balkanization of the shared supply chain state and the advantages that accrue from that. Simple things like Single-Version-of-the Truth Multi-party business transactions with more than two parties (*workflow arity* > *2*) are not feasible with this approach.

2. *Per-Business-Transaction* Micro-community Blockchains

In this approach every individual Business Transaction (for example, each individual Order) forms a micro-community with the participants of only *that particular* Order. While this approach may work for very simple applications, it has some major problems.

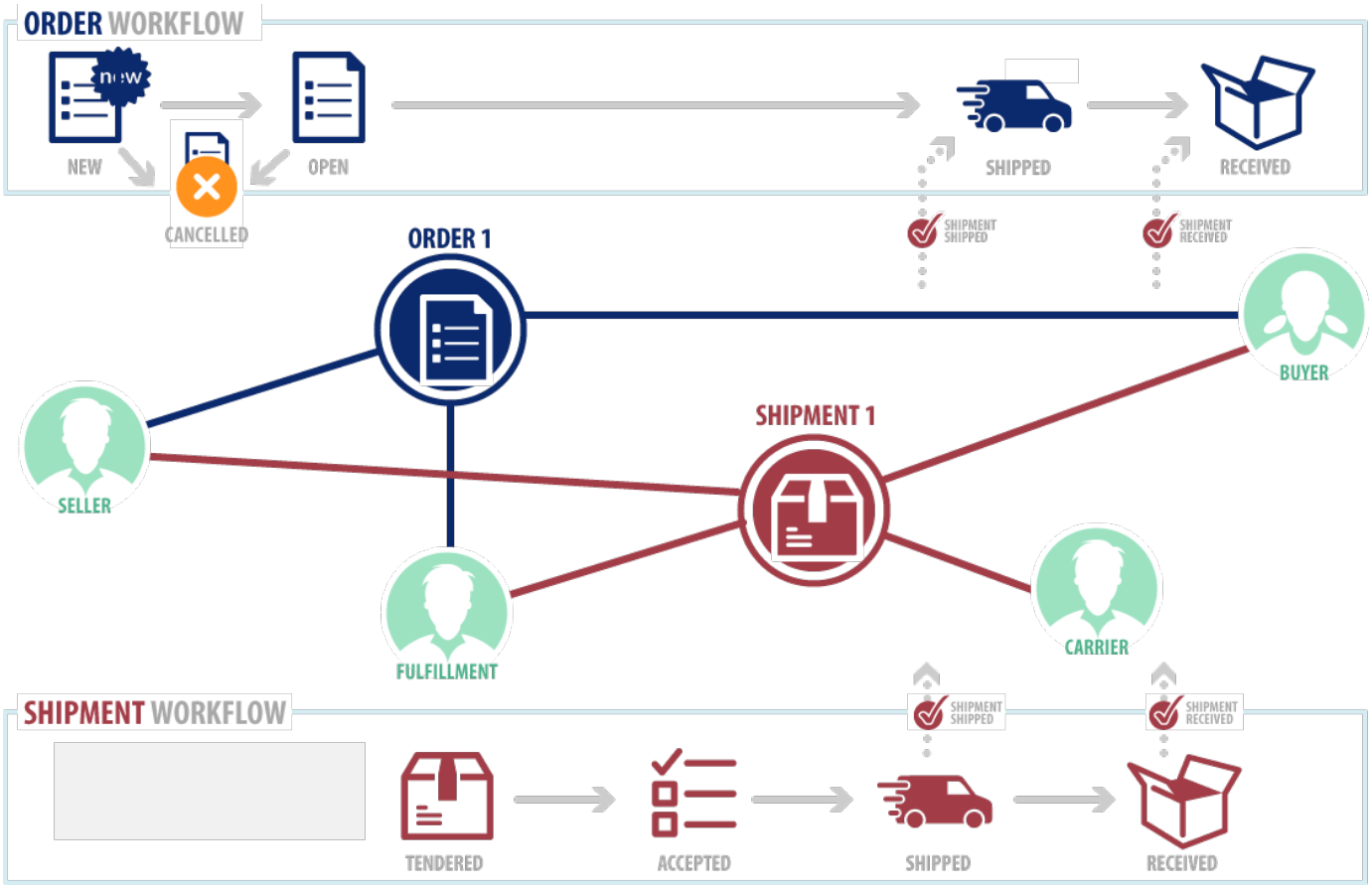
a) It is predicated on the mediated writes having the scope of a single business transaction. This is rare. For example mediated writes to a business transaction often require access to master data information as well as information on related transactions (which may be between other parties).

b) Different Micro-communities are inconsistent with respect to each other. In other words, there is no Single Version of the Truth. Not even delayed Single Version of the Truth of a single Blockchain.

c) Even per-Transaction “everyone-can-see-everything” visibility is problematic in many cases.

d) Cross-micro-community permissions are a problem.

Let’s consider an example to illustrate these problems. In the figure below we see a simple **Drop Ship Scenario**. In a Drop Ship scenario a **Buyer** places an *Order* on a **Seller** who in turn shares the Order with a **Fulfiller** who in turn creates a *Shipment* and tenders it to a **Carrier** for pickup and delivery.



The problem for Micro-Community Blockchains is that Supply Chain transactions are not compartmentalized. Instead they form a graph-like structure.

In this example, if Order1 is one Micro-community Blockchain and Shipment1 is another Micro-community Blockchain, we have many problems:

1. Let's posit that only the Buyer and Seller should see the *price* on ORDER1, whereas Fulfiller has no reason to see the price. This cannot be achieved with a single ORDER1 Blockchain.

    a. One option is to split Order into two slightly varying Order replicas one with the price and one without the price. Now you have introduced even more synchronization problems.

2. The system of record for the SHIPPED and RECEIVED states is on SHIPMENT1. The Order must simply reflect these states. However this raises many issues.

    a. At best there is delay between the Shipment States being propagated to the Order.

    b. Who is responsible for this synchronization? Is it possible to guarantee that there is always a party that has sufficient permissions to perform this synchronization?

3. Let's assume the Shipment is SHIPPED, but this state has not been synchronized with the Order yet (which is still in the OPEN state) and the Buyer CANCEL's the Order. Now we have an inconsistent state. The lack of strong consistency across micro-communities can cause massive inconsistencies.

4. The Buyer, Seller and Fulfiller need item level details on the Shipment. The Carrier must have an *aggregated* commodity-code based view of the Shipment. Because the information is replicated to all parties on the transaction it is not possible to achieve this.

Could some of these problems be solved by putting ORDER1 and SHIPMENT1 into a single, larger micro-community? Even this is problematic. Suppose that ORDER1 had two line items, one fulfilled by Fulfiller1 and another by Fulfiller2. Fulfiller2 tenders to a different Carrier and there is a different Shipment (say, SHIPMENT2). So, now we would have to put SHIPMENT2 also into the same micro-community. But the parties that can see SHIPMENT1 are different than the parties that can see SHIPMENT2.

In general, the problem is that Supply Chain transactions are not compartmentalized. Instead they form a *graph-like* structure. Where do the boundaries of this graph lie? Further, the wider one makes the micro-community, the more severe the everyone-can-see-everything problem becomes. A catch-22!

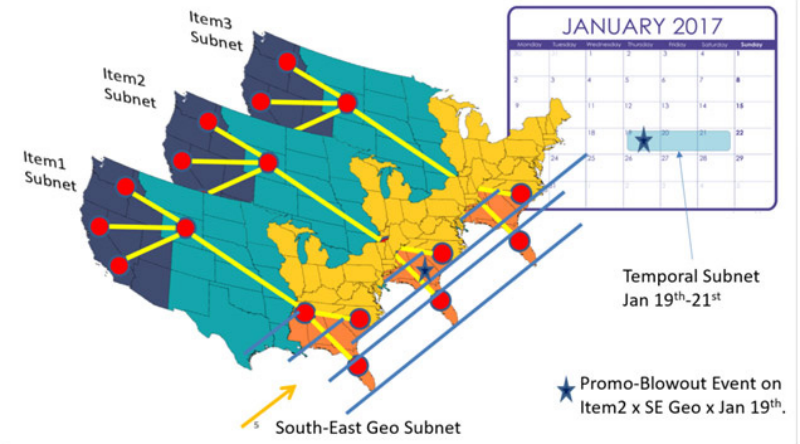
Furthermore, in today's real-time intelligent supply chains external writes by a party are not the only kind of writes. Intelligent agents constantly monitor the real-time shared state and are triggered by certain events. These agents work by reading a portion of the shared state (often called a *Subnet*), determine how to react to these events (typically using optimization algorithms, linear programming, integer programming, heuristic optimization, stochastic algorithms, machine learning algorithms etc.) and then writing back to

Intelligent agents are at the heart of the revolutionizing potential of mediated shared state and are essentially impossible with the Micro-Community Blockchain approach.

the shared state. Each Intelligent Agent works typically over a different (often orthogonal) *subnet*.

Examples of intelligent agents include *Continuous Forecasting*, *Last Minute Allocation*, *Prioritized Expedite*, *Supply Disruption*, *Re-promising* etc. Intelligent Agents.

The figure below shows a simplified view of how different agents may (orthogonally) divide up the overall shared state. This results in the need for a *deep, matrixed* permissions model rather than a *shallow, compartmentalized* permissions model of Micro-community permissioned Blockchains.



These Intelligent agents are at the heart of the revolutionizing potential of mediated shared state and are essentially impossible with the Micro-Community Blockchain approach.

BACKCHAINS AND BACKCHAIN INITIATORS

So, because of the "every-member-can-read-everything" problem and Micro-community Blockchains not really solving the problem, we are seemingly stuck at a small subset of the addressable supply chain problem space.

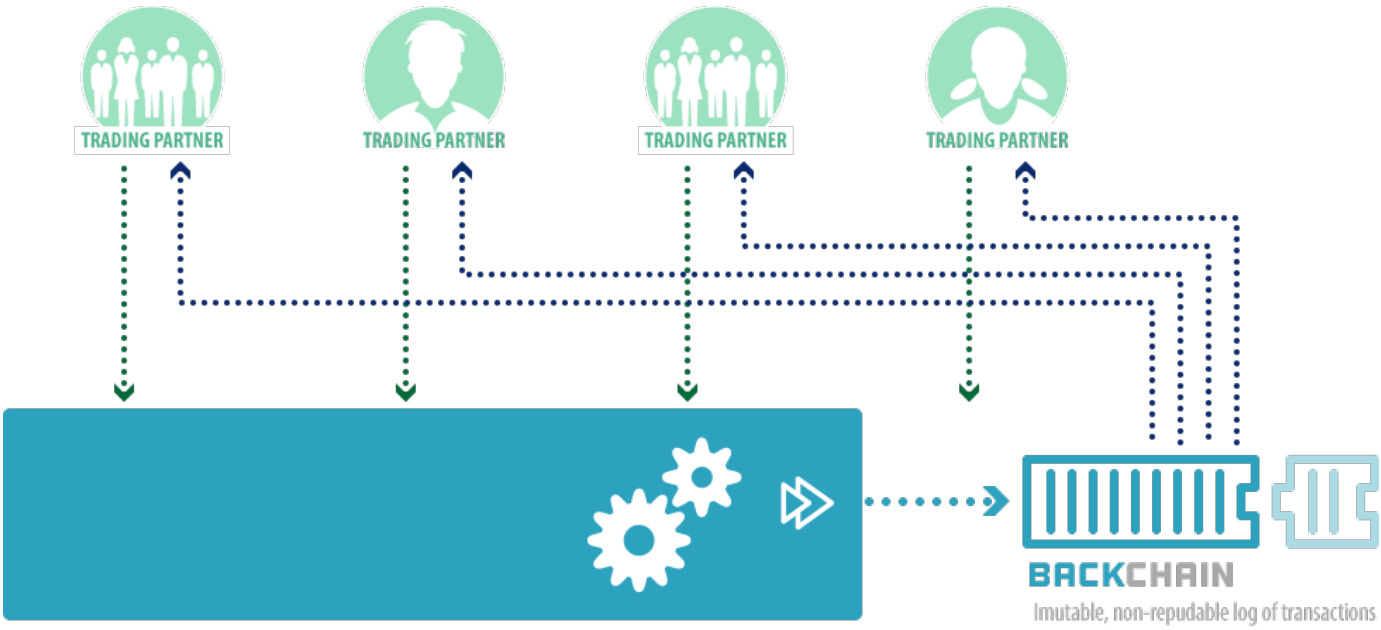
However, there is light at the end of the tunnel. In the move from Public to Community Blockchains we sacrificed some of the decentralization of public Blockchains for a benefit. In this particular case we introduced centralization in the who-can-be-member-of-the-community decision (downside) for restricting the read and write abilities to a vetted community (benefit).

With the **Backchain Pattern** (which we will explore in some detail) we sacrifice one additional element of decentralization for a massive increase in benefit.

In the Backchain Pattern, the writes are mediated by a centralized entity, called a **Backchain Initiator** (downside due to additional centralization) for the ability to tackle almost all real-time multi-party supply chain problems (benefit).

In the Backchain Pattern all parties route their writes through the Backchain Initiator. The Backchain Initiator is the only entity with (unencrypted) read access to the entire mediated shared state. The Backchain Initiator also maintains a *Transient* Single Version of the Truth of the shared state.

This allows the Backchain Initiator to execute arbitrarily complex mediated writes which may access the full shared



state. The Backchain Initiator can also execute Intelligent Agents as it can retrieve any required *subnet* due to its unfettered read access to the entire mediated shared state. The Backchain Initiator can also maintain a single version of the truth unlike the Micro-community approach.

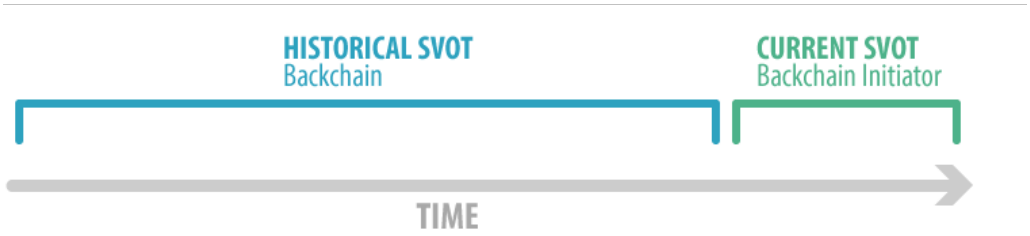
This essentially expands the range of addressable problems to the entire multi-party supply chain problem space. Also, it results in the business benefits of moving to Mediated Shared state in terms of dramatically improved supply chain metrics

Once a transaction is mediated by the Backchain Initiator, it is **sliced by trading partner** and then hashed (or encrypted) and placed onto a Blockchain called the **Backchain** by the Backchain Initiator. This way every party still has an immutable, non-repudiable record of all transactions.

This has the very important property of requiring the parties to only place ***Transient Trust*** in the Backchain Initiator rather than ***Perpetual Trust***. The figure below illustrates the notion of Single Version of the Truth (SVOT) in this scheme.

Note that because the information on the Backchain is encrypted or cryptographically hashed, the Backchain may itself be a Public or Community Blockchain. We also refer to this Backchain as a Content Backchain to distinguish from the Hold Backchain which we will discuss next.

The **Backchain Initiator figure** above shows the various components of the Backchain Pattern.



One of the major purposes of Blockchain is to remove the possibility of compromised intermediaries.We can achieve this with the "Mediator Hold" .

THE HOLD BACKCHAIN

Because the Backchain Initiator places transactions onto a decentralized Blockchain (Backchain) we reduced the need for trust on a centralized entity from Perpetual to Transient.

If the Backchain Initiator is compromised at some point in times or goes out of business, this has no impact on the validity of *historical transactions and their non-repudiability*. This is a massive reduction in reliance on a centralized entity.

But can we go further and even have mechanisms to **mitigate the issue of transient trust?**

Because the Backchain is initiated by a centralized Backchain Initiator, there is the possibility of a compromised initiator. One of the major purposes of Blockchain is to remove the possibility of compromised intermediaries.

The basic issue is write validation. The way this is normally done by *centralized* shared state mediators is to make callbacks to the impacted parties (trading partners) to validate the final state of the transaction just before it is about to be committed to the shared state. However for the purposes of dealing with a compromised mediator this doesn't work, as the mediator (the Backchain Initiator in this case) could just ignore the results of the validation callback that was sent to the impacted trading partner.

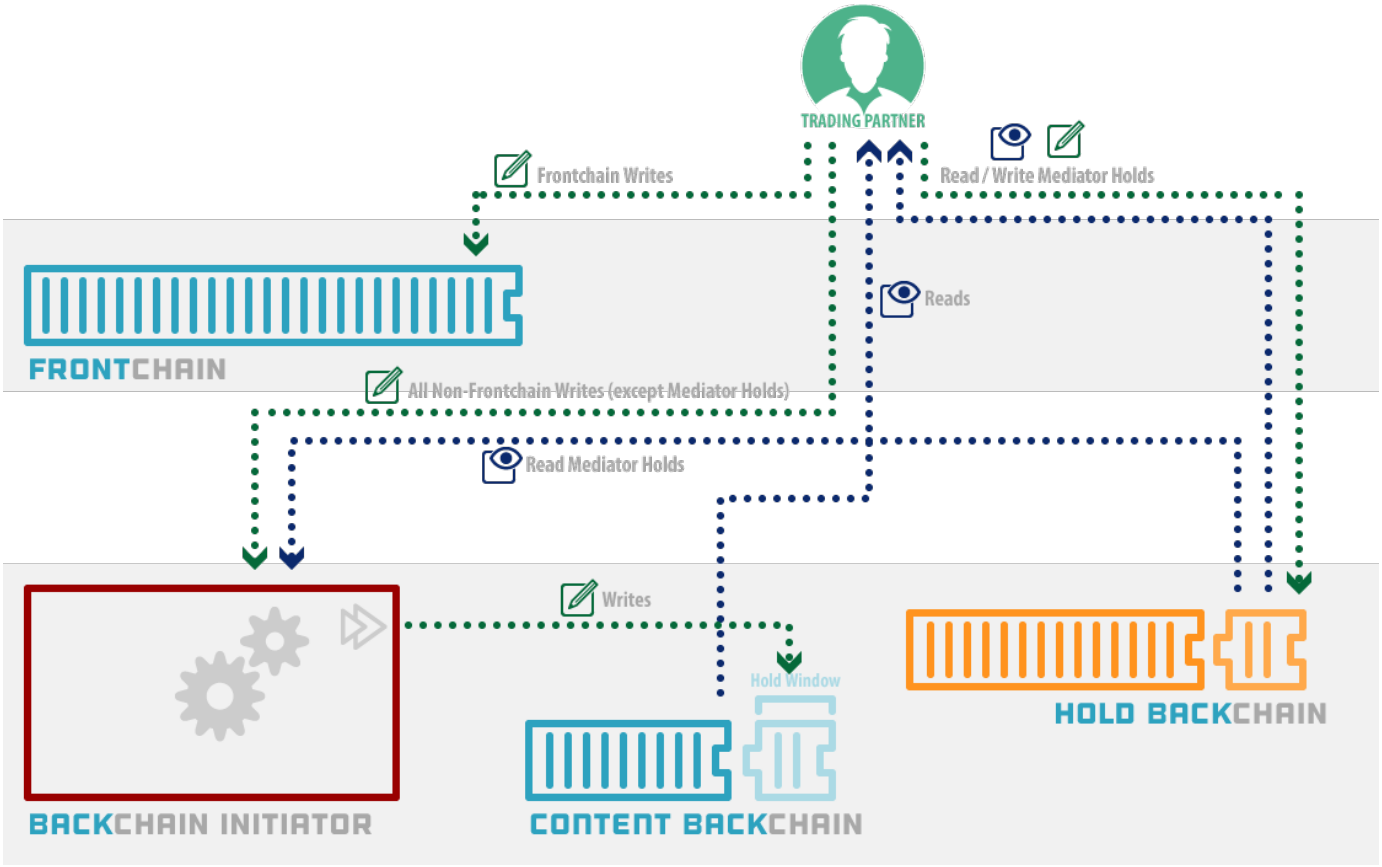
Instead, another mechanism is used which relies on the concept of *Holds*. Most Business Transactions (such as Orders, Shipments, Movements, and Invoices etc.) have the concept of a Hold. The idea of Holds is that they "stop" or "hold" or "limit" the Business Transaction in some way. There are various types of Holds such as *Credit Holds*, *Inventory Holds*, *Invoice Dispute Holds*, etc.

A major characteristic of Holds is that they are considered a part of the Business Process itself (i.e. they are not a technical concept like (say) "rolling back" a transaction). The second major aspect of Holds is that they are often first-class Business Transactions in their own right and are typically asynchronous relative to the Business Transaction they are "holding".

The Backchain Pattern introduces a new category of Holds called the **Mediator Hold**. We will call "normal" Holds such as Credit Holds, **Normal Holds**. Normal Holds are applied by trading partners to Business Transactions based on activities of trading partners (including themselves). They essentially treat the Mediator as transparent or absent.

In contrast the Mediator Hold is a Hold where a Trading Partner has lost trust or has some other issue with the Mediator (Backchain Initiator) itself. In order to ensure that the Backchain Initiator cannot tamper with Mediator Holds, the Mediator Holds are placed by trading partners directly onto another Blockchain called the **Hold Backchain**. Normal Holds go to the Backchain Initiator like all other writes.

The Hold Backchain will have a **Hold Window** built into its Blockchain protocol. The Hold Window is a duration after a transaction is written to the (Content) Backchain during which the Trading Partners may place a Mediator Hold on a Business



Transaction. For example, if the Hold Window is one hour, trading partners will have one hour to raise a Mediator Hold before the transaction is considered no longer subject to this type of Hold.

Like all other Holds, the Mediator Hold is built into the business process itself and all parties will react based on this business process. The Backchain Initiator (which presumably does not consider itself compromised) will itself read the Hold Backchain and *potentially* react to a Mediator Hold placed by a trading partner.

The critical element here is that the centralized mediator has no ability to influence the placement of Mediator Holds. And so if transactions produced via the Backchain Initiator are compromised (i.e. the transient trust was violated), these transactions may be repudiated (within the Hold Window).

In a Community Blockchain, raising a Mediator Hold should be a very rare event. The only foolproof workaround is to stop using the Backchain Initiator and move to a “reduced mode” of operation of using only Front Chains (see below for the definition of this). For example, companies may fall back to a reduced-mode Micro-community Front chain or traditional EDI. In practice, since the Backchain Initiator will most likely be an accountable entity of some sort, the problem will be escalated to some defined governance mechanism.

The figure above shows the interaction of a Trading Partner with a Front Chain, a Backchain Initiator, the Content Backchain and the Hold Backchain.

The hallmark of cryptographic hashes is that it is very easy "to go" from the content to the cryptographic hash but impossible to go in the opposite direction.

ENCRYPTED VS CRYPTOGRAPHICALLY-HASHED CONTENT BACKCHAINS

The Content Backchain which contains transaction logs split by trading partner (deep, matrixed) read permissions do not store the entire (split) transaction in "clear" form. Instead the content (the split transaction) is stored in either Encrypted or Hashed Form.

Note: We use the term "Deep, Matrixed Permissions" to distinguish from the All-members-can-see-everything Permissions characteristic of so-called Permissioned Blockchains (which, as mentioned earlier, we prefer to call Community Blockchains to avoid this confusion).

The structure of a Log Message in the Content Backchain looks like:

1. Metadata (unencrypted)
2. Full Content (encrypted or cryptographically-hashed)

There are pros and cons to the Encryption vs Hashing of the full content which we will explore in this section.

With the encryption approach the entire content (other than some metadata) is encrypted. The advantage of this is that a trading partner can use their decryption key to recreate the entire (subset) of the supply chain state that they have (deep) read access to.

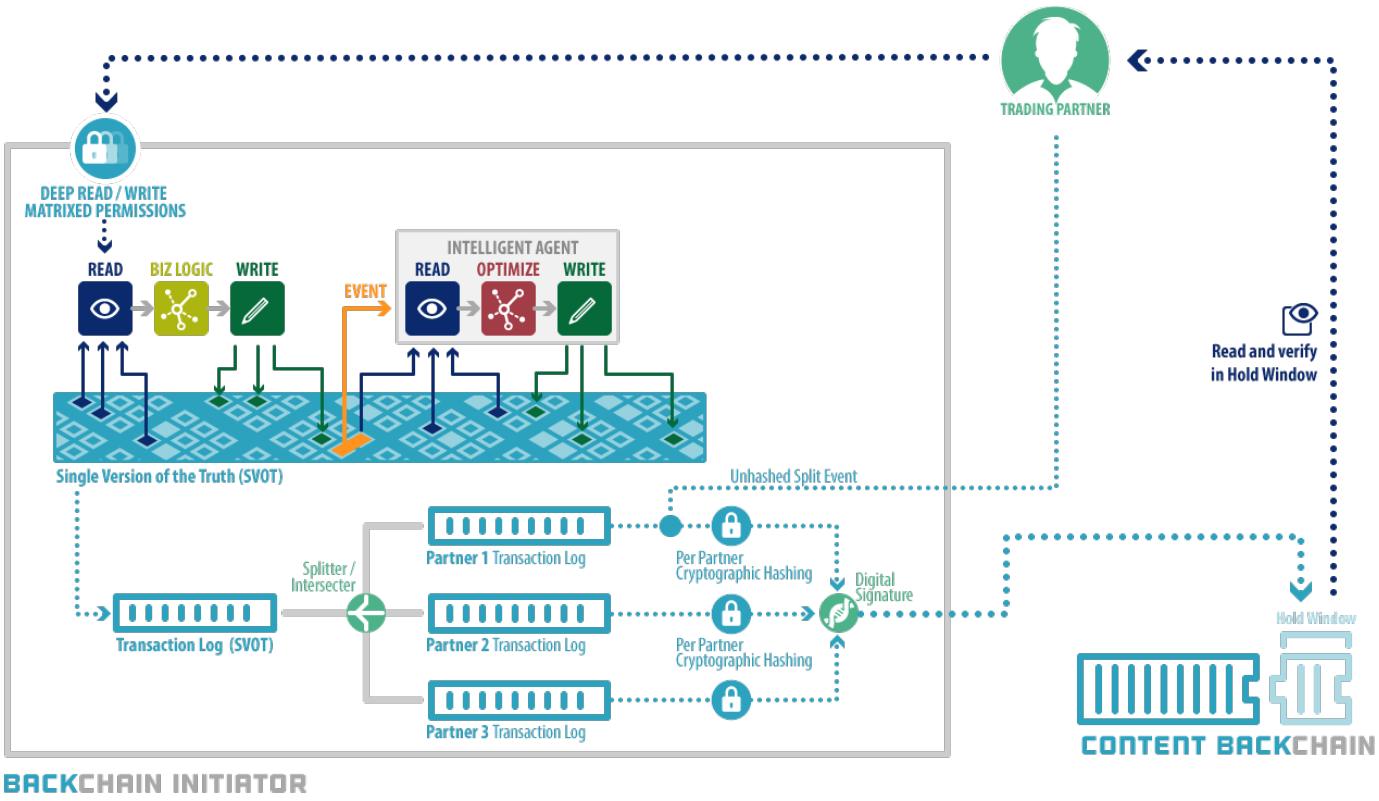
The downside of this approach is that the entire shared supply chain state is replicated to all members (this is a basic property of Blockchains) giving a trading partner essentially an eternity to "crack" other trading partners' encryption.

For this reason we favor a **Cryptographically Hashed** Content Backchain. A cryptographic hash can be thought of as essentially a "digital fingerprint" of a piece of content. Every distinct piece of content will have a unique* cryptographic hash or fingerprint.

The hallmark of cryptographic hashes is that it is very easy "to go" from the content to the cryptographic hash but impossible to go in the opposite direction. So why is this helpful? Firstly, other trading partners will never be able to even attempt to decrypt your content. Secondly if you have the content you can easily "prove" that that transaction (content) had occurred by presenting it and showing that the cryptographic hash of your presented content matches the cryptographic hash on the Content Backchain.

So, how does a trading partner get a hold of the content (transaction) that was cryptographically hashed and placed onto the Content Backchain? Essentially the Backchain Initiator passes the sliced, encrypted content to the trading partner using traditional (non-Blockchain) means.

All trading partners continuously monitor the Content Backchain for any transactions relevant to them. If they "see" one for "themselves" they immediately verify the content (which they should have directly received from the Backchain Initiator) with the hash on the Backchain. If the hashes do not match they should raise an **Invalid-Hash Mediator Hold** on the Hold Backchain. If they never received the original



content they should raise a **No-Content-Received Mediator Hold** on the Content Backchain. The former should be a rare situation. The latter can happen more frequently, if for example there is a connectivity issue between the Backchain Initiator and the trading partner.

Again, like with any Mediator Hold the most drastic resolution is to move to a reduced mode of operation utilizing only Front Chains. In practice, especially in a Community setting, these will be escalated to community governance mechanisms.

The **Backchain Initiator figure** above shows the internal structure of the Backchain Initiator.

DEALING WITH DISSIMILAR VIEWS RESULTING FROM THE SPLIT PROCESS

A key part of the Backchain initiator is to split *n-way multiparty* transactions into *n single-party* transactions prior to hashing and/or encryption. This *multi-party-transaction-split* is performed using each party's corresponding *read* permission on the multi-party transaction.

Because different parties (in general) have different deep read permissions on a particular multi-party transaction they will in general have different permissioned "views" of the same underlying multi-party transaction.

For example in the Drop-Ship example earlier, the Fulfiller's read permission would not allow them to see the price on the ORDER, whereas the Buyer and Seller could. Similarly, the Seller and Fulfiller may have access to shared information on the same ORDER that the Buyer is not privy to.

This leads to a potential problem with non-repudiation of multi-party-transactions (a key purpose of the Backchain).

Each party can only "prove" *their* view of the underlying multi-party transaction. By presenting their view and showing that it hashes to the hash on the Backchain. The first problem is that *their* view may have information they consider private, but the only way to "prove" it is to present the entire view. This would present security issues. The second problem is because *their* view in general will be different from other

If a particular party needs to prove the existence of shared transaction between themselves and another party they will present the corresponding intersection.

party's views, proving that the *shared* multi-party transaction exists in the Backchain transaction log will (in general) not be possible.

This is solved in the Backchain Initiator pattern by computing *all possible intersections* of the single-party transaction views and hashing (or encrypting) these intersections as well. This algorithm is referred to as the *multi-party-intersection* algorithm.

For example, consider a 4-party transaction. Suppose the four parties are A, B, C and D and the views of a particular transaction T based on their read-permissions are VA, VB, VC and VD.

Then we have the following intersections: (Note: the "*" operator is used to represent the intersection operator):

2-way intersections:
VA * VB, VA *VC, VA * VD, VB * VC, VB * VD and VC * VD

3-way intersections:
VA * VB * VC, VA * VB * VD, VA * VC * VD and VB * VC * VD

4-way intersections:
VA * VB * VC * VD

Each party is sent not only their View of the multi-party transaction, but also all of the intersections they participate in. Note that it is possible that one or more intersections are empty.

For example, party A, will be sent the following views in addition to VA:
VA*VB, VA*VC, VA*VD, VA*VB*VC, VA*VB*VD, VA*VC*VD and VA*VB*VC*VD.

Finally the cryptographic hash or encryption of *all* the intersections is placed on the Backchain as well. In the above 4-party example, there will be 11 hashes (or encrypted values) placed on the Backchain.

Now if a particular party needs to prove the existence of shared transaction between themselves and another party they will present the corresponding intersection. For example, if party A needs to prove to party B the existence of the shared portion of the multi-party transaction they will present A*B for non-repudiation purposes.

Similarly if they want to prove to B and C the mutual shared portion of the multi-party transaction they will present A*B*C. This way all possible subsets can be proven as necessary.

In general in an *n-way* multi-party transaction there will be nC2 + nC3 + nC4 + …. + nCn intersection combinations (where C is the combination operator). The Backchain Initiator may choose to only support pairwise non-repudiability in which case only nC2 intersections will be computed.

Now, how are these intersections computed? General purpose algorithms like Longest Common Subsequence can be utilized. If the multi-party transaction is in relational form then a relational intersection can be performed. Additionally,

For all the potential of Blockchain, there is still the necessity of having a non-Blockchain mediator (the Backchain Initiator) to cover the majority of the revolutionary use-cases of single-version-of-the-truth mediated shared state.

the Backchain Initiator can create specialized intersection algorithms based on the format of the multi-party changeset. The nature of the particular intersection algorithm has no impact on how overall multi-party-intersection algorithm operates.

FRONTCHAINS AND BACKCHAINS

Based on the preceding discussion, we can now see how Blockchains can revolutionize Supply Chains. It helps to think in terms of an orthogonal classification of Blockchains, viz. FrontChains and BackChains.

FrontChains are utilized for those supply chain applications where it is ok for “everyone” (or “every member”) to have access to the entire supply chain shared data state. An example of a FrontChain would be a Singleton FrontChain for the provenance of high value items (say Diamonds) in a Supply Chain.

The important thing to note is that we don’t say that the entire Supply Chain runs on FrontChains, but rather that specific Supply Chain Applications run on the FrontChain. FrontChains can also be used for “reduced-mode” operations. For example, if trust in the Backchain Initiator is lost, trading partners could fall back to a Pairwise Micro-community FrontChain.

For supply chain applications where this is not desirable (which is most), the Backchain Pattern is used. This comprises 3 logical entities.

1) A Backchain initiator (not a Blockchain)
2) The Content Backchain
3) The Hold Backchain

Often #2 and #3 will be combined into a single Blockchain (just called “the Backchain”).

CONCLUSION

At the start of this discussion we assessed that the potential for Blockchains to revolutionize Supply Chains was a qualified yes. We can now see what that qualification is. The major qualification is the necessity of having a non-Blockchain mediator (i.e. the Backchain Initiator) to cover the majority of the revolutionary use-cases of single-version-of-the-truth mediated shared state.

However, through the use of the Content Backchain we reduce the trust needed from perpetual trust to transient trust. And with the Hold Backchain we further protect against violations of transient trust.

By combining Blockchain with a non-Blockchain mediator (the Backchain Initiator) to cover the majority of the revolutionary use-cases of single-version-of-the-truth mediated shared state, we can realize Blockchain's full potential in managing complex, multi-party supply networks.

For more information, contact One Network at:

Tel: +1-866-302-1936 (toll free)
Email: inquiries@onenetwork.com

One Network Enterprises™

Corporate Headquarters US
One Network Enterprises
4055 Valley View Ln, Suite 1000
Dallas, TX 75244
Tel: +1-866-302-1936 (toll free)
+1 972-385-8630
Email: inquiries@onenetwork.com
Web: www.onenetwork.com

International Headquarters
One Network Enterprises (Europe)
PO Box 59383
London NW8 1HH, UK
Tel: +44 (0) 203-28-66-901
Email: europe@onenetwork.com
Web: www.onenetwork.com