University of Groningen

Hyperledger Fabric

Soelman, Mark; Andrikopoulos, Vasilios; Perez, Jorge A. ; Theodosiadis, Vasileios ; Goense, Karel; Rutjes, Arne

Published in:
2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)

DOI:
[10.1109/DAPPS49028.2020.00019](https://doi.org/10.1109/DAPPS49028.2020.00019)

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2020

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Soelman, M., Andrikopoulos, V., Perez, J. A., Theodosiadis, V., Goense, K., & Rutjes, A. (2020). Hyperledger Fabric: Evaluating Endorsement Policy Strategies in Supply Chains. In *2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)* (pp. 145-152). IEEE. <https://doi.org/10.1109/DAPPS49028.2020.00019>

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

Hyperledger Fabric: Evaluating Endorsement Policy Strategies in Supply Chains

Mark Soelman, Vasilios Andrikopoulos, Jorge A. Pérez
Computer Science Department
University of Groningen
Groningen, The Netherlands
m.soelman@student.rug.nl, {v.andrikopoulos, j.a.perez}@rug.nl

Vasileios Theodosiadis, Karel Goense, Arne Rutjes
IBM Client Innovation Center Benelux
Groningen, The Netherlands
{vasileios.theodosiadis, karel.goense}-cic.netherlands@ibm.com
arne.rutjesISC@nl.ibm.com

Abstract—Hyperledger Fabric is a permissioned blockchain solution, in which network participation is controlled by predefined rules. This makes it an attractive platform for enterprise settings. In Fabric, endorsement policies are used to specify the peers that must confirm a transaction before it can be considered as valid and appended to the ledger. This work examines various implications of Fabric’s endorsement policy component, for which different endorsement policy strategies (and subsequent trade-offs) are evaluated by modeling two real-world supply chain case studies. This work discusses how vulnerable endorsement policies can lead to admitting inauthentic data on the ledger. To address this issue, several approaches are proposed to balance integrity with limited disclosure of confidential information, with or without hosting a network peer directly. Furthermore, the concept of multi-tenancy in blockchain networks is introduced as a way of reducing the technological barrier in technology adoption.

Index Terms—Permissioned blockchain, Endorsement policy, Hyperledger Fabric, Supply chain, Notary nodes, Multi-tenancy

I. INTRODUCTION

Blockchain networks have been at the epicenter of research efforts during the last years. This trend will intensify in the near future, as many organizations are extending or experimenting with blockchain networks with their stakeholders. The technology aims at disrupting different domains from *digital identity*¹ to *notary public*². Blockchain networks are especially relevant in *supply chain* use cases; this significance has been recognized in academia [1]–[3] and industry [4]–[6]. Supply chain is of particular interest in blockchain networks because parties with conflicting interests often exchange valuable and confidential information. In this setting, blockchain adoption increases transparency among network participants, while reducing information asymmetry and operational costs [1]. The literature also highlights how blockchain contributes to save transaction verifiability costs, by bypassing intermediaries [7].

Besides the widely known *permissionless blockchains* in which anyone can join the network (such as Bitcoin [8] and Ethereum [9]), *permissioned blockchains* are becoming increasingly popular [10]. In a permissioned blockchain, multiple organizations work together on shared distributed ledgers

whose permissions can be configured per organization and its members. This setting is attractive to collaborating businesses and organizations, as the governance for who can join the network can be technically enforced. Moreover, the knowledge about participants reduces the need for heavy consensus mechanisms, as offline dispute resolution becomes possible. In this setting, several industrial blockchain applications have been recently launched to improve the pipelines of industrial stakeholders in their collaborations. For example, TradeLens optimizes business transactions and inventory management, leading to a reduction in corporate expenses [6]. Business partners formed a consortium to enhance trade finance letters of credit [5]. Another use case is IBM Food Trust, which contributes to an increased transparency, waste reduction, and better insights of a product’s ingredients to its consumers [4].

Popular frameworks for building permissioned blockchains include Tendermint³, Quorum⁴, Corda [11], and Hyperledger Fabric [12]. This paper focuses on Hyperledger Fabric (Fabric in the following), an open source project managed by the Linux Foundation, which is reported to be the most widely used framework for enterprise consortia⁵. Fabric introduces its own architecture, in which many components are pluggable and configurable, including the consensus protocol and the code to execute and validate transactions. In this architecture, a proposed transaction or change to the ledger is only executed by a subset of network peers, before the transaction is added to the shared ledger. Fabric’s modularity helps the development of permissioned blockchains, tailored to specific requirements of a business in a wide variety of fields and use cases [4], [6].

One of Fabric’s components is the *Endorsement Policy (EP)* mechanism. An EP is an expression that specifies the network peers that must confirm a transaction, so that it can be considered as valid and subsequently appended to the ledger. From a business perspective, EPs contribute to the stability and trust of the network among its members — often competitors. For example, the following simple EP specifies that a transaction must be endorsed, or confirmed, by two out of three organizations, before it can be admitted to the ledger:

³<https://tendermint.com/>

⁴<https://www.goquorum.com>

⁵<https://hackernoon.com/hyperledger-fabric-the-most-popular-hyperledger-framework-b4485dea6a2c>

¹<https://sovrin.org/use-case-spotlight-the-government-of-british-columbia-uses-the-sovrin-network-to-take-strides-towards-a-fully-digital-economy/>

²<https://www.drakkenberg.ch>

```
OutOf(2, 'Org1.member', 'Org2.member',
      'Org3.member')
```

The aim of this paper is to investigate EPs by taking common needs and requirements of blockchain networks into consideration, specifically those applying to supply chains. The paper investigates why EPs are important for the integrity of the information on the ledger, and how it would be possible to create *illegal transactions* and admit these to the ledger. In this context, a transaction is illegal if it makes changes to the ledger that are not generated by the execution of a smart contract or chaincode. Multiple ways are proposed to make it more difficult to create these illegal transactions, including the addition of a *notary organization* to the network. This setting also studies a *multi-tenancy* case, where some organizations do not use their own peers in a blockchain network, but they take advantage of it via another organization's peer(s).

This paper is structured as follows. Section II describes the architecture of Fabric and EPs. Section III explains the challenges involved in defining EPs. Section IV presents various supply chain compositions, with different approaches for EPs and discusses their (dis)advantages. Section V reflects back on our work; Section VI reviews related work. Section VII closes by summarizing our findings and mentioning follow-up work.

II. BACKGROUND

First, a technical background on Fabric is given: its Execute-Order-Validate architecture and its concept of endorsement policies (EPs). For an in-depth presentation of Fabric's architecture, the reader is referred to [12].

A. Key concepts

In Fabric, multiple organizations collaborate within a *network*. A *Membership Service Provider (MSP)* is configured for each organization, allowing it to have full control over certificates and, thus, over the identities of its members. A network can contain multiple *channels*. Each channel is isolated from the others with separate ledgers holding their world state, participating organizations, chaincodes, EPs, and transaction history. Organizations must be in the same channel to interact with each other.

A *client* represents a user of the blockchain and does not maintain a copy of the ledger itself. To access or manipulate information stored on the blockchain, a client connects to one or more *peers* in a channel and submits transaction proposals. Every peer participating in a channel actively maintains a copy of the ledger. A peer receives new blocks, which contain new transactions from the *Ordering Service Nodes*, and validates the transactions in the block. Every peer keeps track of both valid and invalid transactions. A subset of (all) peers are *endorsing peers* or endorser, which have the additional task of endorsing transactions, i.e. to execute incoming transaction proposals and return the result of the transaction. Ordering Service Nodes, on the other hand, comprise the *Ordering Service (OS)*. They receive endorsed transactions from the clients, and distribute the transactions to the peers without looking at the transaction itself, i.e. without attempting to

Fig. 1. Transaction Flow in Hyperledger Fabric

validate them. The submission and acceptance/rejection of transactions for inclusion in the blockchain proceeds in three phases, which is discussed next.

B. Execute-Order-Validate Architecture

Fig. 1 summarizes the transaction flow in a network, following the three phases of its Execute-Order-Validate architecture.

1) *Execution Phase*: The flow starts with a client or application that wants to make a transaction. Before creating an actual transaction, the client submits a *transaction proposal*. The client software looks up the relevant EP(s), which specifies the endorsing peers, and submits the transaction proposal to these endorers. Upon reception of the transaction proposal, each endorsing peer *simulates* it by setting up an isolated container, which has a copy of the state of the blockchain. The corresponding *chaincode* (business logic) is executed with the parameters of the transaction inside the container so that the actual blockchain of the endorser remains unchanged. During the simulation, the endorser maintains a *readset* and a *writeset* of the transaction, which contain the state dependencies of the transaction and the changes it makes to the ledger, respectively. Once the simulation of the transaction proposal is completed, the endorser returns a signed endorsement to the client, containing the readset and the writeset. Since the client could have proposed the same transaction to multiple endorers, it can receive multiple endorsements.

2) *Ordering Phase*: Once the client has received enough endorsements, it can compose and submit the transaction to the OS, which consists of OS nodes that have a pluggable mechanism for reaching consensus on the order of transactions [12]. These nodes create blocks consisting of these transactions and broadcast them to all peers.

3) *Validation Phase*: After a peer has received a block containing transactions, block validation takes place. The process of validating blocks is performed by all peers individually. First, the peer runs the *Validation System Chaincode (VSCC)* to check whether the EP is satisfied in terms of appropriate number of valid endorsements coming from the expected

sources. This step is performed in parallel; the order of transactions is not important. Next, the peer performs a read-write conflict check. This check compares the versions of the asset states (dependencies) referenced in the readset, with the current version of the asset as stored on the peer. Differences in these versions lead to a conflict, as the transaction was valid for another version of the asset. Furthermore, the endorsements included in the transaction are checked to be consistent, as having different states of an asset would otherwise cause forks of the ledger. This step must be executed in sequence rather than in parallel, as two consecutive transactions could be made on the same version of the asset. If the transaction does not successfully pass either all checks in the VSCC or the checks performed by the read-write checks, the transaction is marked as *invalid*. Otherwise, it is marked as *valid* and appended to the ledger by implementing the changes recorded in the writeset.

C. Endorsement Policies

EPs can be written using a small domain-specific language, consisting of AND, OR, and OutOf expressions, an (optional) minimum number of endorsers required by the EP, and the (non-empty) list of endorsers and their roles relevant for the EP. Although OutOf-expressions can be written using only OR and AND expressions, they are simpler and more efficiently evaluated by Fabric's VSCC [13]. The syntax is as follows:

```
Expression(#Endorsers*, [Endorser_i.role]+)
```

In this paper, the `role` attribute is restricted to either `peer` nodes, or `member` nodes i.e., both peers and non-peers. Some concrete examples follow. Notice that expressions can be nested to form more complex expressions:

```
// endorsement required by all these peers
AND('Org1.peer', 'Org2.peer', 'Org3.peer')

// endorsement by any of these couples of peers
OR(
  AND('Org1.peer', 'Org2.peer'),
  AND('Org2.peer', 'Org3.peer'),
  AND('Org3.peer', 'Org4.peer')
)

// endorsement by three out of four referenced peers
OutOf(3, 'Org1.peer', 'Org2.peer', 'Org3.peer',
        'Org4.peer')
```

EPs are sufficiently expressive to account for multiple real-life situations. For instance, when an individual requests a new passport, a government agency should validate his/her identity before approving the transaction; to this end, the corresponding EP could use an OR-expression between different authorized agencies that could endorse the individual's identity and request. As another example, in the sale of a car both buyer and seller must agree on the price mentioned in the transaction, and both of them must appear in the EP: the transaction should not be added to the ledger without the consent from one of them (an AND-expression). While these examples highlight the importance of who should be included in an EP, the endorsement process in blockchain networks is executed automatically, without any human intervention. A result of the endorsement process is the writesets generated by

an endorser. This endorser creates those writesets according to rules specified in the chaincode that the endorser's organization has agreed upon. In other words, the process of endorsing a transaction could be considered as attesting to the validity of a transaction according to the business rules of the chaincode.

Fabric supports the definition of *chaincode-level* and *key-level* EPs, with a different scope. The former apply to *all assets* in the world state of the corresponding chaincode, whereas the latter can be set for *a specific asset* stored on the ledger. A chaincode-level EP is applied unless some key-level EP exists. Each individual asset could in principle have its own key-level EP. The term 'key' refers to the key that the asset has in the world state. These two kinds of policies differ in their flexibility. To upgrade chaincode-level EPs, the entire chaincode needs to be updated. For example, when a peer specified in an EP leaves the network or when a new peer joins the network, the policy needs to be updated. In contrast, changes to key-level EPs are programmable in the chaincode, and could be changed with each transaction appended to the ledger. Key-level EPs are stored in the writeset of a transaction, like any other update of an asset.

In general, key-level EPs are useful when the default chaincode-level EP is insufficient, as in the case of a channel where each peer would like to individually approve of all updates that happen with the assets owned by itself. Key-level EPs are also very useful when transactions need to be kept private from (a part of) the network. Consider, e.g., an asset transfer between two parties that do not want to disclose the details (e.g. agreed price) to the rest of the organizations in the channel. As discussed in the following, however, this is just another manifestation of an underlying phenomenon: a trade-off between (i) the need to keep the ledger valid and (ii) the need of organizations to maintain some degree of confidentiality in their transactions with each other.

III. CHALLENGES

A. Ledger Integrity

An important aspect of EPs is that transactions are only executed by the endorsing peers in the execution phase. When a transaction is broadcast, neither the OS nor the peers will execute the transaction. The peers in the network only check whether the transaction contains correct and enough endorsements according to the EP, but not the actual contents of the transaction. This aspect could potentially be abused by *colluding participants* to the network. Such parties can collaborate to submit signed and untrue writesets, not generated by the chaincode, under the condition that the colluding peers can together satisfy the EP. Let us consider the example of a supply chain with an EP that two out of ten peers have to endorse every transaction:

```
OutOf(2, 'Org1.peer', 'Org2.peer', [...],
        'Org10.peer')
```

Any two peers that are specified by the principals in the above EP could collude to create a fake writeset, sign it, and send it to the OS to make a malicious change to the ledger. Because the

transaction is only checked for its endorsements (rather than its contents), the peer has no reason to believe that this transaction could be illegal, and it would therefore be accepted to the ledger like any other. Such adverse behavior can be prevented by requiring a larger number of endorsers. Malicious actors will then need more colluders to fabricate writesets in order to succeed. Alternatively or additionally, it is possible to add a custom VSCC enforcing all peers to perform semantic checks on the contents of all submitted transactions before they are accepted, i.e. whether specific changes to the ledger conform to predefined (network-wide) rules. Although this is not meant to replace the business logic in the chaincode, it can and has been used to validate specific sensitive information.⁶

B. Data Confidentiality

Fabric provides *Attribute-based Access Control*, which restricts the execution of chaincode to certain roles, for instance for the *query* or *get* method. However, confidential data is still distributed to the peers and can be accessed by anyone with direct access to a peer's database.

To accomplish true data confidentiality (i.e., to ensure transaction data are only sent to authorized participants), Fabric offers two built-in solutions: *channels* and *private data collections*. As discussed in Section II-A, a channel helps two or more peers communicate by using transactions in a private network. Multiple channels can be used to hide transactions from peers not included in the channel. Private data collections allow peers to create transactions with part of the transaction hidden from other peers in the same channel. Authorized peers will receive the confidential information, while the unauthorized ones will obtain a hash of it instead. Besides these built-in options, solutions on the application layer (e.g. encryption) can be used to obfuscate the data from certain parties. Regardless of the chosen solution, ultimately, an endorsing peer needs access to the data to execute business logic and validations. A trade-off between integrity and confidentiality therefore arises, as discussed next.

C. Confidentiality vs. Integrity Trade-off

The trade-off between integrity and confidentiality is made clear by examining the balance between them. In a private setting, for example, an EP could specify that only a small select group of endorsers are enough to satisfy it. This will increase the transaction confidentiality (due to the controlled number of endorsers), but it will reduce integrity as the barrier for potentially malicious organizations to collude is lowered. Alternatively, it is possible to require an endorsement from more peers, making it more difficult to collude. Consequently, these additional organizations have to be included in the private data collections or channels, causing the confidential information to be revealed to them.

A solution to the aforementioned problem could be to introduce a *notary service* consisting of one or more mutually distrusting neutral organizations to the network. The

assumption here is that these *Notary Organizations (NOrgs)* are selected for not being biased towards one participant over another. A NOrg could be a market regulator, service provider, or legal entity formed for a blockchain consortium. Introducing these NOrgs to the network is well-aligned with the permissioned and joint nature of Fabric, as a NOrg could be an organization acting as auditor or a licensed notary in the traditional sense. One possible EP could then be that an endorsement is needed from at least one NOrg. For instance, a transfer of goods could be endorsed by the seller, the buyer, and a governmental body or a trade association. An immediate benefit is that all simulations would be executed by a peer with no interest in the actual contents of the transactions and no benefit from colluding. Thus, by introducing a neutral node in an EP, fraudulent organizations are less likely to collude in a way that satisfies the EP.

On the other hand, a risk of requiring every transaction to be endorsed by the same NOrg is that the channel loses its decentralized characteristic. Even with legal and operational controls in place, a single point of trust and failure is in principle unwanted. The EP can specify that every transaction has to be endorsed by at least two or more NOrgs. Then, multiple independent and neutral organizations have to endorse every transaction, making it impossible for a single NOrg to collude. Essentially, multiple NOrgs monitor each other. However, an additional risk of unavailability emerges when transactions have to be endorsed by all NOrgs of a channel. If at least one NOrg refuses connections from a certain country/organization, some participants might not be able to satisfy the EP, and thus, cannot add transactions to the ledger. If this NOrg goes completely offline, then no transaction proposals can be endorsed. A solution to this would be to have more NOrgs in the network than required by the EP, allowing a client to choose out of multiple NOrgs. Then, when some NOrgs are unavailable, a client can simply request for an endorsement from another NOrg.

The concept of having more notary nodes in the network than strictly required, was also advocated by Corda in their technical white paper [11]. For instance, an EP could specify a transaction has to be signed by two collaborating organizations and two out of three NOrgs, as shown in the example below:

```
AND (
  OutOf(2, 'NOrg1.member', 'NOrg2.member', NOrg3.
    member'),
  OutOf(2, 'MSP1.member', [...], 'MSP50.member')
)
```

The likelihood of collusion, the needed level of confidentiality, and the desirability of including NOrgs is therefore highly dependent on the composition of the network and the specific use case. The same also holds for any other solution which attempts to balance these needs. For this purpose, the following discussion is grounded to specific case studies coming from the industry.

IV. SUPPLY CHAIN CASE STUDIES

A supply chain involves multiple organizations with a common goal of moving goods from suppliers to retailers. Supply

⁶<https://jira.hyperledger.org/browse/FAB-11149>

Fig. 2. Simple supply chain with a Supplier (S), trader 1 (T1), trader 2 (T2), trader 3 (T3) and Retailer (R) are transacting a product. The network might have more peers, and therefore potential endorsers, but only T1-T3 are shown for simplicity's sake.

chains are complex, as they involve many organizations with different interests. According to an empirical and quantitative study by the World Economic Forum, supply chains face many barriers including market access, border administration, transport and communications infrastructure, and business environment [14]. Although the use of blockchain for supply chains might not completely eliminate these barriers, it could certainly help reduce them [15]. In addition, there is a growing demand for proving the origins of certain goods, for instance, for the ethical sourcing of rare minerals, or food safety.

Two different supply chain compositions are discussed: a simple supply chain and a supply chain with multi-tenancy, which are abstractions from real-life use cases in which some of us are involved. In both cases, it is assumed that all transactions are private and may not be seen by other organizations.

A. Simple Supply Chain

Within this paper, the simple supply chain is defined to consist of a single supplier, three traders and a single retailer, as shown in Fig. 2. Traders buy goods from an originator or another trader, and sell them to a trader or retailer for a higher price. Typically, the retailer is the last participant of the supply chain who sells the product or good to a consumer. In a production implementation, there can be hundreds of participants in such a chain, including banks, customs, inspection agencies, logistics companies, and insurers. Blockchain is used to digitize and exchange relevant data such as inspection certificates, receipt confirmations, trade agreements or other commercially significant events. The information is often competitively sensitive and EPs are crucial to ensure the integrity of the data.

Approach 1: EP for optimal confidentiality: For each trade in a simple finished goods supply chain, there are at least two organizations involved: the seller and the retailer (final buyer). The most privacy-oriented EP specifies that a transaction must be endorsed by only two organizations:

```

OutOf(2, 'S.member', 'T1.member',
        'T2.member', 'T3.member', 'R.member')

```

Requiring only a single endorsement allows a single organization to make any change to the ledger unilaterally. Requiring more than two endorsements causes the transaction contents to be potentially revealed to uninvolved organizations. The main advantage of this approach is therefore that confidential information is only disclosed to the organizations that have an interest in the transaction. The fact that only two organizations are able to collude is a major concern, especially in a supply

Fig. 3. Supply chain with a Supplier (S), traders (T1, T2, T3), two tenant traders (T4, T5) in an alliance with T2, and a Retailer (R). As in the case of the simple supply chain, the network might have more DPs, and more tenants which are not shown.

chain setting. However, if the use case does not permit confidential data passing through a neutral trusted organization, this EP style is preferred.

Approach 2: EP for optimal integrity: An EP can be defined that ensures the information written to the ledger is legitimate and properly generated by executing chaincode. To increase the difficulty of collusion about false changes on a ledger, multiple NOrgs are included:

```

AND (
  OutOf(2, 'NOrg1.member', 'NOrg2.member',
          'NOrg3.member'),
  OutOf(2, 'S.member', 'T1.member',
          'T2.member', 'T3.member', 'R.member')
)

```

This approach ensures that multiple NOrgs have to endorse a transaction, before it is considered as valid. This implies that colluders will require the help of two NOrgs, before being able to succeed, which is unlikely. The downside of this approach is that the confidential information is revealed to the two NOrgs, although the client has the ability to individually choose them. By having the third and extra NOrg in the EP, new transaction proposals can still be endorsed if one NOrg is unavailable. It is worth mentioning that the private information is not revealed to other traders. The concept of requiring a subset of multiple endorsements from NOrgs, can also be applied in the next approaches to improve the integrity, but is left out for simplicity.

B. Supply Chain with Multi-tenancy

In cloud environments, *multi-tenancy* is defined as a software architecture in which multiple tenants share the same resource or instance of an application [16]. A *tenant* is defined as a group of users having the same view on an application and can be regarded as a single entity. In our case, the term multi-tenancy is used to specify that some organizations within the network do not have (endorsing) peers and therefore rely on peers belonging to other organizations for endorsements. An organization may not want to host a peer for various reasons, such as to reduce costs, or a lack of time or expertise to host and maintain a peer. An important drawback of choosing for multi-tenancy is the necessity of trusting a third party to sign transactions on the organization's behalf. For simplicity, these organizations without peers are referred to as tenants since they share the same resources (peers). Organizations that host their own peers are referred to as Direct Participants (DP).

For this second case study, a similar composition is used: there is a single supplier and a single retailer with three traders in between. However, the second trader (T2) shares its peer with two tenants (T4 and T5) as illustrated in Fig. 3, forming an *alliance* with them.

Approach 3: Introducing a NOrg on behalf of the tenants: For this case study, it is possible to have an EP similar to the one for optimal privacy in the previous case study, allowing a transaction to be endorsed by only two organizations. However, when a tenant wants to trade with a DP, a third party is necessary to endorse the transaction on behalf of the tenant. Two DPs can still trade with each other without revealing their transaction contents to a third:

```
OutOf(2, 'S.member', 'T1.member', 'T2.member',
        'T3.member', 'R.member', 'NOrg1.member')
```

A disadvantage of this approach is that tenants are highly dependent on the NOrgs and unable to participate in endorsing transactions. This also means that two tenants are not able to trade, unless a second NOrg is introduced, and requiring both notaries to sign the transaction. Further, if a NOrg in the EP is unavailable, a tenant is unable to trade. In general, this approach does not attempt to reduce the chance of collusion.

Approach 4: A NOrg must always endorse a transaction: The EP of the previous approach can be modified for higher integrity of the ledger, by making an endorsement from a NOrg mandatory:

```
AND('NOrg1.member',
    OR('S.member', 'T1.member', 'T2.member',
        'T3.member', 'R.member'))
)
```

A clear advantage of this approach is all transactions must be endorsed by at least one NOrg, causing private data to be revealed to it. This EP also does not tackle the problem of centralization and unavailability, presented in Section III-C, although this could be addressed by introducing more NOrgs, similarly to Section IV-A. The disadvantage is all transaction contents are exposed to at least one NOrg. In addition, if two tenants want to trade two counter-signing NOrgs have to be introduced, as above.

Approach 5: Enforcing attestation of all involved traders using a custom VSCC: Let us assume a DP wants to trade with one of the tenants. If the DP creates the transaction proposal, the tenant cannot endorse it, as it has no peers. On the other hand, if a client of the DP creates the transaction proposal, this client can request an endorsement from the DP. After this client receives the endorsement back from the DP, the client can inspect the writeset. If the client then inspects the endorsement created by the DP and attests it, the client of the tenant can submit the transaction to the OS. If the client does not approve of the writesets generated by the DP regarding the trade, the client can discard the transaction. By following this transaction flow, the organizations of the tenant and the DP have the influence to accept or reject the transaction, without delegating control to a third party such as a notary.

To realize this concept, an EP can be defined that requires only a single endorsement from any DP. However, for

transactions between two DPs, at least two endorsements are needed, i.e. from both organizations. To enforce this, semantic checks can be added by using a custom VSCC. If the buyer and the seller have their own peers, the VSCC verifies that the transaction contains their endorsements. If a transaction contains only one endorsement during validation, the VSCC checks that the endorsement is from a DP, while the ordered transaction itself must be signed and submitted by a tenant. The assumption made here, is that the fields containing the buyer and seller are public and not part of a private data collection. If the fields referencing the buyer and seller are marked as private fields, then it is not possible to define such a semantic check. The corresponding EP is:

```
OR('S.member', 'T1.member', 'T2.member',
    'T3.member', 'R.member')
```

A benefit of this approach is that a tenant is able to participate in the supply chain, without relying on a proxy (e.g. NOrg). However, all trades with a tenant must be created by this tenant. In other words, a DP cannot propose a purchase to a tenant, although the inverse is possible. Another advantage of this approach is that the custom VSCC ensures that every trade between two organizations is actually agreed upon by these same organizations. Therefore, it is impossible for two organizations to create a trade between two others, irrespective of whether they collude or not.

Approach 6: Enforcing the approval of all involved traders using a custom VSCC and a NOrg: Similar to previous approaches, Approach 5 can be expanded to always include an endorsement from a NOrg, while still allowing a DP to approve or reject trades:

```
AND('NOrg1.member',
    OR('S.member', 'T1.member', 'T2.member',
        'T3.member', 'R.member'))
)
```

An advantage of this approach is that it still allows a tenant to trade with a DP, while still allowing the tenant to have full control over what trades it approves or rejects, involving its own organization. Thanks to the VSCC, two DPs that are trading must both endorse a transaction, although this is not explicitly specified in the EP. Additionally, it will be much harder to collude on fake writesets, because every transaction must be endorsed by a neutral organization. More (redundant) NOrgs could be added to the EP to decrease the risk of centralization and unavailability.

V. DISCUSSION

EPs are a simple but powerful tool in maintaining control over the information which actually becomes part of the ledger in Fabric networks. In essence, EPs allow network members to define who they trust to validate their transactions. When defined correctly, they also minimize the opportunities for collusion between bad-faith actors. EPs can even allow competitors to participate in the same network without requiring them to disclose information with each other, but only with their customers. Furthermore, the use of EPs can enhance the concept of multi-tenancy in blockchain applications, which

promotes co-opting the technology without the burden of running a full-scale peer. This is of particular interest for the industrial sector.

However, at the same time multi-tenancy by means of EPs re-introduces the need for a (limited but not insignificant) amount of *trust*. More specifically, tenant organizations must put some trust in the DPs they are rely upon and their good faith in endorsing policies on their behalf. It can be easily seen how a malicious DP can hurt a tenant in their alliance by refusing to endorse a valid policy (on time) or by endorsing clearly manipulated policies in collusion with other peers. This is an inherent risk of multi-tenancy, and needs to be considered against the benefits of not running a peer. In any case, evidence of the malicious behavior of a peer would be reflected in the shared ledger anyway, so tenants could at least identify there is an issue with their peers. Lastly, what has also become clear from the previous section is that EPs are largely case-specific, following the nature of the specific domain and application in which they operate. As such, there are no generic rules for defining them, but rather a set of potential approaches to be followed, with both advantages and disadvantages.

While not covered by this work, there are two further issues to be discussed. The first relates to the effect of EPs in the performance of blockchains, especially with respect to their availability. The security triad of confidentiality, integrity, and availability has been discussed in the context of permissioned blockchain networks before [17]. The use of EPs with many endorsers, or ones with specific notaries can decrease availability due to their explicit dependency on these parties for the EPs to be validated. Without a wide-scale deployment of Fabric using EPs yet, however, it is difficult to evaluate their actual impact to the operations of the involved organizations, especially since the processes involved for endorsement as discussed in Section II are fully automated by Fabric. With respect to other aspects of performance now, Thakkar et al. [13] studied the performance of Fabric v1.0 and concluded that EPs with subexpressions are generally slower, and that the latency of a transaction increases linearly with the amount of endorsements it includes due to the execution of VSCC. However, it is unclear to what extent these results hold for the latest version of Fabric (v1.4). Investigating this issue would be the subject of future work.

The second issue relates to mechanisms enabling privacy in Fabric. Anonymity and unlinkability have not been discussed in this paper, although they are currently supported by Fabric to some extent. Specifically, they depend on Identity Mixer (or Idemix) which is a cryptographic protocol suite based on Zero-Knowledge Proofs (ZKP) [18]. Despite it provides anonymity and unlinkability of the client, the client's organization is revealed, providing only anonymity of clients within the same organization. Therefore, using Idemix by more than one organization is not recommended by the Fabric team⁷, limiting its usefulness in a supply chain setting. Another limitation is that Idemix does not provide anonymity and

unlinkability of endorsers up to Fabric v1.4. According to Fabric's documentation, these limitations should be resolved by future versions, providing true anonymity and unlinkability for both clients and peers of multiple organizations within the same channel.

VI. RELATED WORK

Many variants of blockchain architectures are currently under development. Among those similar to Fabric, the popular Quorum network requires every node to validate public transactions, which could unnecessarily increase the overhead [19]. On the contrary, private transactions are visible only to the transacting parties. In case of colluding, this setting might lead to undesirable consequences and undermined trust. Corda, on the other hand, takes advantage of notaries, guaranteeing a transaction will be signed only if all the input states are unconsumed, without using any blockchain [11]. We discussed how this concept can be adopted in Fabric for both cases.

With respect to the adoption of blockchain in supply chains, Chod et al. introduce a blockchain protocol, *b_verify*, which stores hashed transaction data on an existing public network (e.g. Bitcoin) as well as analytical modeling for corporate lending, by leveraging transparency through blockchain [1]. The authors also provide a use case in agricultural supply chain, which deals with public data by law. Therefore, privacy is not an issue in their case.

El Maouchi et al. propose a permissionless traceability system for supply chains [3]. However, their approach allows everyone to access the stored data, while they require every node of the network to validate broadcast transactions. In other words, privacy and performance issues emerge. The same research group designs a protocol to overcome those limitations [20]. This paper is the first work to focus on decentralized privacy in supply chain traceability systems by applying a variety of existing cryptographic tools. Nevertheless, transactions' input using the protocol El Maouchi et. al adopted, RingCT protocol, can be "fully deanonymized" [21]. This results in performance limitations in creating transactions, because of the ring signatures. This significant overhead might discourage the adoption of leading organizations for a significant number of transactions.

By default, Fabric uses Role-based Access Control. Khan et al. [22] propose an alternative way of managing access control for private blockchain frameworks using Distributed Usage Control (DUC). Besides determining if access to a certain object on the ledger can be granted, using DUC would make it possible to track the usage of accessing these objects stored on the ledger. After a certain quota is exceeded, access to the objects could be denied. Their paper also suggests a prototype implementation in Fabric, by introducing a new system chaincode using key-level EPs. The discussion in this work therefore provides a more general view on the subject by evaluating different approaches to endorsement policies.

The most relevant related work we are aware of is a study by Lu et al. [23] on reaping the benefits of blockchain technologies compared to traditional automotive supply chains.

⁷<https://hyperledger-fabric.readthedocs.io/en/release-1.4/idemix.html>

In that setting, suppliers and manufacturers can be transaction endorsers, without taking into consideration cases where those two actors are allowed to see only limited information. Additionally, the prior work does not discuss non-chaincode modifications due to a potential collusion of malicious endorsers. On the contrary, this work follows a more technical point of view by comparing different approaches that can be used in a broader scope.

VII. CONCLUSION

The Endorsement Policy (EP) component is important to ensure that all information admitted to a Hyperledger Fabric's shared ledger is actually generated by executing appropriate smart contracts. Specifically, using a simple but expressive language, EPs define which network peers are responsible for validating transactions for inclusion in the ledger. In previous sections we described various techniques that enable organizations to participate in supply chain networks, so as to create private transactions and their corresponding ledger state. The challenge of choosing an EP in a private setting is analyzed, leading to the identification of a trade-off between the integrity of the information stored on the ledger and the need for data confidentiality. Finally, the notion of a notary service is introduced as a (sub-optimal) solution to this trade-off. Our work shows that defining EPs is domain- and application-specific, with multiple parameters to be taken into consideration. A special interest in multi-tenancy settings is underlined in this work, because organizations are often involved in a blockchain network as a peer of another organization.

Directions for future work include performance evaluations (in terms of network overhead and effectiveness of the proposed EP strategies) in the presence of colluding peers. For the former setting, a testbed should be established to perform benchmark measurements⁸, including transaction throughput and latency, number of pending transactions, CPU and RAM overhead for notary nodes, under different EPs (and subsequent block size - in case of key-level EP) configuration [24]. Selecting the appropriate tools requires significant attention, due to the fast pace of frameworks' evolution. Additionally, to validate proposed EPs it would be insightful to simulate various scenarios with different percentages of colluding organizations. The availability of knowledge and datasets can be a limiting factor for this purpose, given that blockchain enterprise networks have been launched only recently.

VIII. ACKNOWLEDGEMENTS

This research is partially supported by an IBM Community Grant, awarded to the University of Groningen. We thank the IBM CIC Benelux's Management Team for their support, our colleagues who offered us industrial insights, and the anonymous reviewers for their suggestions.

⁸E.g. by means of <https://github.com/hyperledger/caliper>

REFERENCES

- [1] "Blockchain and the value of operational transparency for supply chain finance," Mack Institute for Innovation Management, Tech. Rep. Working Paper Series, 2018. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3078945
- [2] F. Tian, "An agri-food supply chain traceability system for china based on rfid & blockchain technology," in *13th International Conference on Service Systems and Service Management*, 2016.
- [3] M. El Maouchi, O. Ersoy, and Z. Erkin, "Trade: A transparent, decentralized traceability system for the supply chain," in *1st ERCIM Blockchain Workshop, European Society for Socially Embedded Technologies*, 2018.
- [4] IBM. (2019) IBM Food Trust: trust and transparency in our food. [Online]. Available: <https://www.ibm.com/nl-nl/blockchain/solutions/food-trust>
- [5] R3. (2019) Cryptoblack r3 case study. [Online]. Available: https://www.r3.com/wp-content/uploads/2019/04/CryptoBLK_CS_Jan2019.pdf
- [6] IBM. (2018) Blockchain for supply chain. [Online]. Available: <https://www.ibm.com/blockchain/industries/supply-chain>
- [7] C. Catalini and J. Gans, "Some simple economics of the blockchain," in *MIT Sloan Research Paper No. 5191-16*, 2017.
- [8] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *bitcoin.org*, 2008.
- [9] V. Buterin *et al.* (2014) Ethereum: A next-generation smart contract and decentralized application platform. [Online]. Available: <https://github.com/ethereum/wiki/wiki/white-paper>
- [10] M. Shirer, J. Goepfert, and S. Soohoo. (2019) Worldwide blockchain spending forecast to reach \$2.9 billion in 2019, according to new IDC spending guide. [Online]. Available: <https://www.idc.com/getdoc.jsp?containerId=prUS44898819>
- [11] M. Hearn. (2016) Corda: A distributed ledger. technical white paper. [Online]. Available: <https://www.corda.net/content/corda-technical-whitepaper.pdf>
- [12] E. Androulaki, A. Barger *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*. ACM, 2018.
- [13] P. Thakkar, S. Nathan, and B. Viswanathan, "Performance benchmarking and optimizing hyperledger fabric blockchain platform," in *2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*. IEEE, 2018.
- [14] J. Moavenzadeh *et al.* (2013) Valuing growth opportunities. [Online]. Available: http://www3.weforum.org/docs/WEF_SCT_EnablingTrade_Report_2013.pdf
- [15] R3. (2019) Can blockchain future-proof supply chains? a brexit case study. [Online]. Available: <https://www.crowell.com/files/2019-January-Can-Blockchain-Future-Proof-Supply-Chains.pdf>
- [16] R. Krebs, C. Momm, and S. Kounev, "Architectural concerns in multi-tenant saas applications," *Closer*, vol. 12, 2012.
- [17] H. C. Hanebeck, N. Hewett, and P. A. McKay. (2019) Inclusive deployment of blockchain for supply chains. [Online]. Available: <https://www.weforum.org/whitepapers/inclusive-deployment-of-blockchain-for-supply-chains-part-3-public-or-private-blockchains-which-one-is-right-for-you>
- [18] J. Camenisch, S. Mödersheim, and D. Sommer, "A formal model of identity mixer," in *International Workshop on Formal Methods for Industrial Critical Systems*. Springer, 2010.
- [19] A. Baliga, I. Subhod, P. Kamat, and S. Chatterjee. (2018) Performance evaluation of the quorum blockchain platform. [Online]. Available: <https://arxiv.org/pdf/1809.03421.pdf>
- [20] M. El Maouchi, O. Ersoy, and Z. Erkin, "Decouples: a decentralized, unlinkable and privacy-preserving traceability system for the supply chain," in *34th ACM/SIGAPP Symposium on Applied Computing*, 2019.
- [21] D. A. Wijaya *et al.*, "Monero ring attack: Recreating zero mix-in transaction effect," in *17th International Conference On Trust, Security And Privacy In Computing And Communications*. IEEE, 2018.
- [22] M. Y. Khan, Zuhairi *et al.*, "An extended access control model for permissioned blockchain frameworks," *Wireless Networks*, 2019.
- [23] D. Lu *et al.*, "Reducing automotive counterfeiting using blockchain: benefits and challenges," in *International Conference on Decentralized Applications and Infrastructures (DAPPCON)*. IEEE, 2019.
- [24] "Hyperledger blockchain performance metrics," Hyperledger Performance and Scale Working Group, Tech. Rep., 2018. [Online]. Available: https://www.hyperledger.org/wp-content/uploads/2018/10/HL_Whitepaper_Metrics_PDF_V1.01.pdf