Auditable secure network overlays for multi-domain distributed applications

Cushing, R.; Koning, R.; Zhang, L.; de Laat, C.; Grosso, P.

Publication date
2020

Document Version
Final published version

Published in
IFIP Networking 2020 Conference and Workshops

License
Article 25fa Dutch Copyright Act

Link to publication

Citation for published version (APA):
Cushing, R., Koning, R., Zhang, L., de Laat, C., & Grosso, P. (2020). Auditable secure network overlays for multi-domain distributed applications. In *IFIP Networking 2020 Conference and Workshops: June 22-25, 2020, Paris, France* (pp. 658-660). IEEE. https://ieeexplore.ieee.org/document/9142725

Auditable secure network overlays for multi-domain distributed applications

Reginald Cushing
University of Amsterdam
Amsterdam, The Netherlands
r.s.cushing@uva.nl

Ralph Koning
University of Amsterdam
Amsterdam, The Netherlands
r.koning@uva.nl

Lu Zhang
University of Amsterdam
Amsterdam, The Netherlands
l.zhang2@uva.nl

Cees de Laat
University of Amsterdam
Amsterdam, The Netherlands
delaat@uva.nl

Paola Grosso
University of Amsterdam
Amsterdam, The Netherlands
p.grosso@uva.nl

***Abstract*—The push for data sharing and data processing across organisational boundaries creates challenges at many levels of the software stack. Data sharing and processing rely on the participating parties agreeing on the permissible operations and expressing them into actionable contracts and policies. Converting these contracts and policies into a operational infrastructure is still a matter of research and therefore begs the question how should a digital data market place infrastructure look like? In this paper we investigate how communication fabric and applications can be tightly coupled into a multi-domain overlay network which enforces accountability. We prove our concepts with a prototype which shows how a simple workflow can run across organisational boundaries.**

***Index Terms*—overlay network, marketplace, cryptography, actors, auditing.**

I. INTRODUCTION

Data sharing across organisational boundaries has the potential to open up new insights, as well as to create novel business opportunities. Digital Data Marketplace (DDMs) [1] [2] are emerging as architectures to support this mode of interactions. They rely on the participating parties agreeing on the permissible operations (*market transactions*) and expressing them into actionable contracts and policies. Converting these contracts and policies into a operational infrastructure is still a matter of research and therefore begs the question how should a digital data market place infrastructure look like?

For better accounting and enforcement of policies, the application and network can not be observed in isolation of each other. Network accountability has often been limited to nodes on the network [3]. Our approach is to tie the application and network into an overlay where activity on the network can be observed as functions of the applications. This enhanced observation allows to create an additional plane to the traditional data and control plane. We call this plane the audit plane which can observe the application and network activities and verify the actions. We show this in a working prototype.

This research is partly funded by the DL4LD project. https://dl4ld.nl
Annex to ISBN 978-3-903176-28-7© 2020 IFIP

II. DEFINING THE ACTORS

The complexity of such a system can be better understood by describing a DDM as a group of actors with specific roles. An actor is a uniquely identifiable entity in the system that has certain functions, is reactive as well as proactive i.e. function calls can be a result of a reaction to a message from another actor or a result of an internal routine. To further categorise actors into roles, we can segregate them into layers. The lower layers deal with connectivity while higher layers deal with policies. In our scenario an actor or actors are part of one of the layers and communicate with other actors in different layers (Figure 1) or domains to facilitate and coordinate market applications.

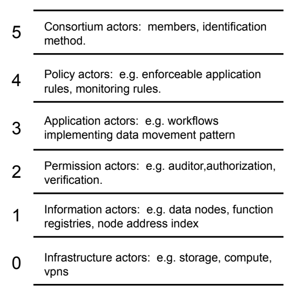

5
Consortium actors: members, identification method.
4
Policy actors: e.g. enforceable application rules, monitoring rules.
3
Application actors: e.g. workflows implementing data movement pattern
2
Permission actors: e.g. auditor,authorization, verification.
1
Information actors: e.g. data nodes, function registries, node address index
0
Infrastructure actors: e.g. storage, compute, vpns

Fig. 1. Roles of actors in a DDM can be categorised into layers. The lower the level the closer to infrastructure the actor roles are. E.g. Layer 0 deal with having the capabilities to setup virtual resources such as storage, compute and setting up VPN network connections.

III. TRANSACTION-BASED SECURE NETWORK OVERLAY

An operational infrastructure is composed mainly of actors from layers 3,2,1. The main purpose of a network overlay is to bring together these actors. In network terminology actors are synonymous to nodes on the network thus a network node can be considered the implementation of an actor. For the rest of this paper actor and node are used interchangeably. Addressing on the network is done using public/private keys where the public key is the address of the node and the private key is only known to the node. The advantages of this cryptographic addressing mechanism are twofold: First, nodes are not tied to IP addresses and thus can be dynamically allocated to different physical locations. Second, communications from nodes are signed by their private keys which can be verified by any node in the network since the address (public key) is used to verify the signatures. The latter is important since a core function of the network is creating an audit trail of activity on the network and audits need to verify to whom communications belong. Unlike other addressing schemes such as IP, these cryptographic addresses are non-transferable i.e. the same address can not be reused for different nodes at any point in time which strengthens traceability since any actions signed by a node can always be traced back to that node. Name services are responsible to translate public key address to actual IP endpoints to allow communication to proceed over lower protocols. Another side effect feature of such addressing is that nodes can sign each others keys creating a web of trust between nodes that can easily be verified by following the signatures.

Nodes on the network expose functions which can be called by other nodes. The function routes takes the form:

$$publickey/modulename/functionname$$

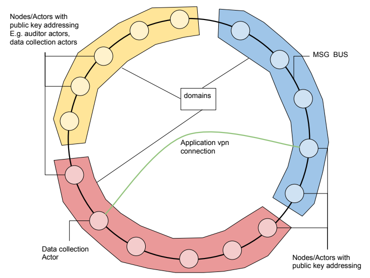
Registries assist in the discovery of the functions exposed on the network. Since such registries can pose an attack vector in a multi-domain architecture, part of the agreed consortium rules dictate how many registries per domain are hosted on the network and what consensus method is used to agree on a registry query e.g. all registries must return the same result to a query for that query to be deemed valid or a majority have to agree on a result.

Data actors are responsible for exposing the shareable data collections. The cryptographic addressing scheme allows for cryptographic keys assigned to particular datasets. The owner of the dataset is the node that possesses the private key for the collection. The dataset addresses are the bases of application transactions whereby the application's data movement is listed as a transaction between nodes on the overlay. Data object can thus be given unique addresses in the form

$$publickey : datahash$$

In this case the address points to the unique data collection and the data hash identifies a data object within the collection e.g. a file. Since the address is the public key, any recipient of the data can check data signatures done by the data collection owner's private key.

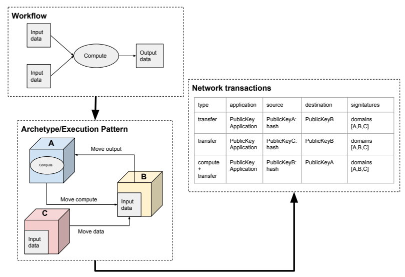
Communication between nodes is of two main types, signal and data. Signalling is done through messaging (Figure 2) to invoke functions on other nodes. Data communication is done through dedicated connections per application transaction. For example, to open a connection between two data nodes, the application will perform the signalling to the responsible nodes that manage the endpoints. Obviously, not all applications are authorised to just open connections, for this reason signing and verification nodes are responsible to issue signatures for applications. These signatures are used by the control actors to perform the required actions such as open connections, copy data, etc.



Nodes/Actors with public key addressing E.g. auditor actors, data collection actors
MSG BUS
domains
Application vpn connection
Data collection Actor
Nodes/Actors with public key addressing

Fig. 2. Actors organised on an overlay network. A message bus

Applications on the network are a list of transactions which represent the dataflows of an application. Figure 3 shows how a typical workflow application can be realised into a set of transactions which are applicable on the overlay network. The workflow application is organised into a pattern to conform to an agreed archetype (these patterns are agreed upon at the policy layer). The application combined with the archetype results in a list of transactions. A transaction would instruct copying of data between nodes of different domains and instruct computations to be performed on the different domains. Transactions are used as instructions to drive the network. Since transactions are made of identifiable source, destination and application, every transaction can be tracked, monitored and audited. This level of granularity of auditing allows to build an audit trail of applications on the network.



Network transactions

type	application	source	destination	signitatures
transfer	PublicKey Application	PublicKeyA: hash	PublicKeyB	domains [A,B,C]
transfer	PublicKey Application	PublicKeyC: hash	PublicKeyB	domains [A,B,C]
compute + transfer	PublicKey Application	PublicKeyB: hash	PublicKeyA	domains [A,B,C]

Fig. 3. Realisation of a workflow application to a set of network transactions.

Auditing is a major concept in our overlay. Auditor actors are able to sign application transactions. Signatures are produced according to their internal policies (from layer 5). The overlay can have auditor nodes from different domains in which case transactions get multiple signatures. The signed transactions act as an verification and authorisation layer which the lower control layer will use to effect transactions. For example, an application include a transaction to move data from A to B, auditors check if this action conforms with pre-agreed policies in which case the transaction is signed. The actors in the control layer will then provision a network connection from A to B and move the data. In the case where auditors disagree on signatures, the consensus mechanism decided by the consortium dictates the outcome e.g. majority auditors sign or all auditors need to sign. To prevent audit logs from being tempered with, independent auditor nodes are responsible to maintain separate databases that can be cross-referenced to detect tempering.

IV. Prototype

To prove our approach we implemented an overlay prototype with the concepts introduced in section III that we showcased for the first time during the 2019 SuperComputing conference in Denver. Our current prototype shows a secure execution environment that can be used in a multi-domain DDM. In particular, we focus on orchestrating virtual infrastructure such that it only allows the agreed application transactions to occur and to provide detailed logging that is used during the runtime stage of the auditing process.

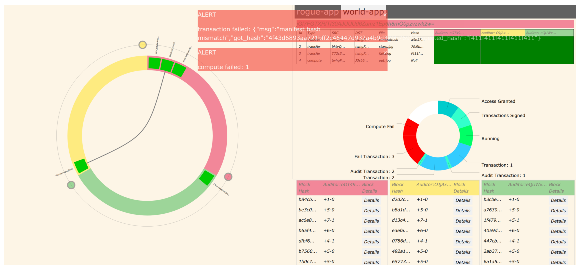
All nodes expose their functions on a message queue. The function path described in Section III is used as a message routing key thus enabling the responsible node to react to messages directed to it specifically. In our prototype we assign names to certain node roles. Most importantly we introduce the following node types:

- **Bucket node** are data nodes which are Docker containers that contain either Data (*data buckets*) or Compute objects (*compute buckets*). A bucket allows a dataset to be directly addressable using the public key.
- **Controller node** manages the life cycle of buckets i.e. creation, deletion and VPN connections between buckets. Each domain has at least one controller.
- **Planner node** is the node responsible for the planning and execution of the application transactions. This is a Docker container with a also a unique address. Each application has has a single Docker container as its planner thus each application has a unique address.
- **Auditor node** is a node responsible for authorisation through signatures of actions such as application transactions. The auditor is Docker container which monitors the communication of the overlay and signs requests. Each domain has at least one Auditor.

By logging the state and the connectivity of the buckets and other nodes, the auditor can analyse the different logs for inconsistencies e.g. a transaction not being part of an application or a transaction happening outside its allowed time window. When inconsistencies occur, the auditor raises alerts, lower the confidence in the secure execution of the application, or abort the execution causing the application to fail. In this case, its up to the partners to find the underlying problem or disagreement and to resolve it.

To maintain a temper resistant audit log, the prototype borrows some ideas from blockchain. Each auditor maintains a linked list of logs using hashes. A set of logs is called a block. Periodically, a new block entry is created by hashing the previous set of logs and including the hash in the next list entry. The list of hashes means that a modification to a log entry will break the hashing. Still nothing stops a malicious attempt to temper with the log to modify the whole database. For this reason new block hashes are announced to the auditors which they will include them into their own logs. With this approach logs from different auditors can be cross-referenced for tempering.

Figure 4 depict a running prototype with a *hello world* workflow. The workflow consists of two data image inputs a compute and an image output. The archetype of this workflow dictates that a third party domain will do the computation thus compute, and two data inputs are transferred to a third domain for computation and the output is copied out. The demo depicts the detection of data being tempered which is detected by transaction signatures failing verification. Other scenarios include low auditor approval on transaction signatures, malicious planner trying pass on extra transactions and rogue application on the network.



ALERT
transaction failed: {"msg":"manifest hash mismatch","got_hash":"4f43d6893aa721bff2c46447d932a4b9d3

ted_hash":"f411f411f411f411f411"}

ALERT
compute failed: 1

rogue-app world-app

jGf7FGTXRfTI30AJUUUd6Zumz1Ep6h8rhO0pzvzwk2w=

Id	Type	SRC	DST	File	Hash	Auditor: oOT49...	Auditor: OJjAx...	Auditor: eQUWx...
1	[illegible]	[illegible]	[illegible]	compute.sh	a9e37...			
2	transfer	bktvQ...	twhgF...	stars.jpg	7fc9b...			
3	transfer	T72c3...	twhgF...	fail.png	f411f...			
4	compute	twhgF...	J3sL6...	out.jpg	Null			

Block Hash	Auditor:oOT49...	Block Details
b84cb...	+1-0	Details
be3c0...	+5-0	Details
ac6e8...	+7-1	Details
b65f4...	+6-0	Details
dfbf6...	+4-1	Details
b7560...	+5-0	Details
1b0c7...	+5-0	Details

Block Hash	Auditor:OJjAx...	Block Details
d2d2c...	+1-0	Details
b8d1d...	+5-0	Details
d13c4...	+7-1	Details
e3efa...	+6-0	Details
0786d...	+4-1	Details
492a1...	+5-0	Details
65773...	+5-0	Details

Block Hash	Auditor:eQUWx...	Block Details
b3cbe...	+1-0	Details
a7630...	+5-0	Details
1f479...	+5-1	Details
4059d...	+6-0	Details
447cb...	+4-1	Details
2ab37...	+5-0	Details
6a1a5...	+5-0	Details

Fig. 4. Left circle: shows the overlay network with data buckets in different colour coded domains. The lines between buckets depict dynamic VPN connections being created per transaction. Top right: the workflow transaction list and the planner address that is coordinating the execution of the transactions. Centre right: the application progress. Bottom right: the 3 auditor logs (1 from each domain) logging and auditing the activity on the network.

V. Conclusions and Future Work

In this paper we showed how tying and application with the network into an overlay enhances the information that is observed. This was done by decomposing workflow applications into a list of network transactions that are mapped to the physical network through the overlay. Accountability is ameliorated through cryptographic addressing whereby actions on the network are always signed. Multi-domain logging using linked lists of hashes reduces the risk of domains tempering with their logs which further increases transparency on the network. Here we focused mainly on the data exchange part of the system, the compute part opens other challenges such as validation which we are looking at for future work.

References

[1] S. Cisneros-Cabrera, A. Ramzan, P. Sampaio, and N. Mehandjiev, "Digital marketplaces for industry 4.0: a survey and gap analysis," in *Working Conference on Virtual Enterprises*. Springer, 2017, pp. 18–27.

[2] A. Zerdick, K. Schrape, A. Artope, K. Goldhammer, U. T. Lange, E. Vierkant, E. Lopez-Escobar, and R. Silverstone, *E-conomics: Strategies for the Digital Marketplace*. Springer Science & Business Media, 2013.

[3] A. Haeberlen, P. Kouznetsov, and P. Druschel, "Peerreview: Practical accountability for distributed systems," in *Proceedings of Twenty-First ACM SIGOPS Symposium on Operating Systems Principles*, ser. SOSP '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 175–188. [Online]. Available: https://doi.org/10.1145/1294261.1294279