Approaches to Platform Flows

Table of Contents

Overview

- Lineage
- Ownership
    - Accountability / Attestation
- …

Types of Platforms

- *Infrastructure* platforms, hold or forward the data to the recipient without altering the data content.
- *Enrichment* platforms, which may blend data from several sources before it reaches the recipients
- *Multitenancy* platforms

Overview of current model

The diagram below shows a simplified version of the model used by Waltz to represent flows.

Logical flows connect two endpoints, which are typically applications. The logical flows can be decorated with *data types* (not shown).

Each logical flow may have *physical flows* associated with it. Each physical flows has a *physical specification* which describes the content of the data. Physical specifications can be reused across physical flows.

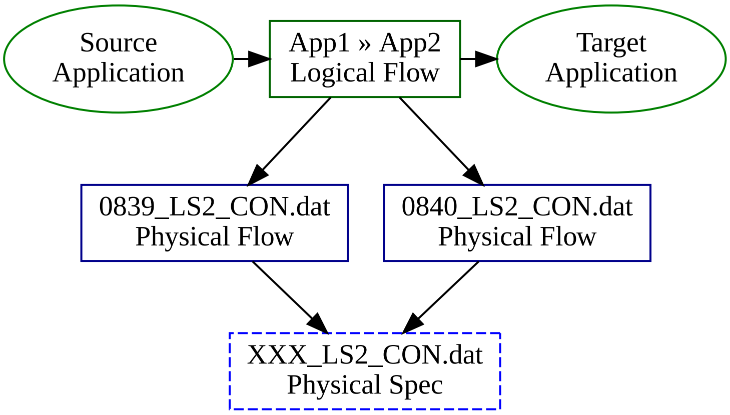
Source Application
App1 » App2 Logical Flow
Target Application
0839_LS2_CON.dat Physical Flow
0840_LS2_CON.dat Physical Flow
XXX_LS2_CON.dat Physical Spec

Figure 1. Current flow model

Waltz's flow database is made up from numerous point to point logical flows and their physical counterparts. The totality of those flows represents the known flow network and can be represented as a directed graph.

Lineage

Lineage can be thought of as a pathway taken though the logical flow directed graph. Starting from a source node, traversing logical flows, to reach a target node.

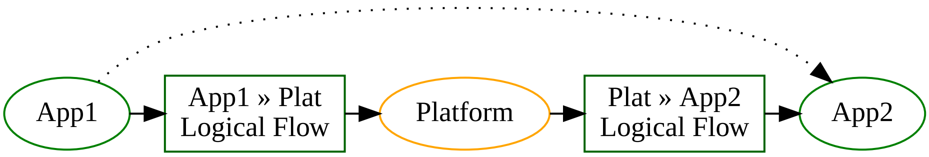

Figure 2. Lineage using the existing Waltz model

In Waltz there are limited ways to express indirect flow relationships such as `App 1` → `App 2` in the diagram above (shown with the dotted arc). To show lineage users have a choice:

- Use **Waltz's Flow Diagram** capabilities which allow users to quickly draw a route through the network (by using an editor constrained by the underlying graph).
- Use **physical flow tagging** to associated flows. This is conceptually similar to the diagrams mentioned above, but some users prefer this basic approach.
- **Export the flows and use tools** such as Solidatus to represent a more detailed view of lineage

Attestation

In the current model we view the **consuming application as the owner of the flow**. The reasoning behind this is based on a simple belief that the consumer is more likely to be aware of the flow than the producer.

For simple app-to-app flows systems this model works well and attestation is straightforward; the consumer should attest all inbound flows. This is depicted in the figure below (`App C` must attest to the accuracy of the flows from `A` and `B`).

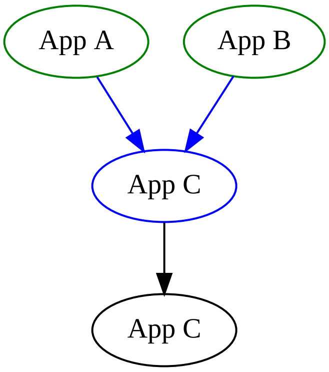

Figure 3. Example of attestation scope for simple apps

However, when we have aggregation/distribution platforms the **attestation burden falls upon the platform**. Often the platform owners are not in a position to be able to accurately perform the required attestation, especially in multi-tenancy situations.

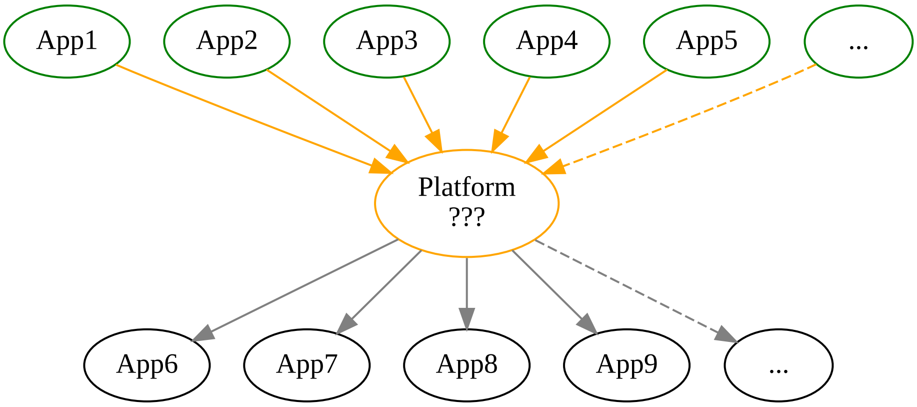

Figure 4. Direct attestation for platforms

This is depicted in the following figure, the green flows show a large volume of inbound flows to the platform which may have little knowledge of their content and/or accuracy.

Attestation Enhancements

To overcome this situation we could:

Option 1:* Exempt platform *applications from flow attestations. This could be achieved via a simple flag, assessment or membership of a group. To ensure the flows are attested, the obligation to attest would move to the providers of data to a platform.

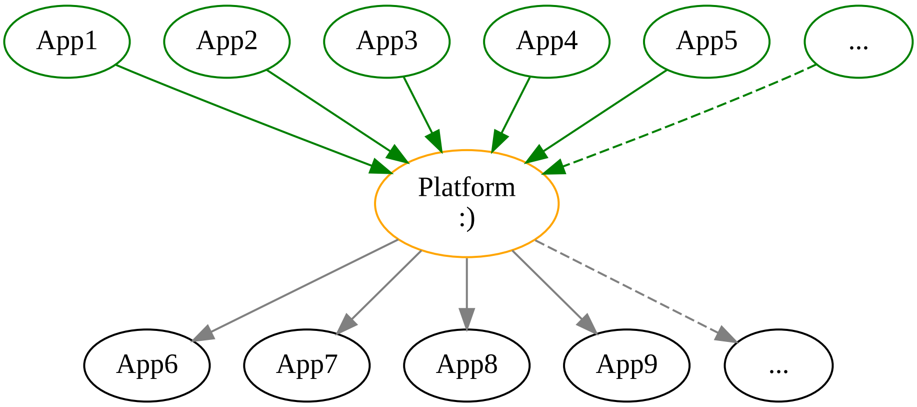

Figure 5. Exempting platforms

CAUTION

We would need to consider what happens when/if platforms exchange data. If they are exempt who has the attestation obligation ?

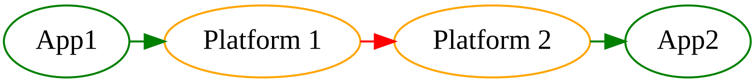

Figure 6. Platform to platform flows, who attests ?

***Option 2:* Declare attestation obligations on a per flow basis**, this would be at the physical flow and/or logical flow level. These obligations will inform the ui who is responsible for each attestation.

Whilst the second option will give more accurate results and handles more situations, we believe the development effort and associated runtime attestation effort will outstrip this gain.

Alternative flow representations

'Via' routing

In this approach the flow of data is recorded between `App1` and `App2`. The platform in between them is recorded as a detail on the physical flow.

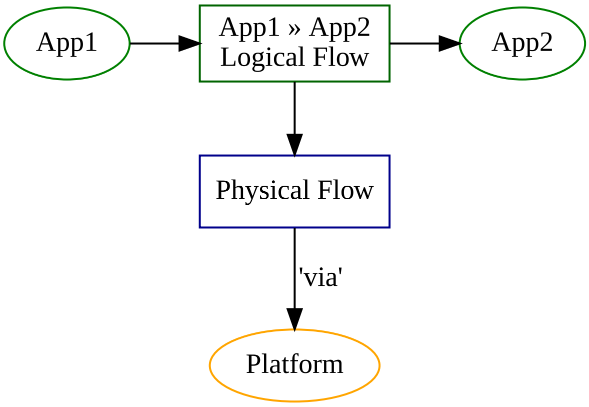

Figure 7. Platforms using physical routing

Pros	Cons
Simplifies basic lineage	Multi-hop *via* routes difficult to express (ordering etc)
Intuitive understanding	Usage of platforms more difficult to understand as may be via a direct flow or an indirect detail on the physical flow.
Somewhat solves the attestation problem as the flow is now between apps.	By omitting the platform from the main flow all recipients must know about, and record, `App 1` as the originator.

'Ultimate Recipient' routing

A variation of *via routing* is to store the intended recipient as an optional (list?) attribute on the physical flow.

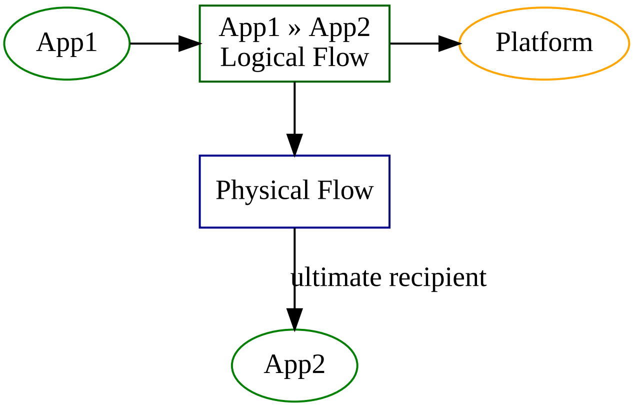

Figure 8. Platforms using physical routing

Pros	Cons
Simplifies basic lineage	Multi-hop *via* cannot be expressed
-	Inconsistency of recording. If App1 does not know, or particularly care about, App2 (or vice versa)
Intuitive understanding	-
-	-

Layers

This approach proposes additional layers of flows. We currently have

- logical
- physical

Appendix: Waltz terminology

This section defines what specific terms mean in this document

- **Logical Flow**, abstractly describes all flows between two entities (typically apps)
- **Physical Flow**, a specific instance of a *logical flow*
- **Physical Specification**, details of what is being transferred in a *physical flow*
- **Data Types**, hierarchical structure representing all types of data within the organization
- **Measurable Category**, (aka *Taxonomy*)
- **Measurables**, hierarchical structure representing items in a *measurable category*
- **Rating Scheme / Item**, used to describe the relationship between entities (typically apps) and *measurables*
- **Measurable Rating**, the actual linkage of an entity to a *measurable* using a *rating scheme item*