**Notes after meeting Torsten: feedback on previous note (08-05-2025)**

This document includes 4 steps:

1. Terminology
2. Dataset
3. The problem we aim to solve using network science (nodes, edges)
4. Representing the Dataset as a Network Graph
5. From Graph to Artifact Recommendation (with metrics)

## **Step 1**: Terminology

* Cluster node: Refers to network infrastructure devices (e.g., routers, switches, NATs)
* NS node: Refers to a node in the network science graph (e.g., service endpoint or workload)

## **Step 2**: Dataset

We will use data from OSI Layers 1 to 4 and associated capture metadata. While the dataset also includes Layers 5–7, we will exclude these from our current analysis but retain them for possible future use.

Dataset Columns by OSI Layer

* Layer 1–2 (Data Link): eth.eth.src, eth.eth.dst, eth.eth.raw, eth.eth.tree, eth.eth.type, eth.eth.type\_raw, eth.eth.stream
* Layer 3 (Network): ip.ip.version, ip.ip.hdr\_len, ip.ip.len, ip.ip.ttl, ip.ip.proto, ip.ip.id, ip.ip.flags, ip.ip.checksum, ip.ip.frag\_offset, ip.ip.src, ip.ip.dst, ip.ip.addr, ip.ip.host, ip.ip.src\_host, ip.ip.dst\_host, ip.ip.stream
* Layer 4 (Transport): tcp.tcp.\*, udp.udp.\*, tcp.Timestamps, udp.Timestamps
* Metadata: frame.frame.\*, file\_type, file\_encapsulation, packet\_size\_limit, number\_of\_packets, file\_size, data\_size, capture\_duration, earliest\_packet\_time, latest\_packet\_time, data\_byte\_rate, data\_bit\_rate, sha1, sha256, capture\_oper-sys, capture\_application, number\_of\_interfaces\_in\_file, interface\_#0\_info

Reference: <https://www.ietf.org/archive/id/draft-tuexen-opsawg-pcapng-03.html>

## **Step 3**: Problem Framing

Original Problem: We aim to migrate workloads to a hybrid cloud. The challenge is to identify the best-suited deployment artifact for each workload, based on its communication patterns.

Definition of 'Best-Suited': An artifact that meets the workload’s technical requirements while enabling operational automation (e.g., scale, failover, isolation). Artifacts abstract compute, storage, and networking resources; their choice impacts performance, resiliency, and integration. By aligning communication behavior with artifact characteristics, we aim to reduce operational overhead.

Why the Problem Still Holds: Although the dataset spans OSI Layers 2 to 7, our analysis focuses on Layers 2–4. These layers are sufficient to reconstruct flows, model dependencies, and infer workload behavior such as statefulness, coupling, and infrastructure reliance. From this, we can derive artifact recommendations even without higher-layer application data.

## **Step 4**: Representing the Dataset as a Network Graph (Layers 2–4 Focus)

The graph includes two categories of NS nodes:

**Node Definition**

| **Node Type** | **Constructed From** | **Purpose** |
| --- | --- | --- |
| Workload Node | (MAC, IP, Port) | Represents a specific application endpoint or service |
| Simplified Service Node | (IP, Port) | Abstracts the service identity for coarser analysis |
| Inferred Role Node | Post-role mining label | Reclassifies nodes as "frontend", "database", etc. |
| Infrastructure Node | MAC/IP without L4 data | Represents routers, switches, NATs, or firewalls |

**Edge Definition**

| **Component** | **Derived From** | **Purpose** |
| --- | --- | --- |
| Edge | (ip.src, src\_port) → (ip.dst, dst\_port) | Directional flow between nodes |
| Edge Attributes | ip.proto, tcp.len, frame.time\_epoch | Protocol, byte volume, and temporal information |
| Edge Weight | Aggregate packet or byte count over time | Used for dynamic and behavioral modeling |

## **Step 5**: From Graph to Artifact Recommendation

After constructing the graph:

1. Use community detection (e.g., Louvain) to group workloads into application-level clusters.
2. Use role mining and centrality metrics to classify nodes by function (gateway, backend, leaf node).
3. Assess communication intensity, duration, and frequency to determine operational characteristics.
4. Use routing paths and TTL behavior to detect infrastructure dependencies.

From these insights, we infer technical needs and recommend deployment artifacts:

| Graph Trait | Workload Characteristic | Suggested Artifact |
| --- | --- | --- |
| High PageRank or Betweenness | Central, orchestrating workload | High-availability VM or managed instance |
| Short-lived UDP sessions, low centrality | Stateless, ephemeral service | Serverless function or autoscaling container |
| Dense intra-cluster traffic | Tightly coupled microservices | Pod or StatefulSet in container orchestration platform |
| Long-lived flows | Stateful backend or database | Reserved VM or persistent container |
| Spans routers or NATs | Distributed, latency-sensitive | Hybrid deployment or edge-based placement |

This structure allows us to map low-level network data to actionable infrastructure design, enabling automated, data-driven decisions in hybrid cloud migration.

## **Step 6**: Example

Check website:

<https://the-it-ninja.blogspot.com/2018/07/building-network-communication-map-with.html>