Real-Time Resource Authorization @ Telenor Norway

LDBC London – 19 Nov 2013
by Sebastian Verheughe

telenor

Telenor Norway

Subsidiary of the Telenor Group

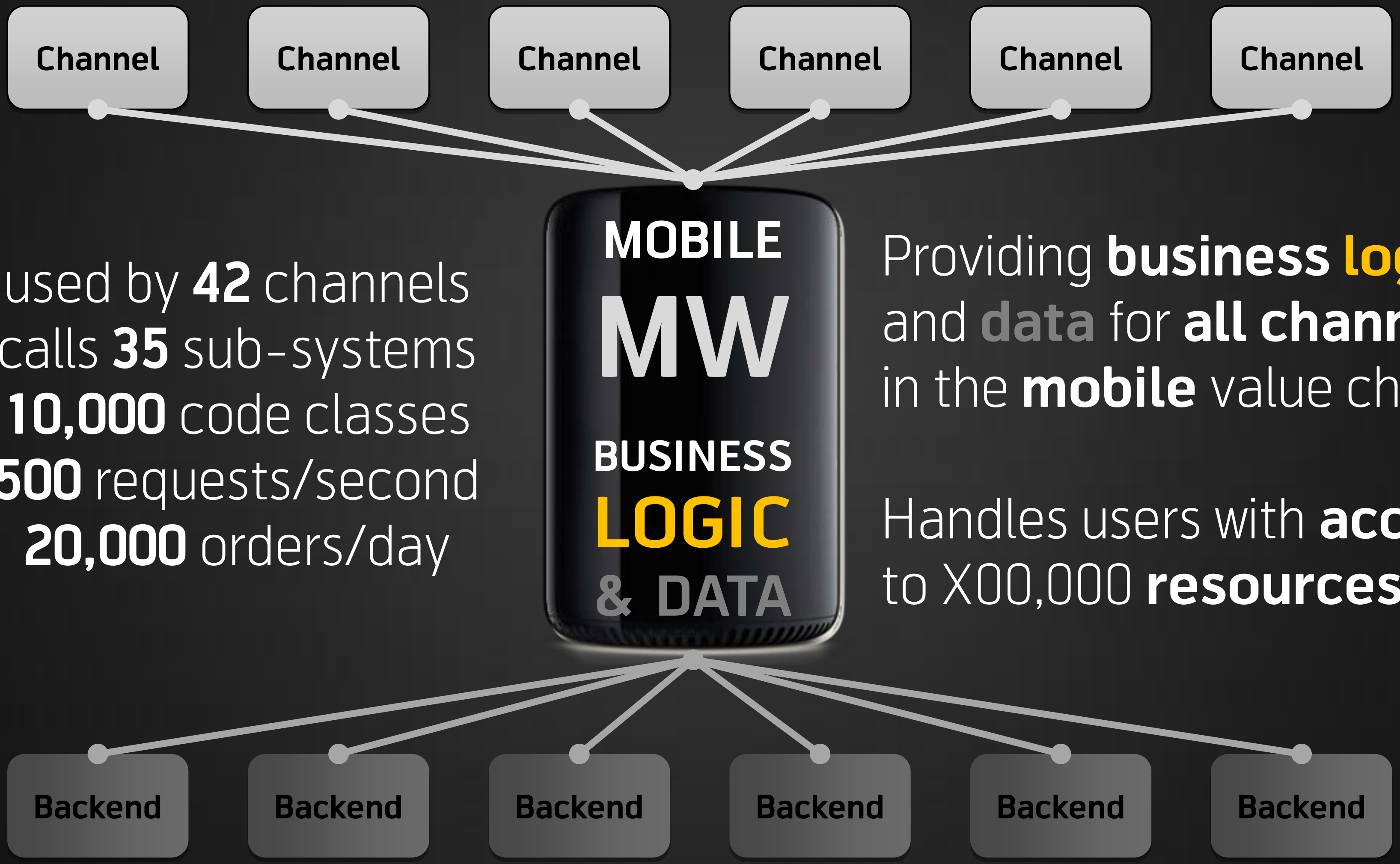
~1 billion GBP in mobile revenues 2012

Sebastian Verheughe

Lead Developer for Neo4j solution

Coding Architect

Telenor Norway Middleware Services



used by **42** channels
calls **35** sub-systems
10,000 code classes
500 requests/second
20,000 orders/day

Providing **business logic** and **data** for **all channels** in the **mobile** value chain

Handles users with **access** to X00,000 **resources**

Our **Problem**

20 minutes to **calculate** all accessible resources

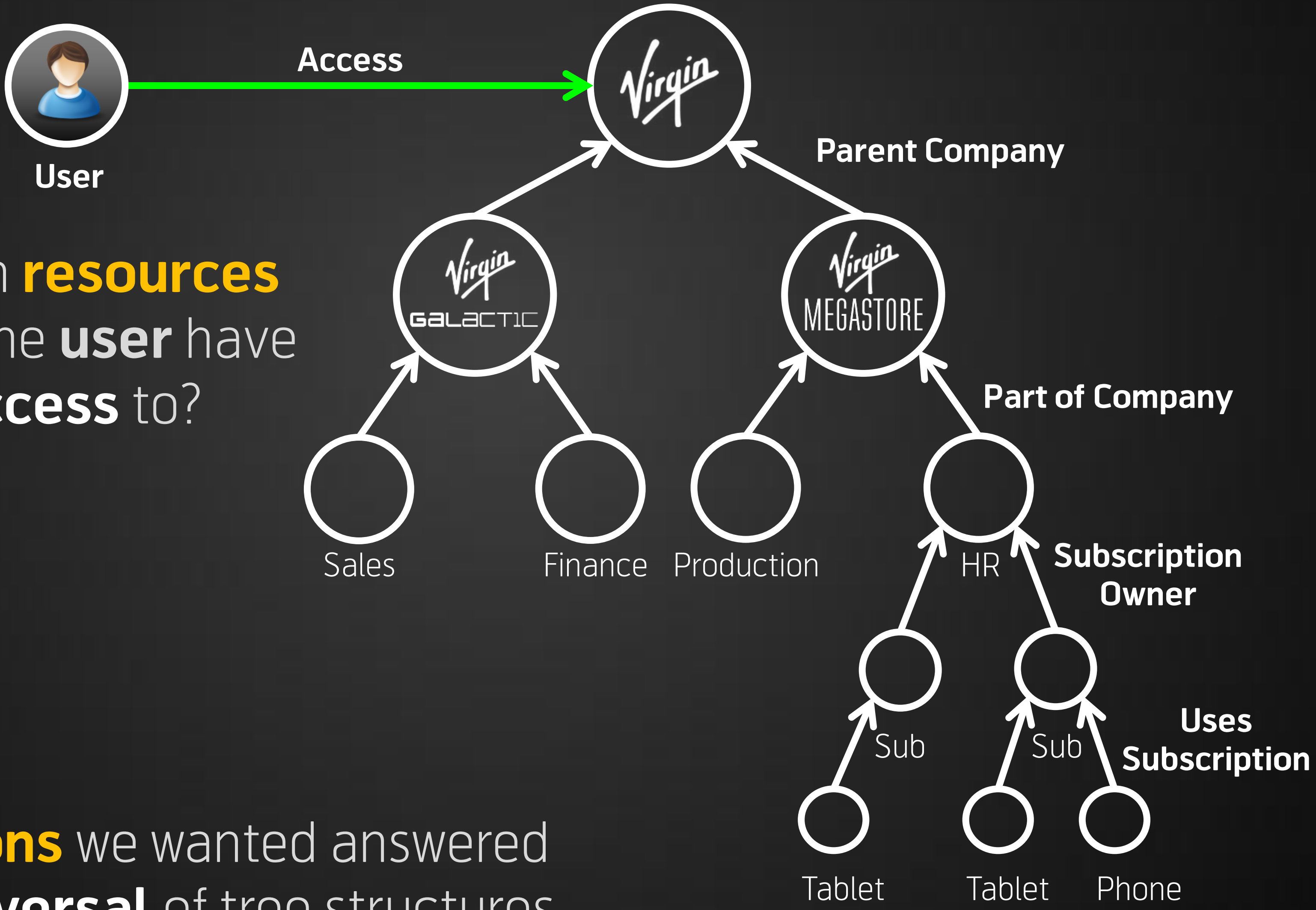
1500 lines of **SQL** to implement the authorization **logic**

“solved” by **caching** data going **stale**

and the solution did **not scale**…

Why a Graph Database?

Which **resources** does the **user** have **access** to?



The **questions** we wanted answered required **traversal** of tree structures.

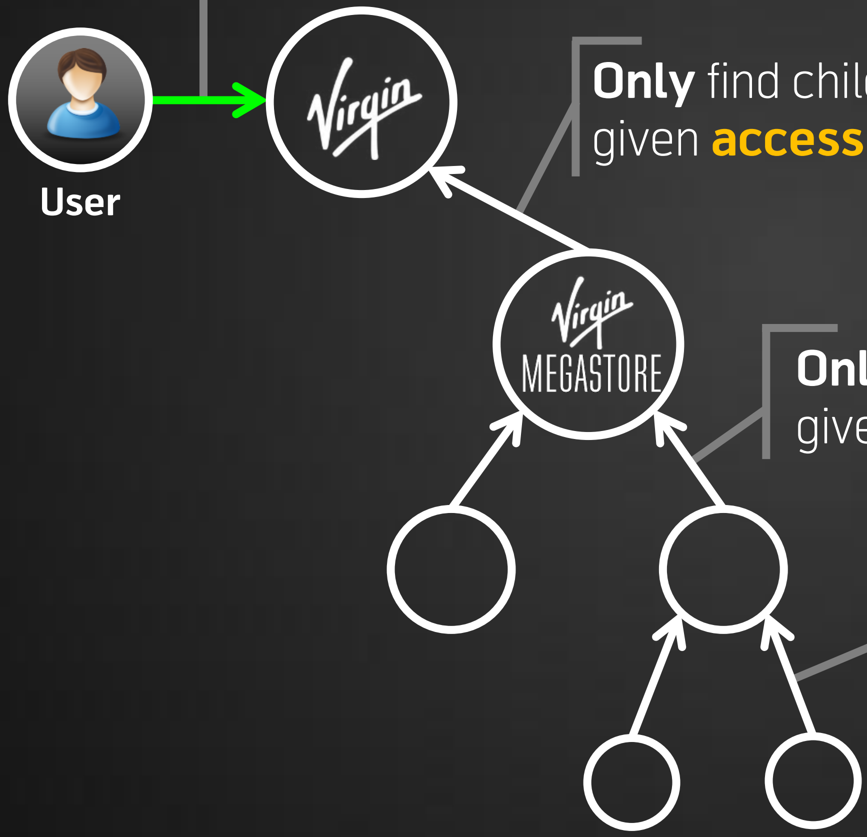
Why a Graph Database?

1. **Performance**
   The graph database gives awesome performance compared to what we were able to get in our old RDBMS.

2. **Query Simplicity**
   The graph query language (Java API) makes is much simpler to write and understand the traversal business logic. The SQL we had was almost impossible to understand.

Conditional **Rules**

ACCESS is given with the following include parameters:
access to subsidiaries and **access to content**

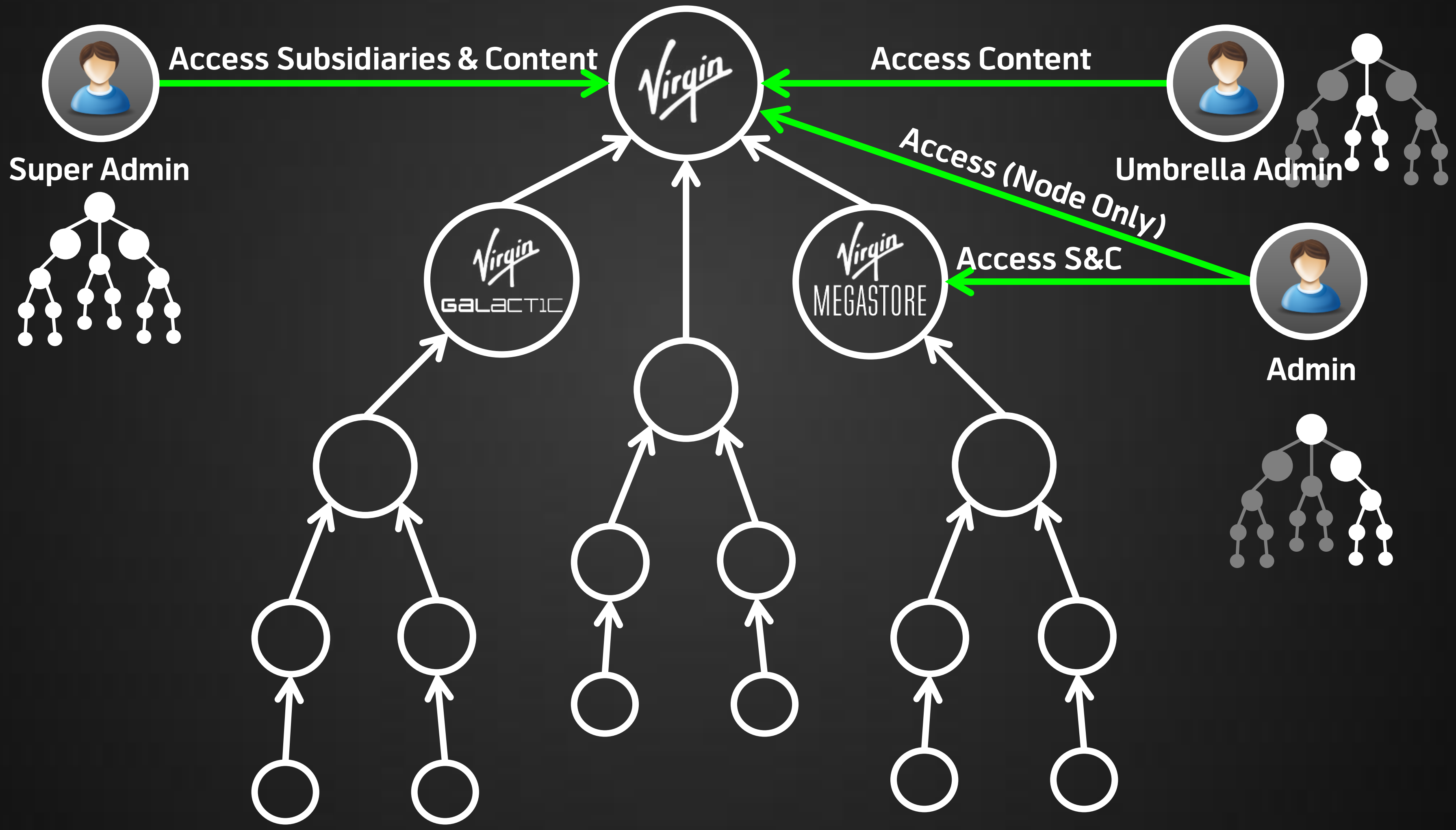


Only find children of **PARENT COMPANY**
given **access to subsidiaries** is allowed

Only look at **PART OF COMPANY**
given **access to content** is allowed

Only look at **SUBSCRIPTION OWNER**
given **access to content** is allowed

Different Access Needs



Graph Algorithm

Prerequisite: The user node

1. Follow all **ACCESS** relationships and **read the access parameters** on the relationship

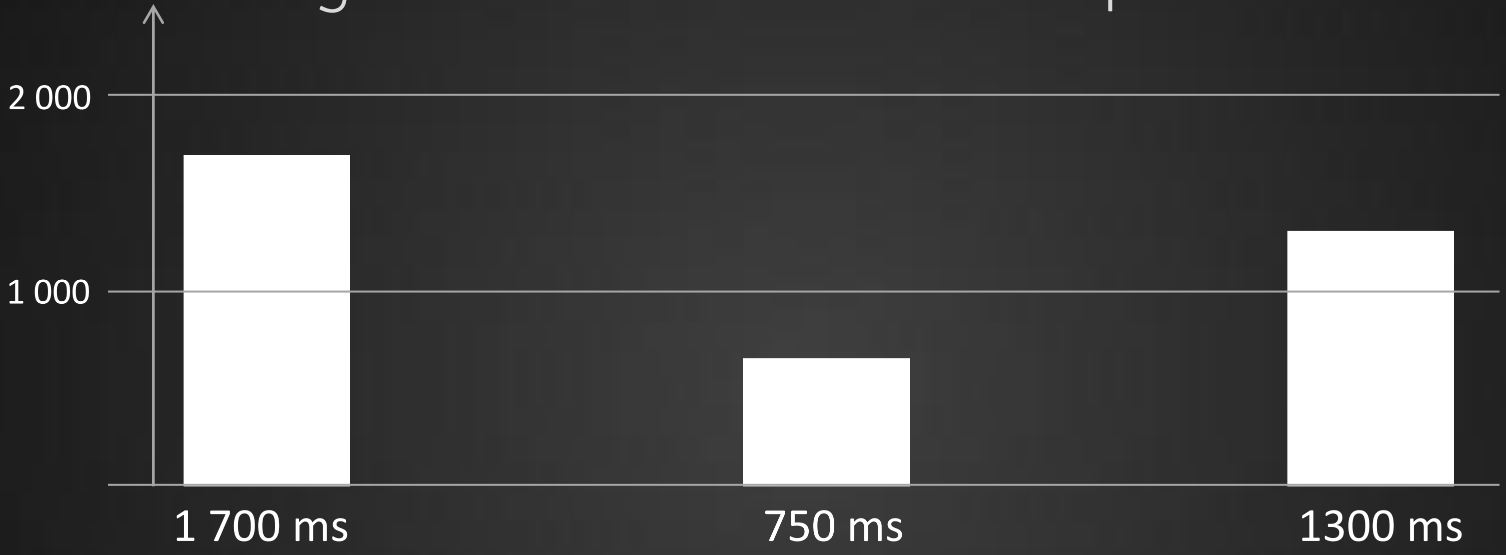
2. Follow all **PARENT COMPANY** relationships given **access to subsidiaries** is allowed

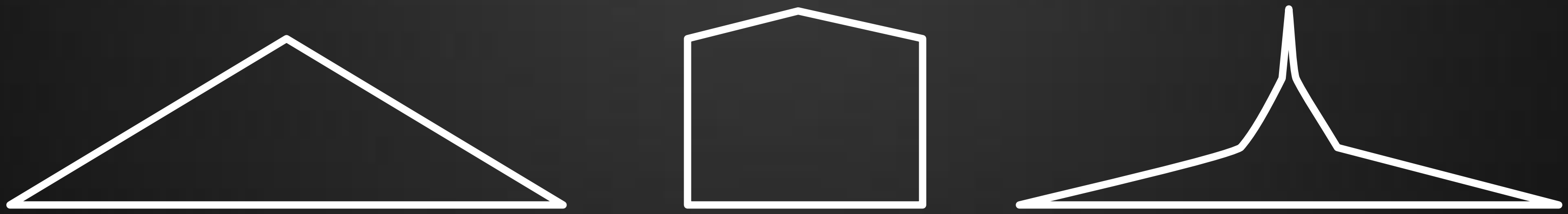
3. Follow all **PART OF COMPANY** relationships given **access to content** is allowed

4. Follow all **SUBSCRIPTION OWNER** relationships given **access to content** is allowed

Different **Graph Structures**

get **all** accessible subscriptions



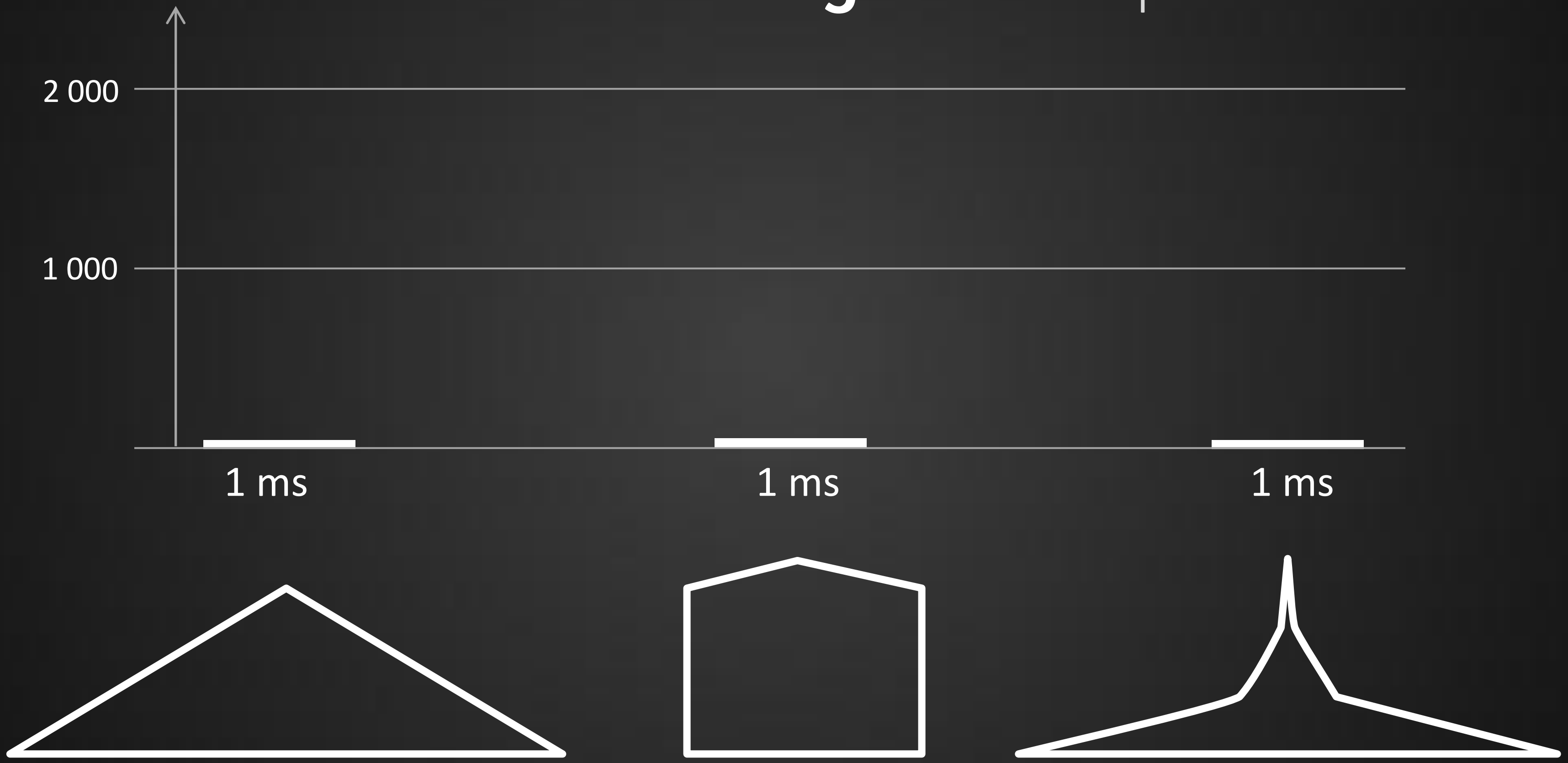


Company X: 147 000 | Company Y: 52 000 | Company Z: 95 000

Data from **test** – repeated **prod** sampling gave **~2.4** sec for **215,000** subscriptions

Different Graph Structures

check access to **single** subscription



Company X: 147 000

Company Y: 52 000

Company Z: 95 000

Production Performance

retrieve **all** accessible resources

	RDBMS Disk	RDBMS (mem cached)	Graph In-Heap
Company X	12 min	18 sec	< 2 sec
Company Y	22 min	58 sec	< 2 sec
Company Z	3 min	15 sec	< 2 sec

Check **single** resource access

1 ms

No operational problems in production

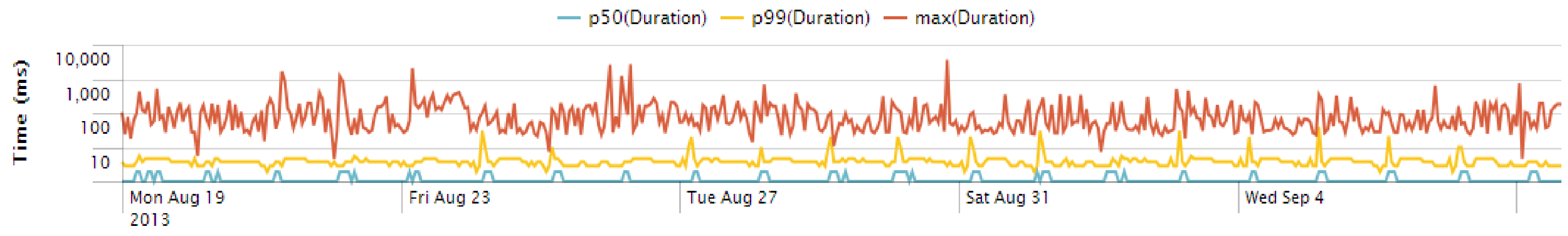
Technical Details

Production Details

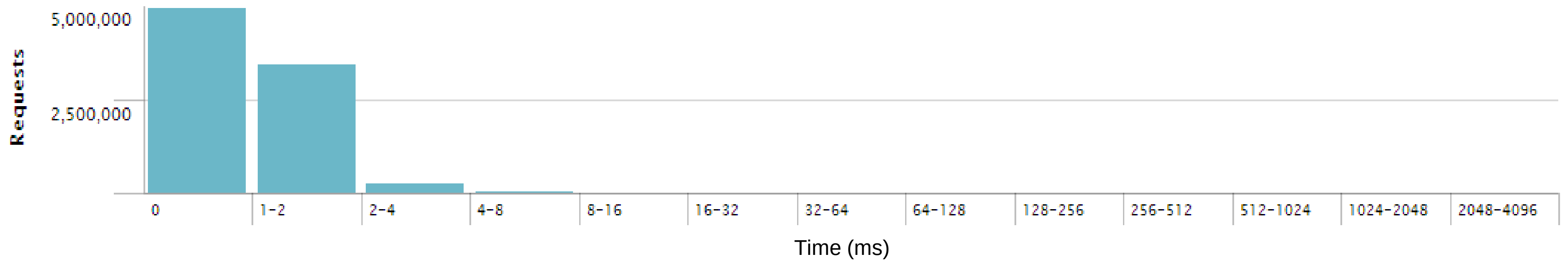
Graph Size	**27 million** nodes (pre-warmed in heap) ~**1x** properties, ~**2x** relationships
Traffic Volume	~**1000** req/min during biz hours ~ **40K** daily **real-time** updates
Performance	Avg: **1 ms**, 99% **< 4 ms**, 99.9% **< 9 ms**
JVM	Sun 6, 20 GB Heap (~15 GB pre-warmed) CMS GC, **No FULL GC** in prod Daily restarted for full database sync

Production Has Access Query

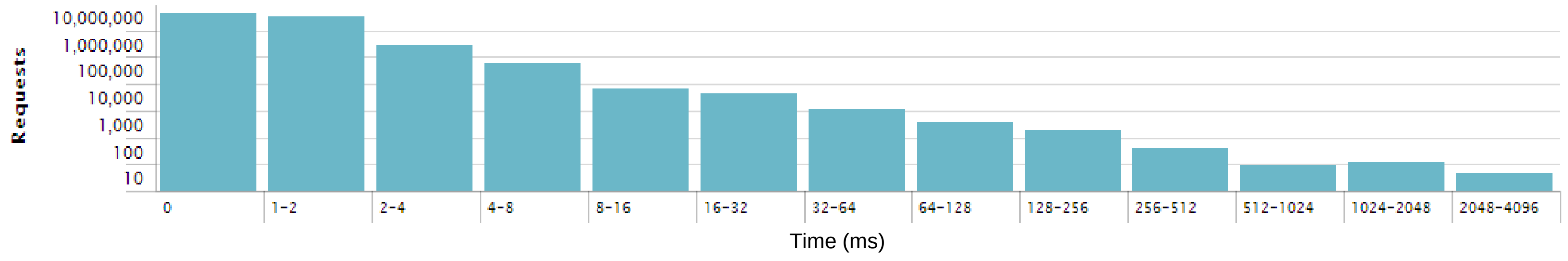
hasAccessToSubscription (last month)



hasAccessToSubscription (last 30 days)

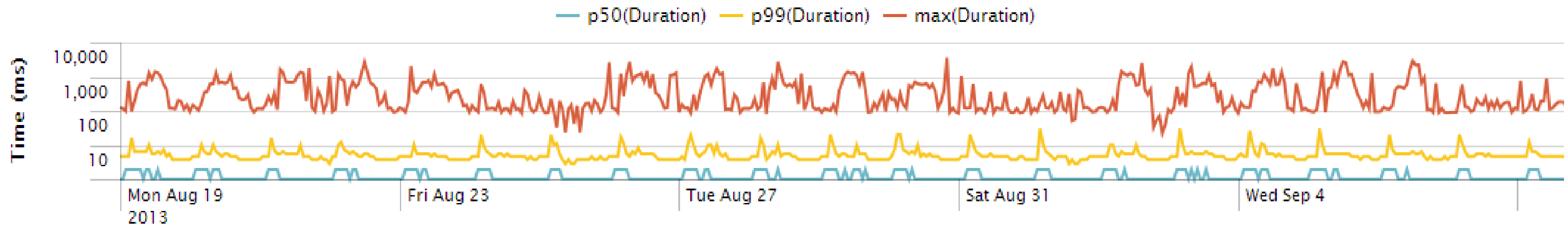


hasAccessToSubscription (last 30 days)

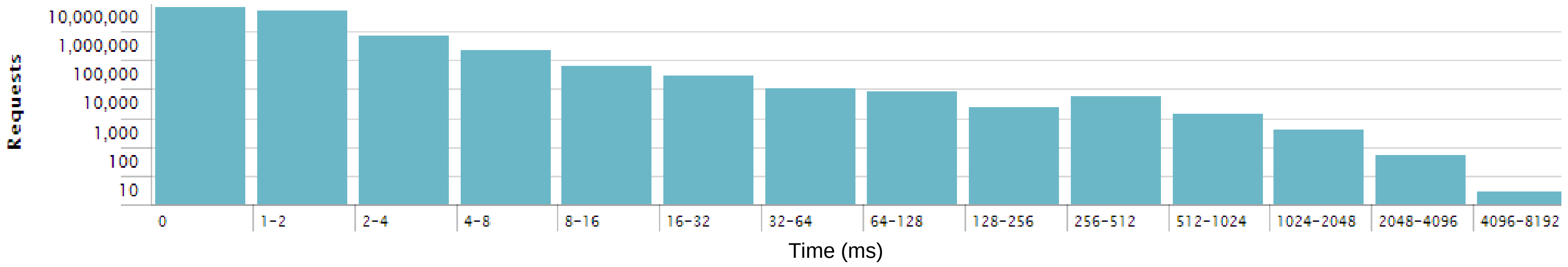


Production All Queries

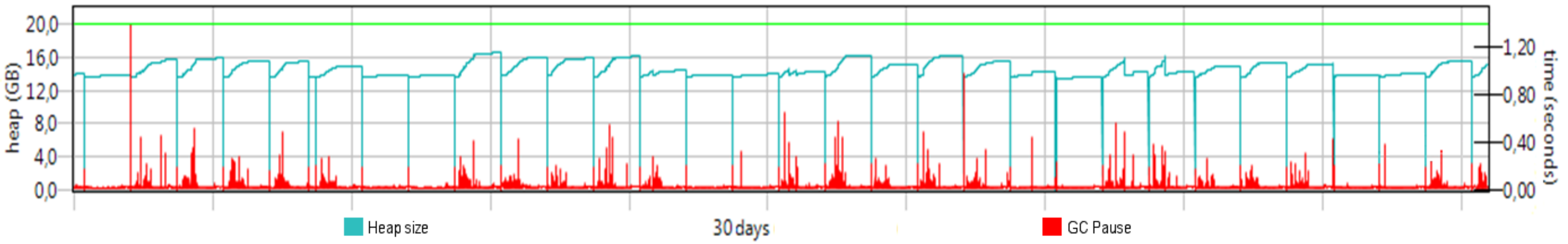
All graph requests (last month)



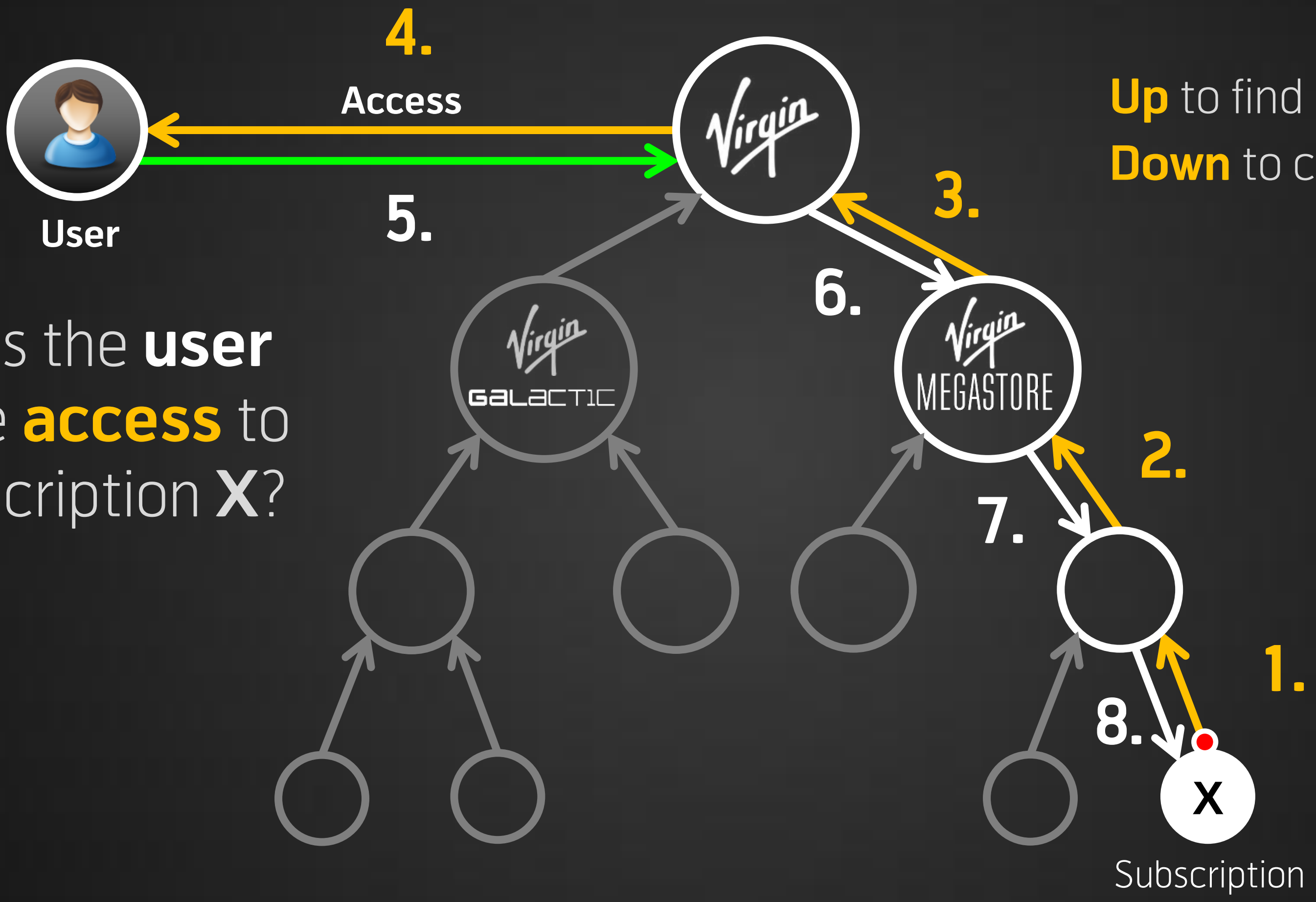
All graph requests (last month)



Garbage collection



U-Turn Strategy



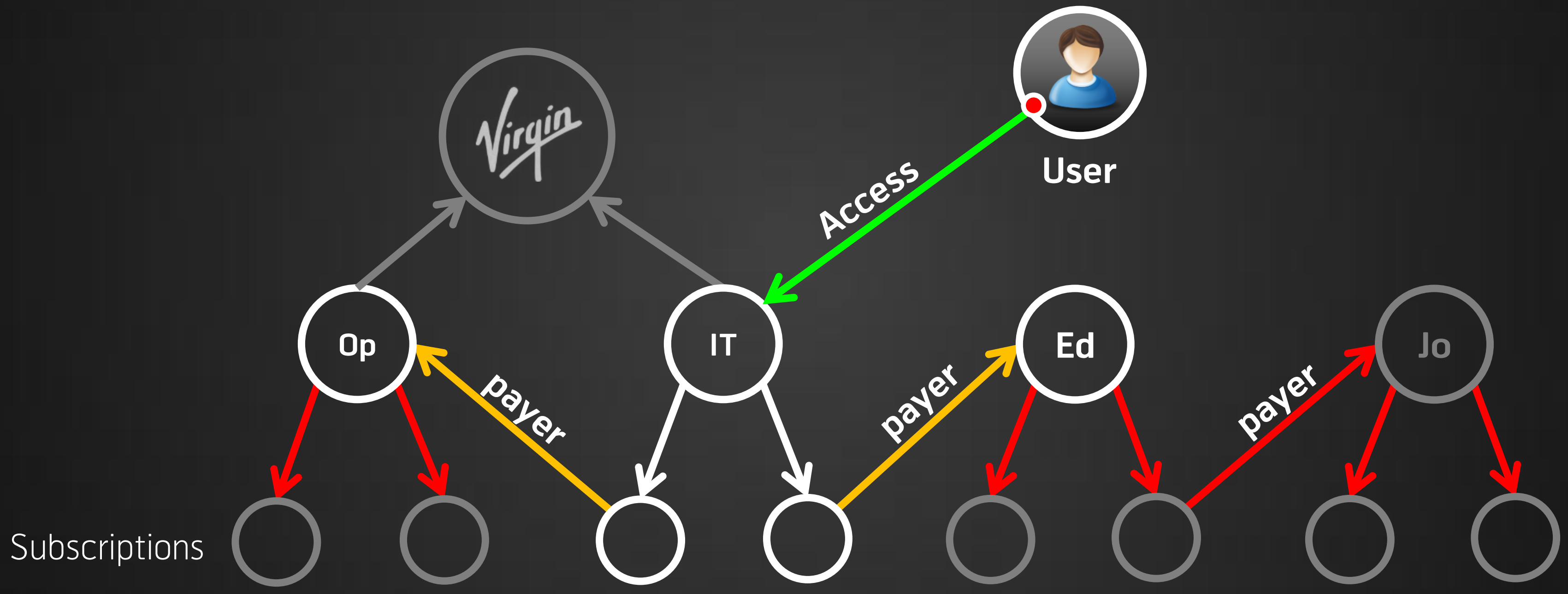
Up to find path **quickly**
Down to check **access**

Does the **user** have **access** to subscription **X**?

Reversing the traversal **increases performance** from **n/2** to **2d** where **n** and **d** are tree **size** and **depth** (we went from 1s to 1ms)

The Zigzag Problem

What if we also have **reversed access** to the subscription **payer**?

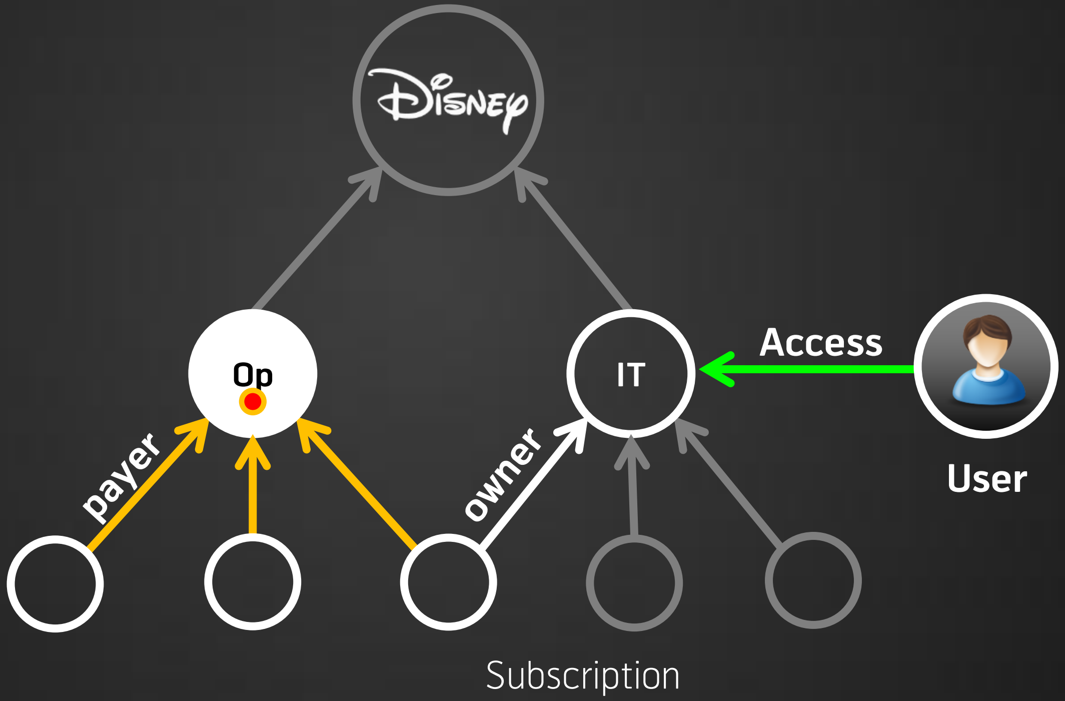


Solvable by adding **state** to the traversal (or check path)

The Many-to-Many Problem

The nodes **Op** & **IT** may be connected through **many** subscriptions

Does the **user** have **access** to department **Op**?



Traversal becomes **time consuming** (e.g. M2M market)

However, we only needed to implement the rule for **direct** access to sub.

Questions?

telenor