Chair of Network Architectures and Services
School of Computation, Information, and Technology
Technical University of Munich

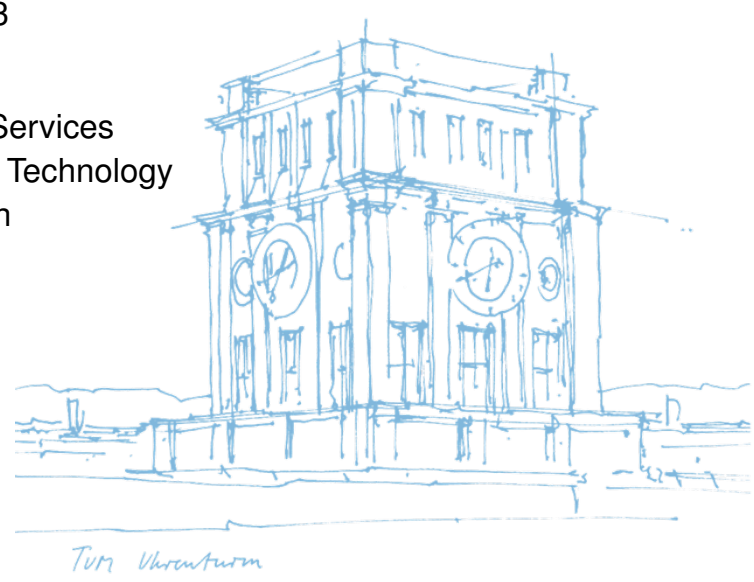
Development of a Framework for Retrieval of Parameters of the Starlink Dish

Final talk for the IDP by

Roberto Castellotti
advised by Leander Seidlitz, Johannes Zirngibl

Monday 27th November, 2023

Chair of Network Architectures and Services
School of Computation, Information, and Technology
Technical University of Munich



Starlink 101

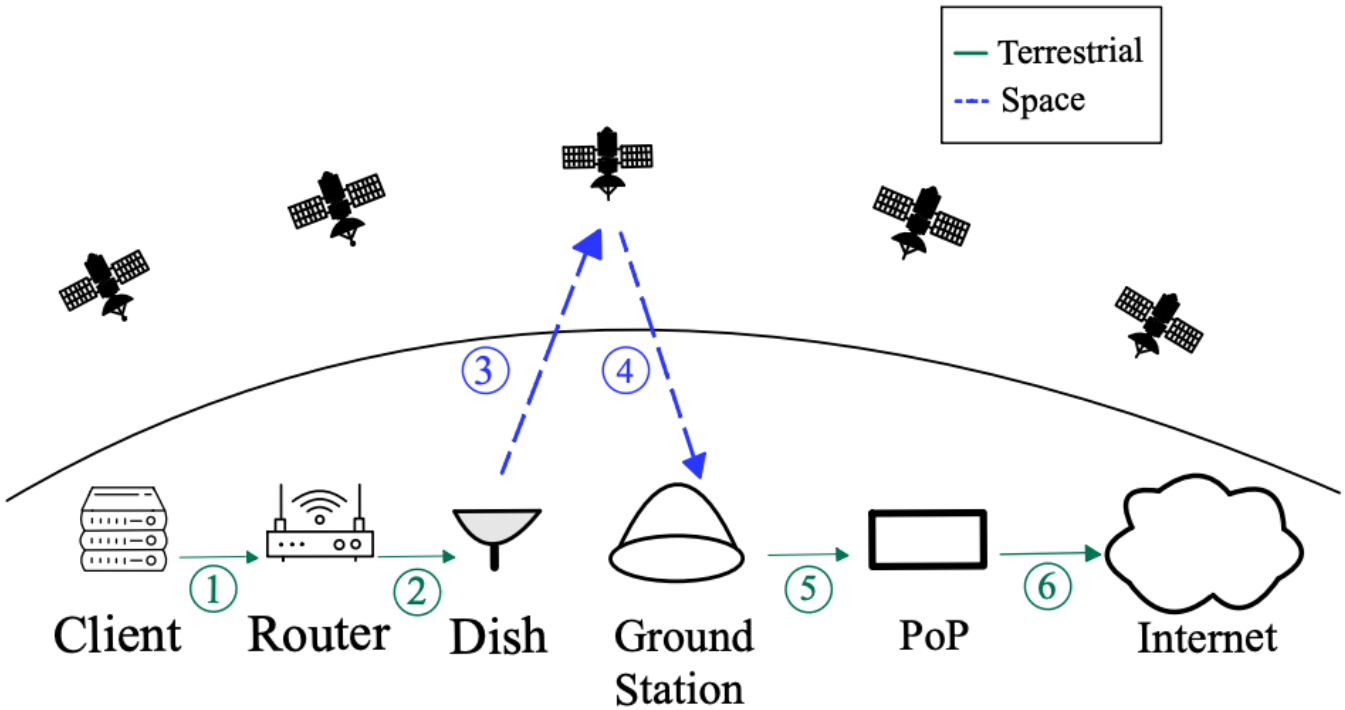


Figure 1: starlink in a nutshell (ignoring ISL), from [1]

Understanding routing decisions

- retrieved ip address blocks from major cloud providers (aws,azure,oracle), as we know their position [1]
- chose 5 geographically sparse targets around the globe (for aws: ap-northeast-2, us-east-1, ap-south-1, sa-east-1, me-south-1)
- tracerouted the targets over several days

[1] the fact we know the position doesn't really mean a traceroute to a certain address is really a traceroute to that geographic area

Understanding routing decisions

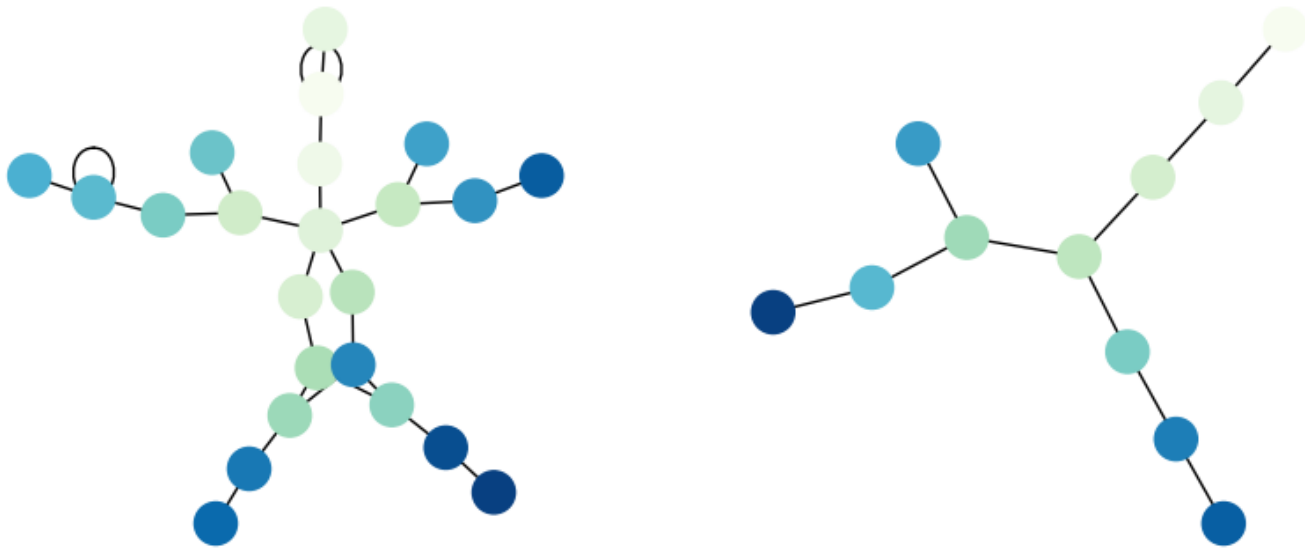


Figure 2: First 7 hops of traceroutes to 5 AWS datacenters using ICMP

Understanding routing decisions

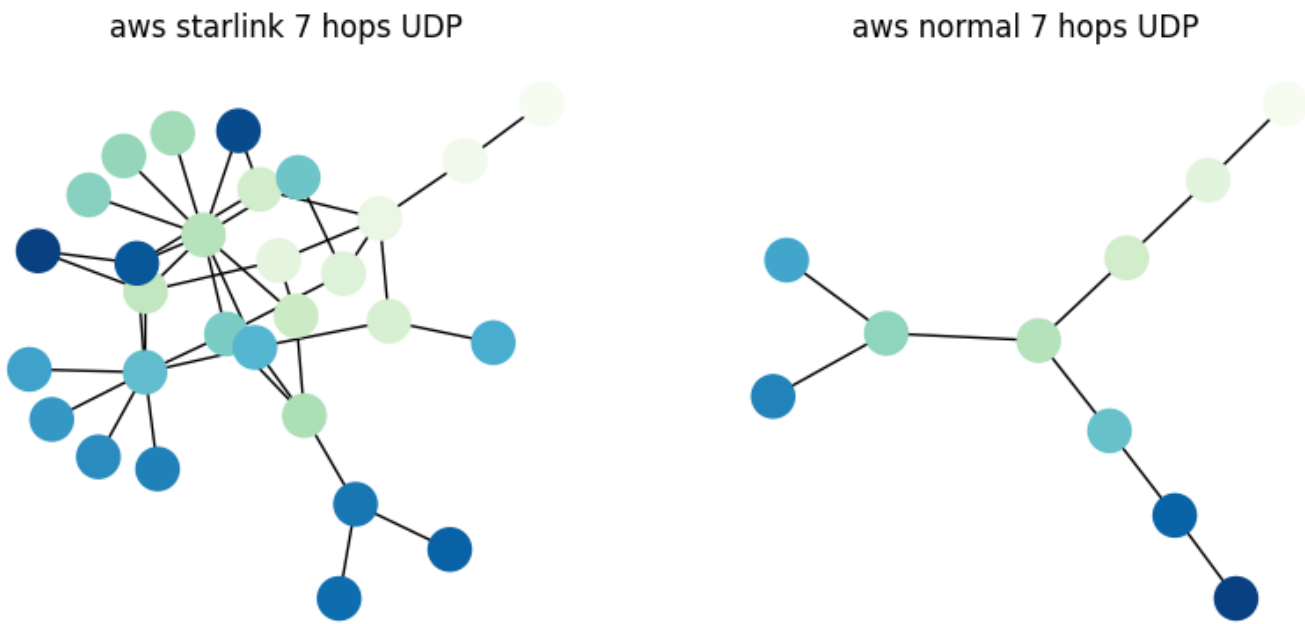


Figure 3: First 7 hops of traceroutes to 5 AWS datacenters using UDP

Understanding routing decisions

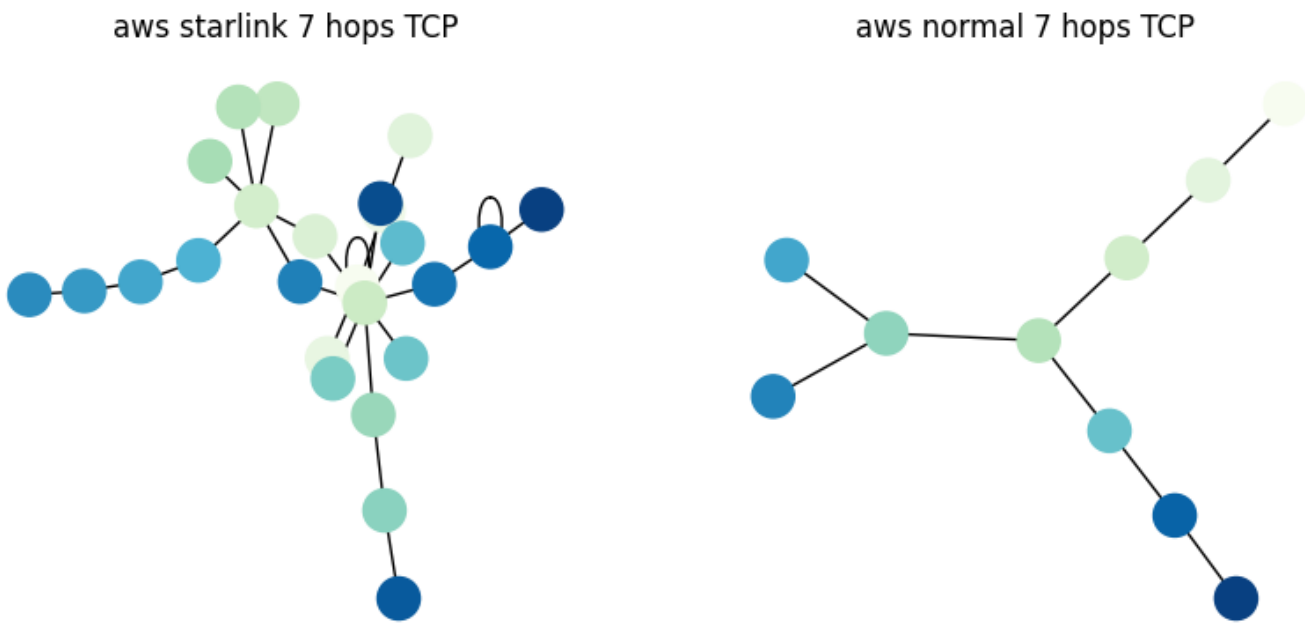


Figure 4: First 7 hops of traceroutes to 5 AWS datacenters using TCP

Measuring RTT changes when applying stress iperf

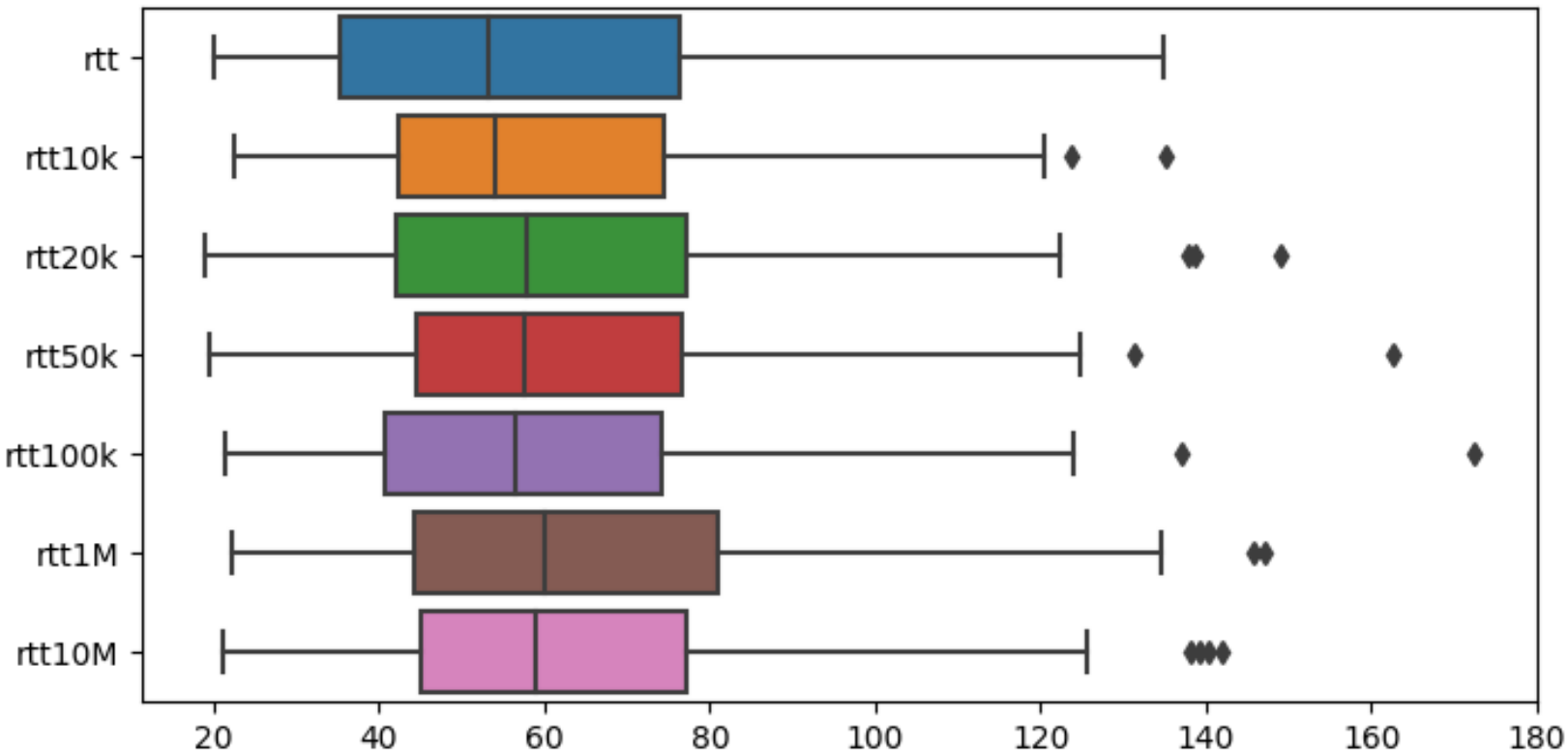


Figure 5: measuring RTT changes when applying stress iperf

Visualize Visible Satellites

- from celestrak.org we can download a list of Starlink's satellites TLEs
- A two-line element set (TLE) is a data format encoding a list of orbital elements of an Earth-orbiting object for a given point in time, the epoch. Using a suitable prediction formula, the state (position and velocity) at any point in the past or future can be estimated to some accuracy. (from wikipedia.org)

common.calculate_visible_satellites

```python
def calculate_visible_satellites(...):
# ...
satellites = load.tle_file(stations_url)
observer = Topos(observer_latitude, observer_longitude, observer_elevation)
t = load.timescale().now()

# Calculate satellite positions
positions = [(sat, (sat - observer).at(t)) for sat in satellites]

# Filter visible satellites
visible_satellites = []
for sat, position in positions:
    alt, az, distance = position.altaz()
    # Satellite is above the horizon
    if alt.degrees > 0 and distance.km < distance_km:
        visible_satellites.append((sat, alt, az))

return visible_satellites
```

Listing 1: visualizing a single obstruction map

count of visible satellites across time

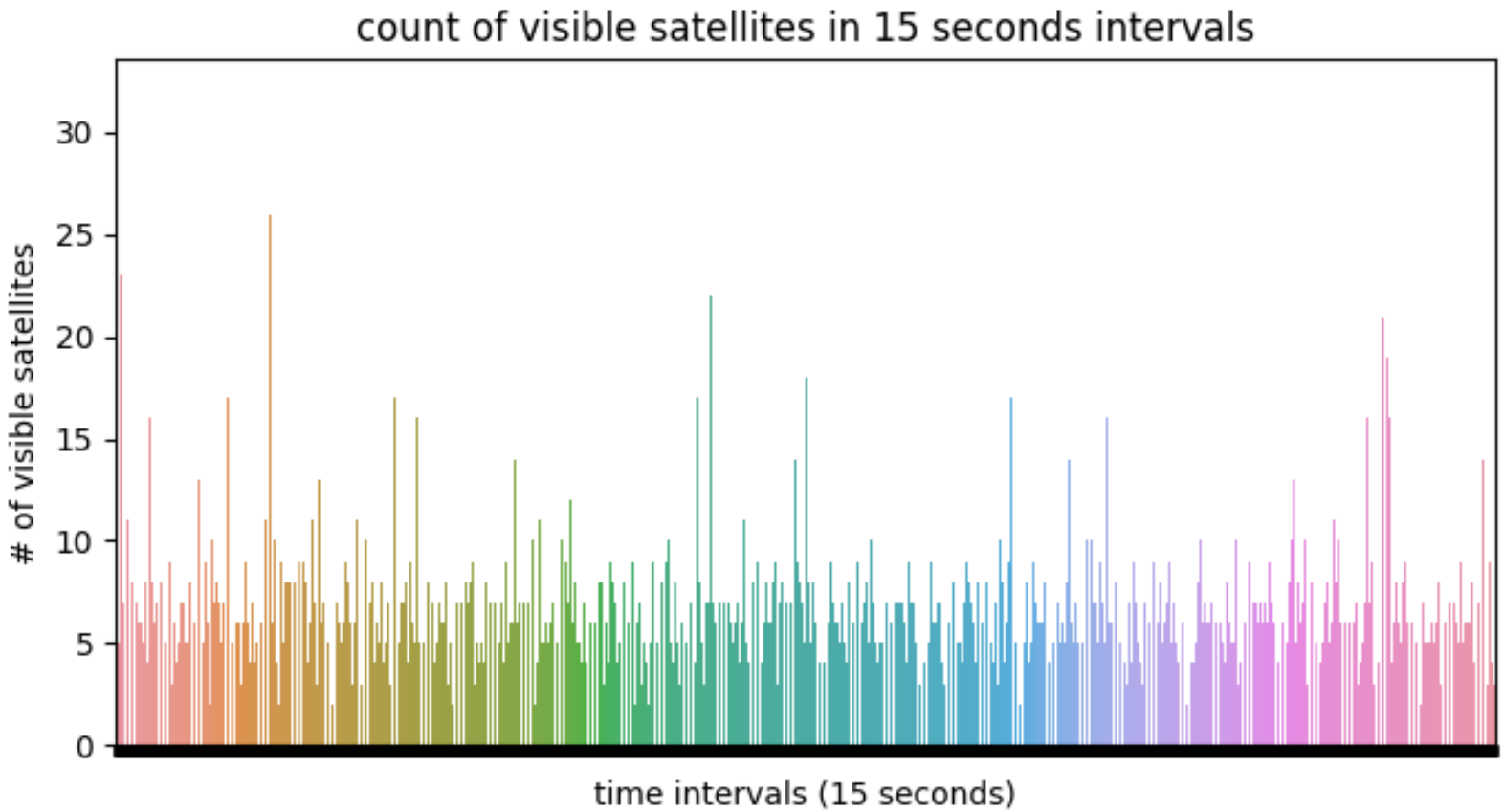


Figure 6: count of visible satellites across time

Visualizing Patterns in Visible Satellites

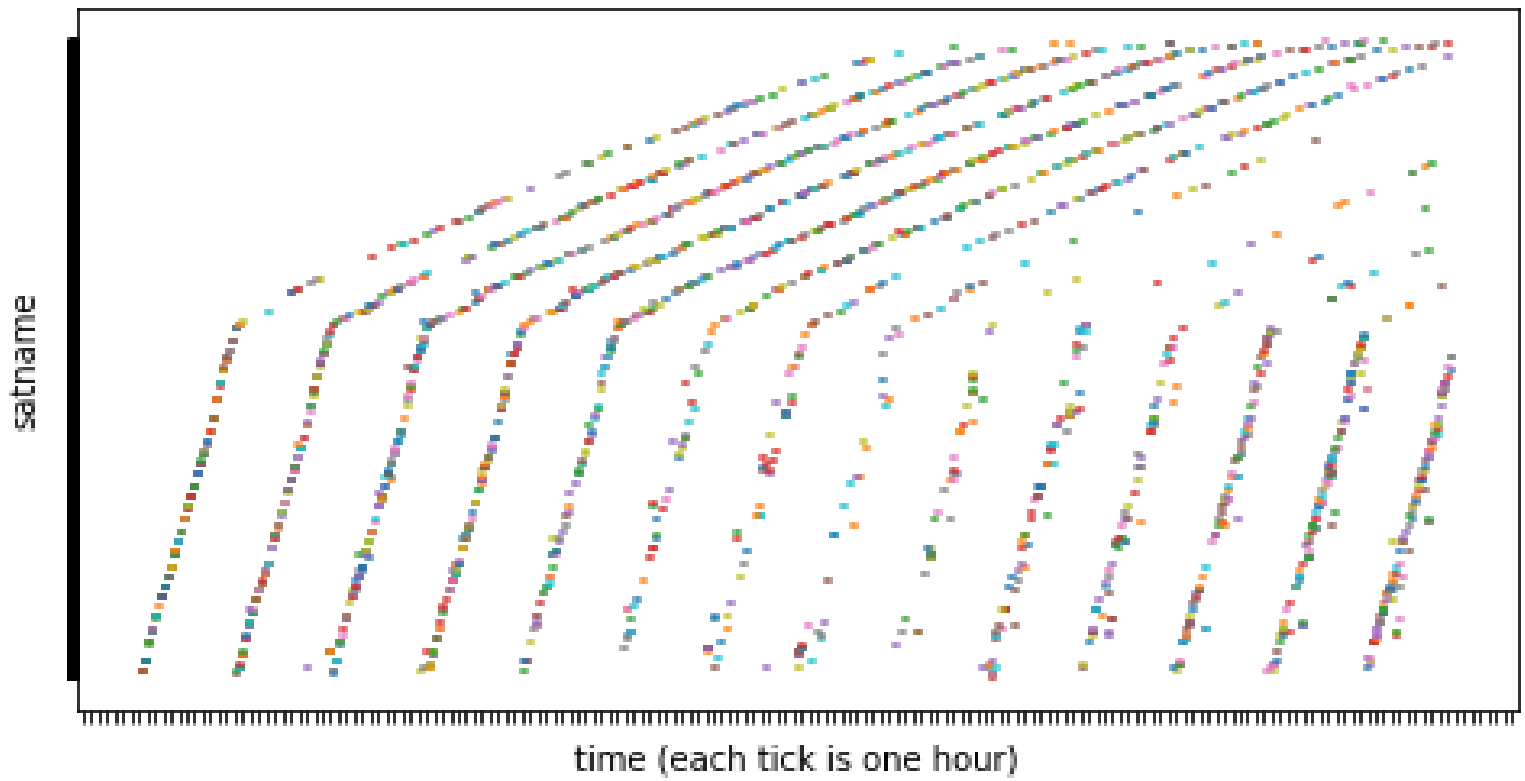


Figure 7: Visualizing patterns in visible satellites

the gRPC api

- the dish exposes a gRPC api with server reflection, "runtime construction of requests without having stub information precompiled into the client." [2]
- 55 "methods" are available, most of them don't work, we have 2 categories of errors: `Uninmplemented`, `PermissionDenied` and a couple of some other specific errors
- working methods: `reboot`, `get_status`, `start_dish_self_test`, `get_history`, `get_device_info`, `dish_power_save`, `dish_get_config`, `get_obstruction_map`
- to see all methods: https://gist.github.com/rcastellotti/e20630366dfeaeada6cc2680f562f6ac

[2] https://github.com/grpc/grpc/blob/master/doc/server-reflection.md

Next Actions

- investigate satellite handovers following the method described in [1] (we have a working script)
- try to correlate satellite handovers with sudden drops in bandwidth
- sneak peak: https://youtu.be/PjfMPr20suw

Bibliography

[1] L. Izhikevich, M. Tran, K. Izhikevich, G. Akiwate, and Z. Durumeric.
Democratizing leo satellite network measurement, 2023.