Title: Network topology detection for the configuration of real-time traffic (Python)

With *Time Sensitive Networking* (TSN) [1] Ethernet finally becomes vendor independently “real-time” capable (bounded latency, minimal jitter etc). Allowing new application scenarios in aerospace, automotive, manufacturing, transportation, and utilities, TSN holds promising assets for the future.

However, the mechanisms behind TSN increase the network complexity significantly and an extended configuration effort is needed to take “real-time” networks into operations. To automate the configuration process a central node was introduced, responsible for the configuration of the network devices (mainly bridges) to enable the TSN mechanisms. This central node needs global knowledge of the network topology (i.e. knowledge about the network participants and which links are between them) to configure the network devices accurately for the “real-time” traffic.

With the Link Layer Discovery Protocol (LLDP) there is already a well-established method to detect connected neighbors of a single network device. By reading these LLDP data from all network devices via a remote management protocol (*Simple Network Management Protocol (SNMP)* [3] or *NETCONF* [4]) a graph based representation of the network can be generated. With this representation the central node gains global knowledge about the network and can work with it.

The scope of this project is the implementation of this topology detection. This includes:

1) Simulate a network with multiple bridges running Link Layer Discovery Protocol (LLDP) and a remote management protocol (SNMP or NETCONF).
2) Read data from the simulated devices with a SNMP or NETCONF client in Python.
    a) starting from one known device, the client should (by reading the LLDP neighbor data of this first device) automatically traverse the network until it has found all devices and links.
3) Build a graph representation (nodes and edges) of the collected data using the Python library NetworkX.
4) Visualize network graph with NetworkX to verify your implementation.
5) Add a Flask Server to your implementation (serving Frontend)
6) Visualize network graph with Javascript library in Frontend
7) Create, update and delete operations for network devices and links
8) Form for stream requirements

Links:
[1] https://en.wikipedia.org/wiki/Time-Sensitive_Networking
[2] https://de.wikipedia.org/wiki/LLDP
[3] https://de.wikipedia.org/wiki/Simple_Network_Management_Protocol
[4] https://en.wikipedia.org/wiki/NETCONF
- https://omnetpp.org/
- https://www.riverbed.com/de/products/steelcentral/opnet.html (educational license available)
- https://ieeexplore.ieee.org/document/8374471
- https://avnu.org/knowledgebase/theory-of-operation/ p. 52ff (free - but name and email necessary for download)

Individual tasks of this project:
- Simulate network with devices and links.
- Implementation in Python (basic knowledge is enough)
- Traverse the network and read the LLDP data via SNMP or NETCONF
- Build a graph representation (nodes and edges) and visualize it.
- Evaluation
- Documentation

Requirements: Python, English or German.
Group size: 1/2
Project extension towards thesis: No
Name of supervisor: Frederik Lynker
E-mail of supervisor: frederic.lynker@fokus.fraunhofer.de