The Future of Model Based Sensor Management

Group Number: 107
Joar Heimonen, Christian Vu, Naly Keli
contact@joar.me
chvu002@student.kristiania.no
nake002@student.kristiania.no

May 15, 2025

Abstract

This document explores the future of model-based sensor management, focusing on the NETCONF protocol and YANG data models. We introduce Red-Netconf, a plugin for the event-driven visual programming language Node-RED, which implements NETCONF support. Alongside this, we present the NetconfAggregator plugin for Grafana, demonstrating how NETCONF can enable flexible, lightweight, and fault-tolerant sensor management solutions that are easy to configure. To validate these tools, we develop a reference implementation comprising two NETCONF-enabled thermometers, a NETCONF-enabled LED strip, and a server running Node-RED, Grafana, and the NetconfAggregator. Additionally, we analyze the security of NETCONF when used over the SSH protocol, highlighting its robustness and potential for secure network management.

Contents

1 Introduction

There are many paradigms of commercial sensor management and monitoring. Organizations use among other things PLC (programmable Logic Controllers), IoT devices and SCADA (Supervisory Control and Data Acquisition) systems to manage and monitor their sensors. For commercial use some of these alternatives are more popular than others. There are also a large amount of different higher level protocols like MQTT, HTTP and SNMP that can be used to manage and monitor sensors. We propose using the NETCONF protocol with YANG sensor models for management. This work will be done in collaboration with Lightside Instruments AS.

This document will cover the following three topics:

- **Work Methodology:** An indept analysis of the knowledge base around work methods like Scrum, Kanban, and Waterfall. With a focus on how our work methodology differs from these.
- **NETCONF and YANG sensor management**: A qualitative analysis of the NETCONF and YANG protocols and how they can be used to manage sensors and a description of our work with Lightside Instruments AS on NETCONF and YANG sensor management.
- **NETCONF Security**: A qualitative analysis of the security aspects of the NETCONF protocol.

2 Lightside Instruments AS

Lightside Instruments is a company specializing in developing instruments with model based network management for use in networking, network interconnect testing and telemetry. They design their instruments with YANG (RFC7950) [6] models and NETCONF (RFC6241) [14] protocol. The instruments are based on IETF standards and drafts, and are implemented with software tools available in Debian, programmable logic and open hardware [27].

3 Technical background

The following are terms and technologies that are essential to understanding this article.

3.1 NETCONF and YANG

NETCONF [14] is a model based Network Configuration Protocol. Each NETCONF device presents the acquiring device with a YANG [6] data model consisting of the device state and parameters. Each data model has a set of constraints making them error correcting.

3.2 Yuma123

Yuma123 [45] is an open source NETCONF and YANG implementation. Yuma123 is available as a Debian package and is maintained by Lightside Instruments AS.

3.3 NodeJS

Node.js [32] is a JavaScript runtime built on Chrome's V8 JavaScript engine. It is used to build JavaScript applications that can run independently of a web browser. Node.js is popular for building server-side applications as it is relatively performant and it allows developers to work with the same language on both the client and server.

3.4 Node-Yuma123

Node-Yuma123 [31] is a NodeJS package that implements a set of Yuma123 bindings. It allows for efficient NETCONF and YANG development in NodeJS.

3.5 Node-RED

Node-RED [29] is an open source low code programming tool for event driven applications. It is developed by IBM and is based on Node.js [32]. Node-RED is used to connect hardware devices and APIs through a visual programming interface.

3.6 Grafana

Grafana [20] is an open source data visualization tool. It is used to visualize arbitrary data from different data sources.

3.7 XML and XPATH

XML stands for Extensible Markup Language [15]. It is a markup language that is used to encode data in a format that is both human and machine readable. XPATH [48] is a language for navigating XML documents. An XPATH expression allows for the selection of arbitrary nodes in an XML document.

3.8 Scrum

Scrum [22] is a framework for agile [5] software development. The term Scrum is derived from the game of rugby, where a scrum is a way of restarting play after a minor infringement [39].

3.9 Kanban

Kanban [24] is a framework for agile software development. The term Kanban is derived from the Japanese word for "signboard" or "billboard" [25].

3.10 Waterfall

Waterfall [47] is a framework for software development. The waterfall model is a linear and sequential approach to software development.

3.11 Extreme Programming

Extreme Programming [16] is a framework for agile software development. Extreme Programming takes the best practices of software development and takes them to the extreme, hence the name.

4 Work Methodology

4.1 Research Question

This review examines the claims that Scrum, Kanban, Waterfall, Extreme Programming and DevOps increases worker productivity substantiated by empirical evidence.

4.2 Scoping Review

4.2.1 Search strategy

The following is our search strategy for the scoping review. We will be searching for quantitative studies on the efficiency of the following work methodologies:

- Scrum
- Kanban
- Waterfall
- Extreme Programming
- DevOps

We will be searching the following databases:

- IEEE Xplore [23]
- ACM Digital Library [3]
- Google Scholar [19] (Meta database)

We will also be searching the following industry websites:

- Agile Alliance [4]
- Scrum.org [22]
- DevOps Institute [33]
- Scrum Alliance [40]

Our search will consists of a set of primary and secondary keywords. The primary keywords are:

- Scrum
- Kanban
- Waterfall
- Extreme Programming
- DevOps

The secondary keywords are:

- Effectiveness
- Efficiency
- Productivity
- Performance
- Success
- Failure

The search will be done using the following search string:

```
(Scrum OR Kanban OR Waterfall OR "Extreme Programming" OR DevOps)
                                    AND
(Effectiveness OR Efficiency OR Productivity OR Performance OR Success OR Failure)
```

4.2.2 Inclusion Criteria

The scoping review will include articles meeting the following criteria:

- Published after January 1, 2020
- Published in English
- Relevant to the research question
- Empirical evidence

- Quantitative studies
- 20 first results from each database
- Evaluating the effectiveness of the following methodologies:
  - Scrum
  - Kanban
  - Waterfall
  - Extreme Programming
  - DevOps

4.2.3 Result

After applying the inclusion criteria to a set of 60 articles, we discovered that 5 of them were duplicates. The 55 remaining articles were screened by title and abstract, resulting in 12 articles being excluded. The 43 remaining articles were assessed for eligibility, resulting in 42 articles being excluded. Only one article was assessed to be eligible for this review.

4.2.3.1 PRISMA flow diagram

Figure 1 shows the PRISMA [36] flow diagram for the scoping review. The PRISMA flow diagram is a standardized way of reporting the results of a scoping review.

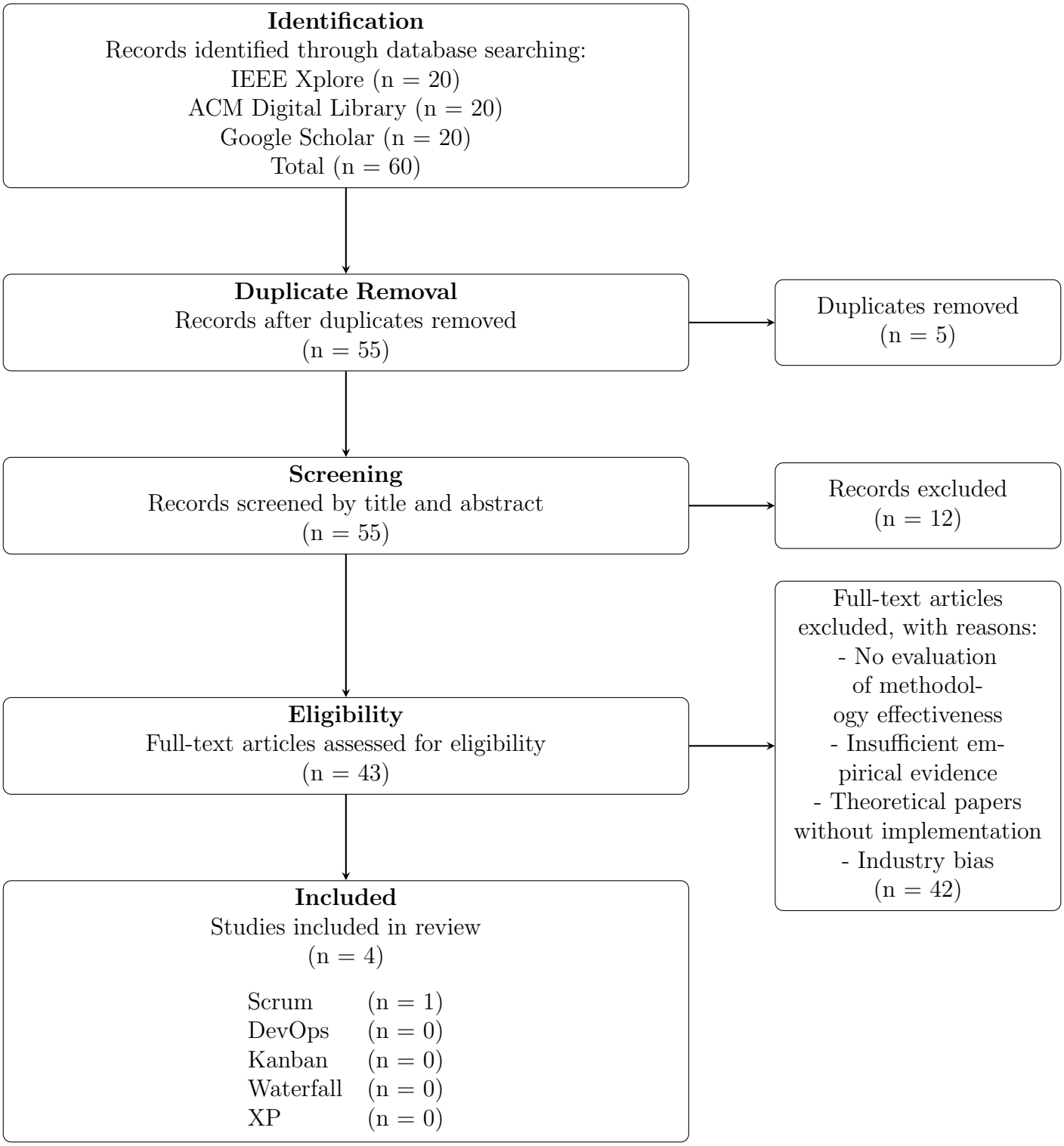

Figure 1: PRISMA flow diagram for scoping review of software development methodologies

4.3 Scrum

Scrum is a framework for agile software development. The term Scrum is derived from the game of rugby, where a scrum is a way of restarting play after a minor infringement [39]. The use of the term Scrum in software development was first introduced by Takeuchi and Nonaka in 1986 in a paper titled "The New New Product Development Game" [43]. In the paper, the authors argue for a new approach to product development where the different stages of development are overlapped, rather than sequentially executed in a "pass the baton" fashion. This differs from the then popular NASA type PPP (Phased Project Planning) model [35].

Modern Scrum development consists of a set of sprints, these sprints consists of a pre-defined set of tasks that are to be completed in the pre-defined sprint time frame. Each task or "story" is assigned an arbitrary number of points that represents the complexity of the task. The sprint is the completed when there a no more points to be completed or the time frame is up. There are many modern flavors of Scrum, like Accenture's [2] Autoscrum which is a scrum framework that was first introduced in the talk "AGILE TRANSFORMATION? FOR COMPLEX SYSTEMS? ...NO WAY!" by Brehm [7] as can be seen in *Figure 2*. Or the scaled agile framework (SAFe) [18] which is a framework developed by Scaled Agile Incorporated which introduces SAFe Scrum see *Figure 3*. Or the Deloitte's [12] "The Agile Landscape v3" that consists of all the different frameworks and methods used for project management. See the Scrum section in *Figure 4*.

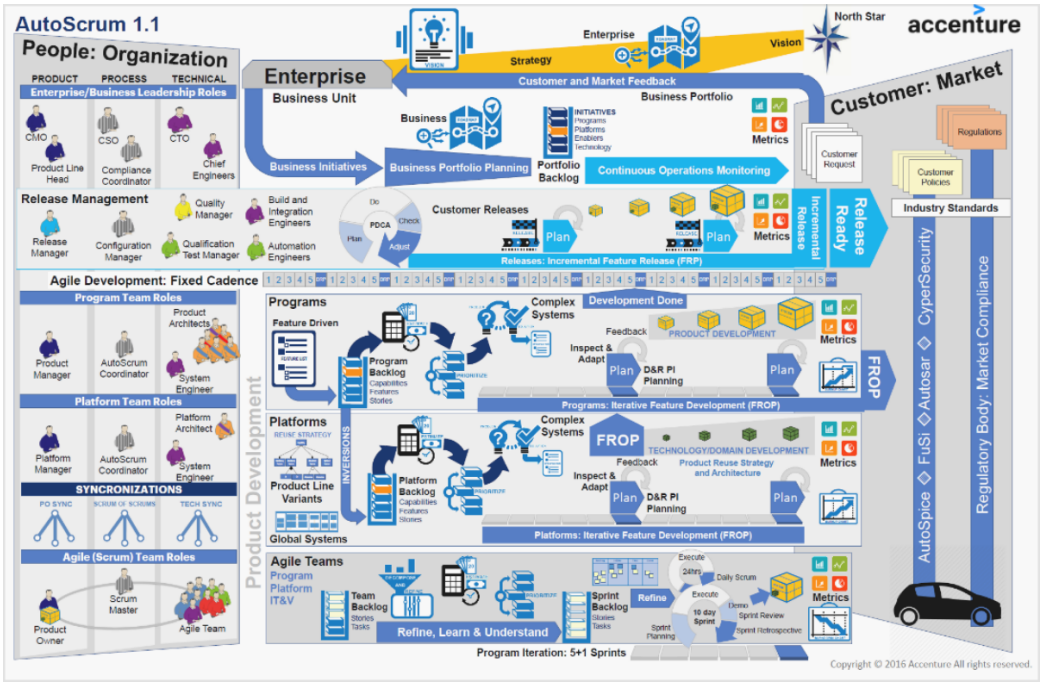

Figure 2: Excerpt from the presentation "AGILE TRANSFORMATION? FOR COMPLEX SYSTEMS? ...NO WAY!" by Brehm [7] showing the Autoscrum framework.

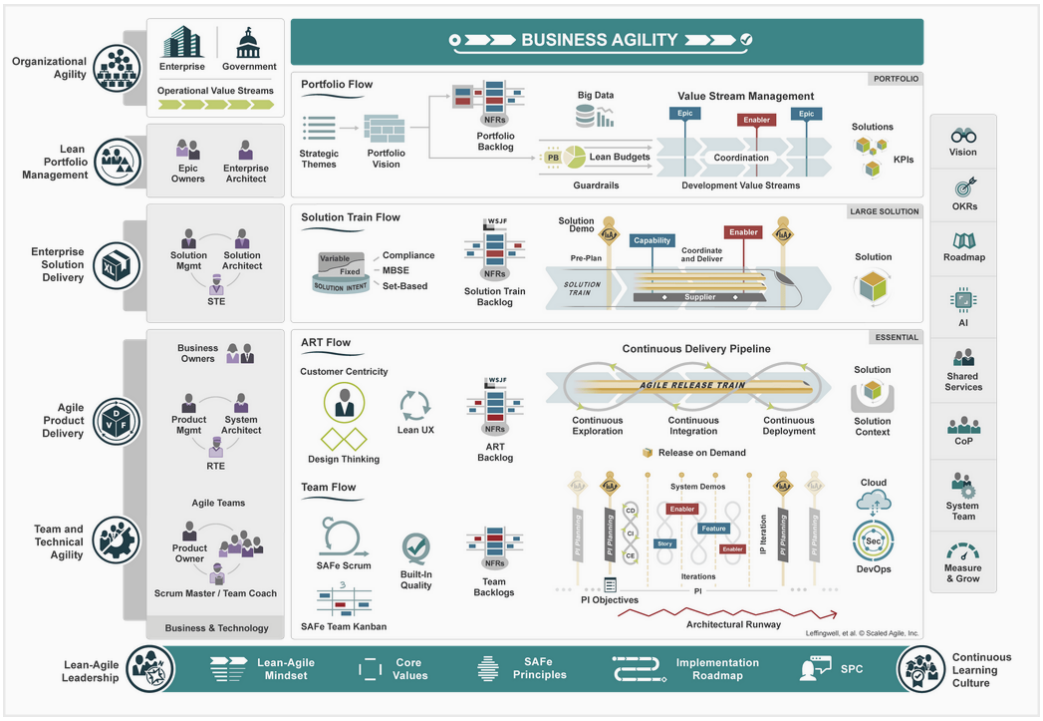

Figure 3: The Scaled Agile Framework (SAFe) [18].

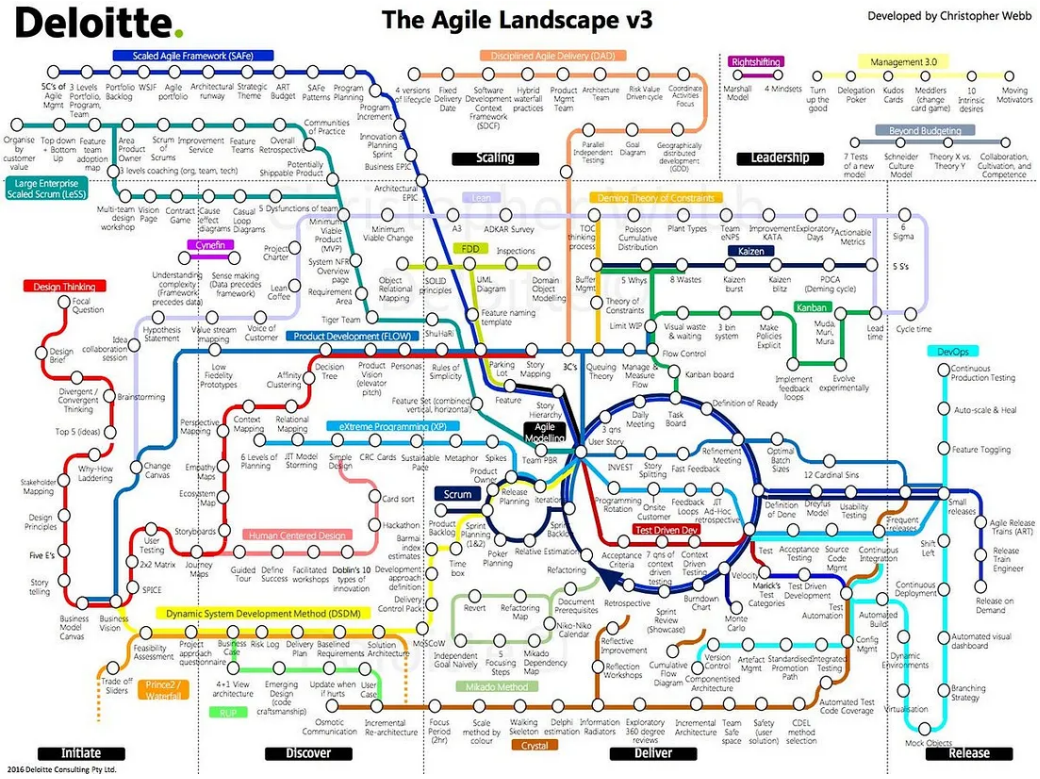

Figure 4: The Agile Landscape v3 [12] showing the different frameworks and methods used for project management.

4.3.1 Evidentiary foundation

With only one article "A Theory of Scrum Team Effectiveness" by Verwijs and Russo [46] we can conclude that there is no empirical evidence that Scrum increases worker productivity or efficiency. Whilst the findings of the article seems promising, and the article is thorough, there simply is not enough research on the topic to comment on the evidentiary foundation of Scrum, this is something taken in too account by the authors of the article. Only a handful of other articles have been published on the topic, but most of them fall outside the exclusion criteria of the scoping review as they have been published before 2020.

4.4 Kanban

Kanban is a framework for agile software development. The term Kanban is derived from the Japanese word for "signboard" or "billboard" [25]. A Kanban board is a visual representation of the tasks that need to be completed, the board consists of multiple columns that represent the different stages of the development process. As a task is worked on it traverses the columns of the board until it is completed. This provides a visual representation of the state of the project and allows for easy identification of bottlenecks in the process. Kanban is a much more flexible and lean approach to project management than Scrum, as it follows closer to the agile manifesto's principles of flexibility and adaptability [30].

4.4.1 Evidentiary foundation

With no articles found in the scoping review we can conclude that there is no empirical evidence that Kanban increases worker productivity or efficiency.

4.5 Waterfall

The waterfall [47] is a linear and sequential approach to software development. Sets of tasks are grouped into phases, where each phase must be completed before the next phase can begin. This is reminiscent of the NASA type PPP (Phased Project Planning) model [35]. The waterfall model mirrors the traditional problem solving process, of breaking a problem down into a set of smaller problems, and solving each of the smaller problems in a

sequential manner. This is something that is found at the core of all project management methodologies.

4.5.1 Evidentiary foundation

With no articles found in the scoping review we can conclude that there is no empirical evidence that Waterfall increases worker productivity or efficiency.

4.6 Extreme Programming

Extreme Programming [16] is a framework for agile software development. Extreme Programming takes the best practices of software development and takes them to the extreme, hence the name. A core part of Extreme Programming is the use of pair programming [34], where two developers work together on the same code.

4.6.1 Evidentiary foundation

With no articles found in the scoping review we can conclude that there is no empirical evidence that Extreme Programming increases worker productivity or efficiency.

4.7 DevOps

DevOps is the combination of development and operations, it is the practice of combining software development and IT operations. This keeps the developers close to the day to day operations of the software they are developing. A result of developer operations is the use of automated continues integration and continues deployment (CI/CD) [10] systems as the responsibility of the deployment falls on the developers. This methodology fosters early error detection and correction.

4.7.1 Evidentiary foundation

We did not manage to find any articles that assess the effectiveness of DevOps versus other paradigmes, but. DevOps and Research Assessment (DORA) [13] is a research department at Google that focuses on researching assessment methods for DevOps. They publish a yearly report on the state of DevOps. Other than DORA independent research into the topic is scarce

and whilst research is being done there is currently not a strong enough evidentiary foundation to make any claims about the effectiveness of DevOps.

4.8 Conclusion

The scoping review has made it clear that there is a lack of research into the effectiveness of different work methodologies. The industry seems focused on researching performance metrics for each methodology, rather than the effectiveness of the methodology itself.

4.9 Our Work Methodology

At Lightside Instruments we work with both hardware and software in a small team. This workflows lends a great deal of flexibility to our work. With most engineers doubling as both hardware and software engineers, we have found it cumbersome to use a strict work methodology like most agile frameworks. Due to the small size of our team we have adapted a DevOps like approach that is tailored for our needs. We have a set of tasks that are defined in a git repository. After a task is completed a pull request is submitted and the task is marked as pending review. When the request is approved the task is marked as completed and the code is merged into the main branch. We not only use automated continues integration and continues deployment (CI/CD) systems for our software, but also for our hardware. We have developed a set of workflows that are used to automatically generate the production files for our hardware.

Whilst there is no empirical evidence or any research suggesting that our work methodology is more effective than any of the other methodologies, we have found it to be a good fit for our needs. And we feel that it adheres close the principles set forth by the agile manifesto [30].

4.10 Observations And Outcomes of our Work Methodology

During the past six months working on this project we experienced several cases of hardware failure. When a case of hardware failure were severe enough to warrant action on our part we called this a "side quest". When a side quest was initiated, the team would temporarily shift focus to address the issue. These side quests often involved debugging hardware, replacing faulty components, or reconfiguring systems to ensure proper functionality. While these

tasks were not part of the original project scope, they provided valuable insights into the robustness of our designs and workflows. For example, during one side quest, we discovered that the Wi-Fi chipset on one of our Raspberry Pi zero 2w's had failed. Due to our decision to solder this Raspberry directly onto our ethernet4pizero-poe card we decided to try and de-solder the broken Wi-Fi chipset instead of re-building the whole prototype. These side quests proved to be an invaluable way of keeping productivity up without shifting focus to unrelated tasks.

Taking on side quests would not have been possible without the flexibility of our work methodology. These are issues that in some cases would have gone unsolved if we had weekly targets to meet or a rigidly designed development plan. During the duration of the project there were plenty of times when side quests led to changes in the project plan and even new projects in itself. A great example of this was the development of easyNetconf after having developed the initial version of Node-Yuma123 we discovered that it quickly became cumbersome to work with due to the object-oriented nature JavaScript and the functional nature of C.

5 NETCONF and YANG sensor management

Hardware management is an essential part of administrating a larger network. Together with Lightside Instruments AS we have developed open source tools for YANG and NETCONF aimed at hardware sensor management. We have built a reference implementation of a NETCONF system that tackles this issue. Our reference implementation consists two NETCONF temperature probes, a data aggregator, a data visualization tool and a tool for quick management of NETCONF devices.

The following sections will describe the project and how each of the components are implemented. An illustration of the architecture of the project can be seen in *Figure 5*.

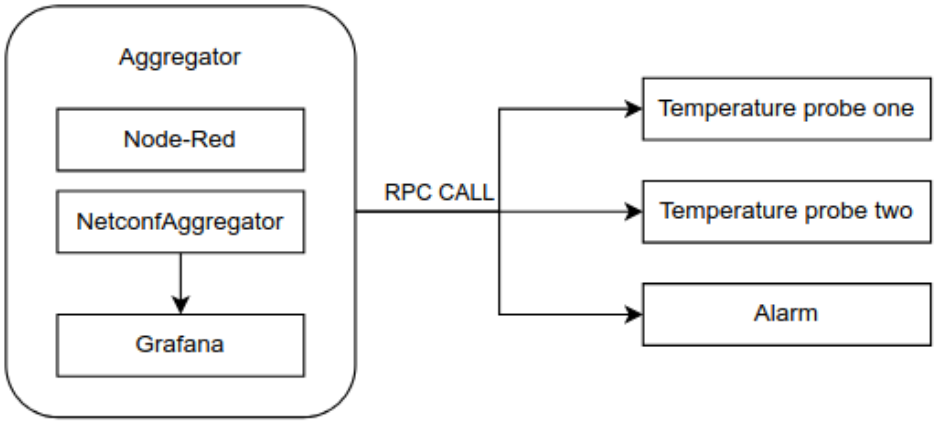

Figure 5: Architecture of the NETCONF and YANG sensor management system.

5.1 NETCONF

NETCONF [14] is a mode based Network Configuration Protocol implemented by the IETF (Internet Engineering Task Force) in the RFC (Request for Comments) 6241 [14].

5.2 YANG Model

The YANG model is a model used to describe the state and actions of a NETCONF device. The Internet Engineering Task Force (IETF) has developed a set of standard YANG models for NETCONF devices. For the purposes of this project we will not be using the standard YANG models, but instead we will be using a custom YANG model developed by Lightside Instruments AS that only describes the state of thermometers. See *Figure 6* for the YANG model.

The YANG model consists of a container "thermometers" that can contain multiple this container, contains a list of thermometers where each thermometer contains two leafs one containing the thermometer name and one containing the temperature in the form of a 32-bit integer that is limited from -27315 degrees Celsius, the field does not support floats and the value

must therefore be divided by one hundred after being read. The max value in the model represents the max value of a 32-bit integer where the most significant bit is used as the sign bit.

```
module lsi-thermometers {
  yang-version 1.1;
  namespace "urn:lsi:params:xml:ns:yang:thermometers";
  prefix thermometers;

  organization  "Lightside Instruments AS";

  description
    "Thermometers monitoring module.";

  revision 2022-07-25 {
    description
      "Initial version.";
  }

  container thermometers {
    config false;
    list thermometer {
      key "name";
      leaf name {
        type string;
      }
      leaf value {
        description
          "Temperature in degrees Celsius multiplied by 100.";
        type int32 {
          range "-27315..max";
        }
      }
    }
  }
}
```

Figure 6: YANG model for thermometer management

5.3 Node-RED

Node-RED is a low code programming tool for event driven applications. It makes it possible to create arbitrary flows that function as a compatibility layer between different systems and protocols. The low code nature of Node-RED makes arbitrary system integration accessible even for the lay person.

5.3.1 Red-Netconf

The development of Red-Netconf [1] arose from the need for an orchestrator to manage interactions between multiple devices. Node-RED was chosen as the orchestrator due to its extensive plugin repository, which supports a wide range of protocols and software. This flexibility makes Node-RED an ideal choice for integrating NETCONF devices with any system. Enabling arbitrary interactions between any device. Red-Netconf [1] implements the following two nodes:

- **Netconf Session**: This node is used to create a NETCONF session with a NETCONF device.
- **Netconf Yangcli**: This node is used to send NETCONF Remote Procedure Call (RPC) to a NETCONF device using yangcli commands.

Using these two nodes we are able to create Node-RED flows that can manage NETCONF devices.

5.3.1.1 Temperature alert

As an example of how the Red-Netconf nodes can be used we created a Node-RED reference flow that collects data from a thermometer and switches on an LED when the temperature is above 25 degrees Celsius, this can be seen in *Figure 7*.

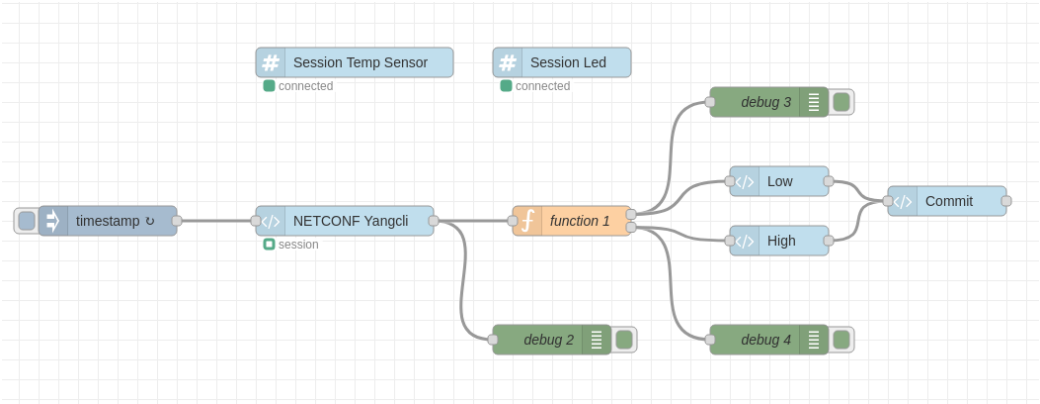

Figure 7: Node-RED flow using Red-Netconf nodes that monitors a temperature sensor and switches on an LED when the temperature is above 25 degrees Celsius.

5.3.1.2 Rover control

Another example is the control system for our rover. The rover is a Roomba vacuum cleaner controlled over rs232 by a Raspberry Pi zero. It presents a NETCONF interface that it can be controlled with this flow can be seen in *figure 8*.

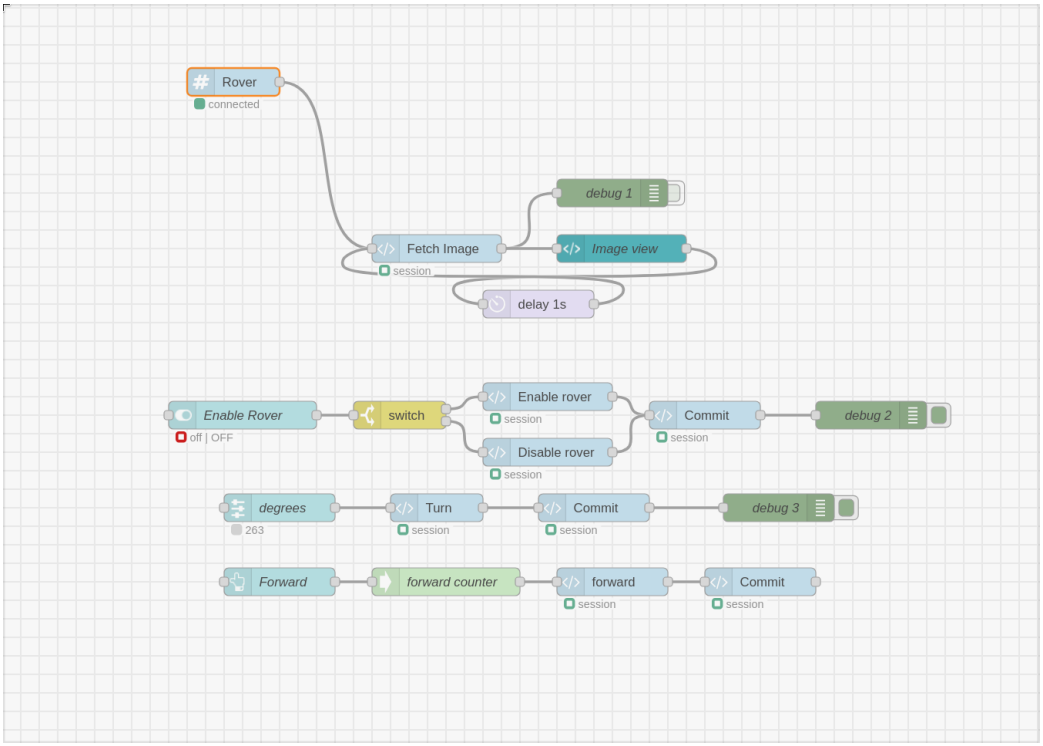

Figure 8: A flow for controlling our NETCONF rover.

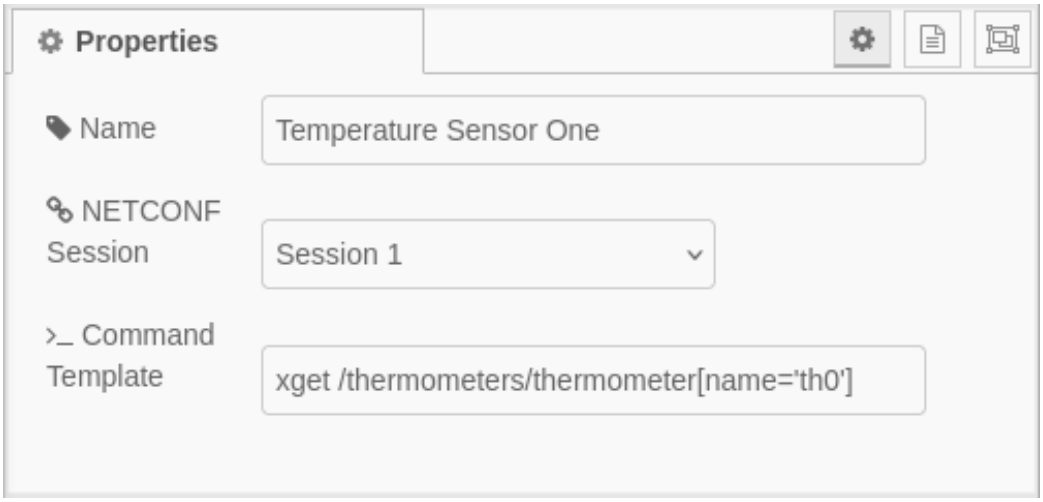

Figure 9: The properties of a Red-Netconf Yangcli node

Yangcli [49] implements a set of commands for interacting with any NETCONF server over ssh. Using yangcli commands in the nodes makes them extremely versatile allowing for a node to perform any Remote Procedure Call. The yangcli commands are translated into Remote Procedure Calls and send to the specified server. An example of a configured node can be seen in *Figure 9*.

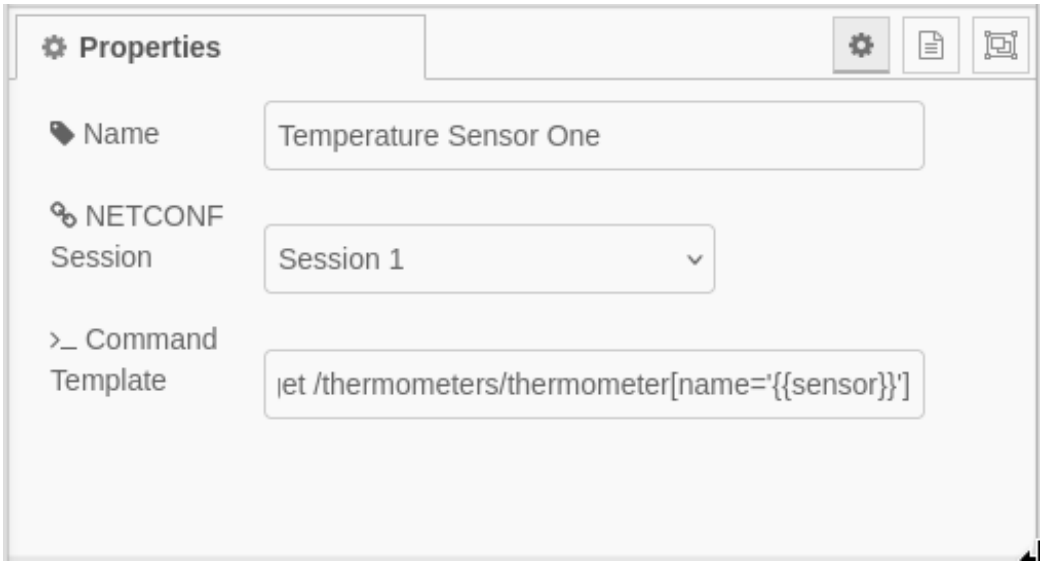

Figure 10: A yangcli-node with a mustache style template for the sensor name

The node also supports Node-RED's mustache style templates, using these templates we are able to access data passed into the node inside our yangcli commands. *Figure 10* shows how we dynamically can fetch sensor data from a device containing multiple temperature sensors using the following yangcli command *xget /thermometers/thermometer[name='*{{*sensor*}}*']* note that instead of using a static name as seen in *figure 9* we are using the mustache style template {{*sensor*}}

5.3.2 Node-Yuma123

Yuma123 is an implementation of the NETCONF standard that is written in C. It consists of a NETCONF server written in C, an RPC (Remote Procedure Call) client and a yangcli interpreter. The yangcli interpreter is a command line tool that can be used to interact with NETCONF devices using YANG models and yangcli commands. yangcli translates the yangcli commands into NETCONF RPC calls and sends them to the NETCONF device. Yuma123 also presents an API that can be used to interact with the NETCONF server called libyuma-dev.

Node-Yuma123 is a set of NodeJS bindings for the Yuma123 API. It is written in C++ and created to resemble the original C API as closely as possible. Node-Yuma123 also implements some new functions like asynchronous connections. Implementing this was quite a challenge as the libyuma-dev API is not thread safe. Instead of running each connection in a new thread we stack the instructions using libuv [26] and run them in the main thread. This resolves most of our threading issues, but some issues still remain. These issues are related to how libyuma-dev allocates memory.

5.3.2.1 easyNetconf

C is a low level functional programming language, there are many implementations of the C compiler. All of these implementations follow the C standard set forth by the American National Standards Institute (ANSI) in the ANSI C standard. Due to JavaScipts object-oriented nature we decided to create a wrapper around the node-Yuma123 library. This wrapper allows us to use the node-Yuma123 library in an object-oriented way.

5.4 Grafana

Grafana is an open source data visualization tool. Since its inception ins 2014 it has become the industry standard for data visualization. It is used to visualize arbitrary data from different data sources. We have developed a solution for fetching and visualizing arbitrary NETCONF data using Grafana. We call this solution NetconfAggregator.

5.4.1 NetconfAggregator

NetconfAggregator is a NodeJS package that uses the easyNetconf wrapper [21] to aggregate NETCONF data and store it in a postgresql database. The data is stored in the form of a time series consisting of an ID, a timestamp and the raw XML data from the NETCONF device. This allows the data to be extracted and visualized using XPATH queries. An example of this can be seen in *Figure 11*. This example shows how the query *"//proc/meminfo/MemFree"* can be used to create a graph of the memory usage on a raspberry pi through the NetconfAggregator datasource plugin in Grafana.

Figure 12 shows the architecture of the NetconfAggregator, consisting of a PostgreSQL database, a NetconfAggregator instance and a Grafana instance running the NetconfAggregator datasource plugin. The NetconfAggregator gets the state of each NETCONF device using the yangcli command *"xget /"* which fetches the state of the device. The response is then stored in the database. When the NetconfAggregator receives a query from the datasource plugin it applies the specified XPATH to the relevant data-range in the database and returns the result to the datasource plugin.

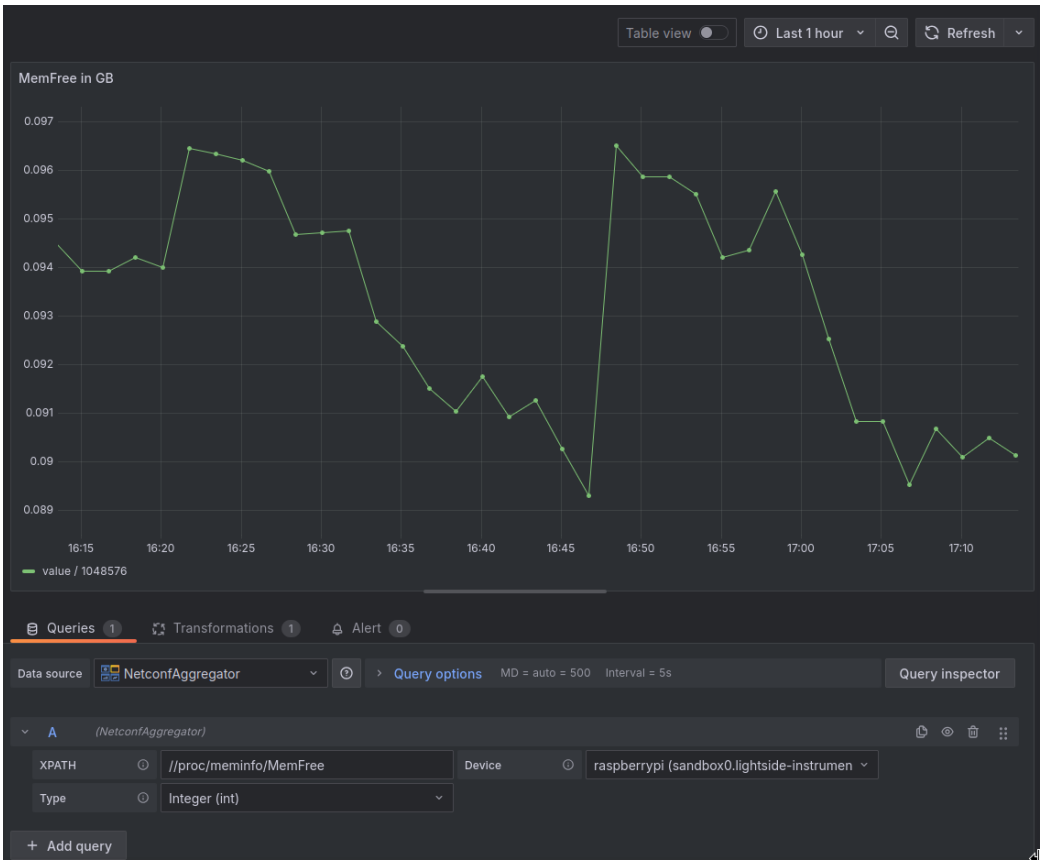

Figure 11: Querying the memory free data on a raspberry pi using XPATH and the NetconfAggregator plugin.

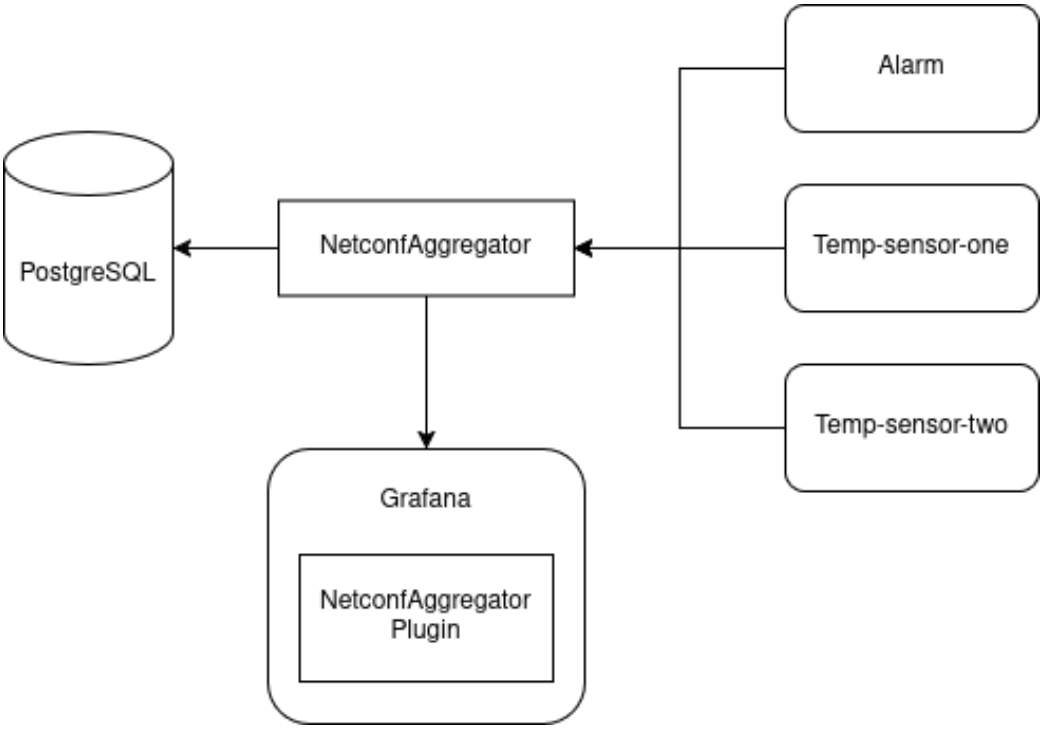

Figure 12: NetconfAggregator architecture.

We quickly discovered that processing XPATHS is processor intensive. Especially when the data set is large. Grafana also has a tendency to send all the queries belonging to a dashboard at once therefore, we decided to implement a caching system, where the results of the XPATH queries are cached in memory, stored in a hash table. However, this did not work as expected, as the queries were arriving at the same time the responses did not have time to reach the cache before the next identical query arrived. To solve this we implemented a queue system, we call it the event queue. When a query goes to processing it is simultaneously added to the event queue. Future identical queries are delayed until the processing of the first query is completed. A delay is added to each request that the server receives. This delay is based on the number of queries that are currently being processed, and thus makes sure that no two identical queries are processed at the same time. An illustration of how requests are processed can be seen in *Figure 13*.

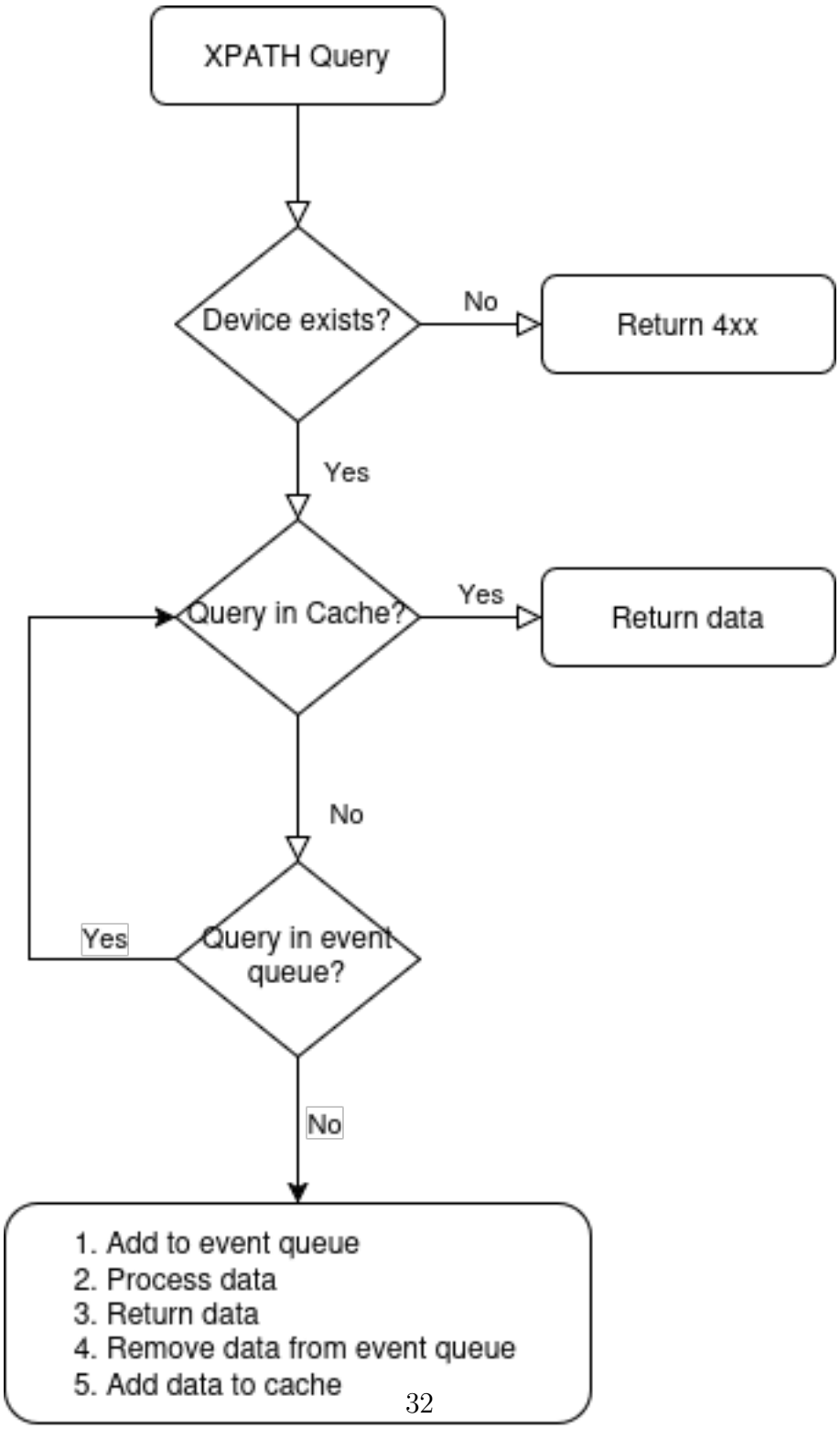

Figure 13: XPATH query processing.

5.5 Decentralized monitoring

We believe that the power of NETCONF lies in its ability to be used in a decentralized manner. This means that multiple aggregators can be used to manage the same NETCONF devices. The example in *Figure 14* shows two servers each containing a node-red instance, a NetconfAggregator instance and a Grafana instance. The node-red instance and the NetconfAggregator functions completely independently of each other. Thus making the monitoring system completely separate from the management system. The example consists of a normal networking setup consisting of three NETCONF enabled switches, a NETCONF enabled temperature sensor and a NETCONF enabled alarm. This architecture allows for arbitrary control and monitoring of the NETCONF devices using the NetconfAggregator and Node-RED.

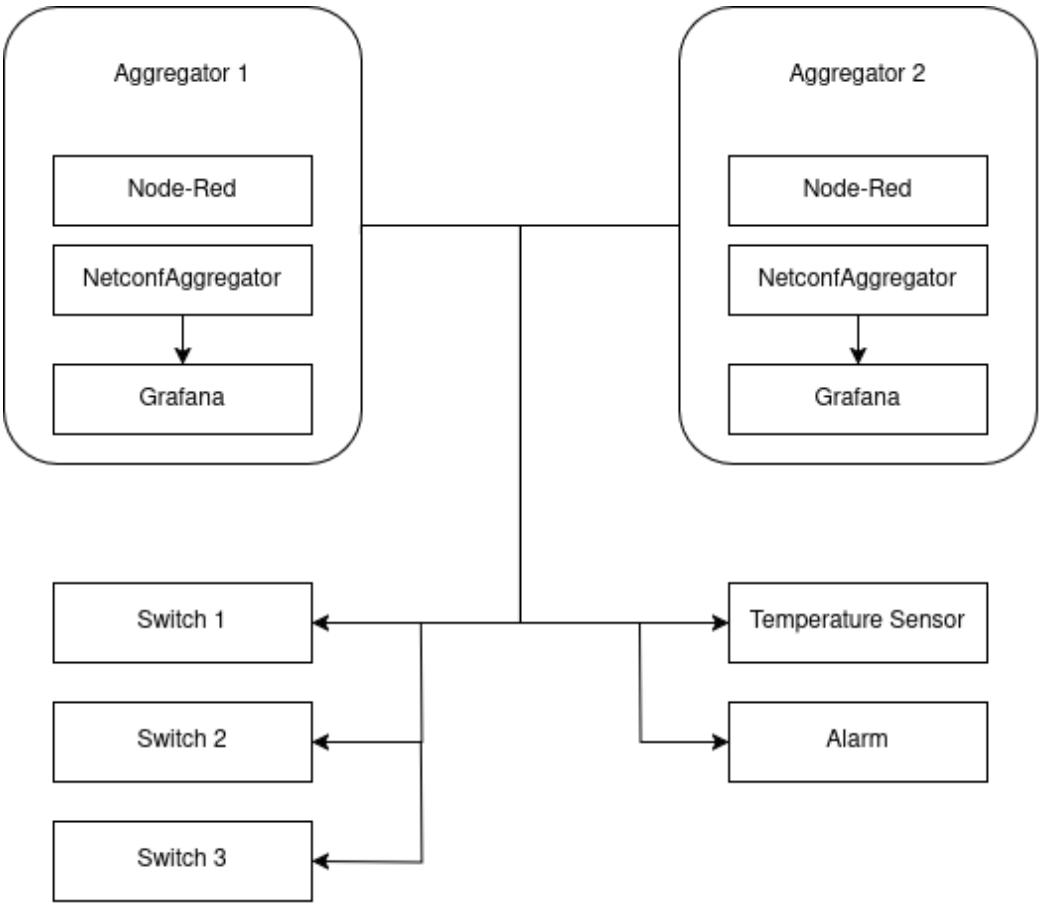

Figure 14: Decentralized monitoring of NETCONF devices using Grafana, NetconfAggregator and Node-RED.

An example of this kind of decentralized architecture can be seen in *figure 14*. This example consists of two servers running both a NetconfAggregator and a Node-RED instance. Due to the way NETCONF works both of these data aggregators and orchestrators can operate on the same devices simultaneously, note that there are a few situations where node-red might create some race conditions but, these are complex flows this only applies to complex flows that are dependent on an internal state. All stateless flows will function without race conditions as the state of a NETCONF device is stored in the device.

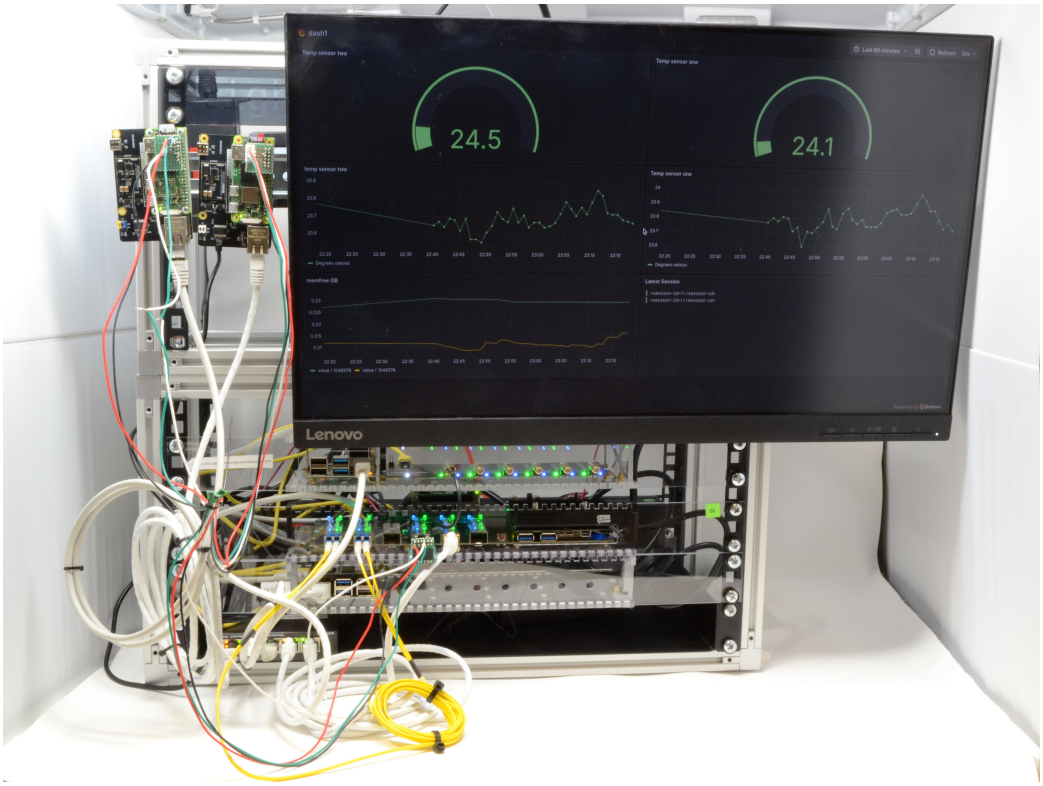

Figure 15: Our reference implementation consisting of two thermometers a NETCONF controllable Led strip, a Node-RED instance, a Grafana instance and a NetconfAggregator instance.

5.6 Our Reference Implementation

After developing our tools and hardware prototypes we decided to create a reference implementation. The goal of the implementation was not only to test the stability of our tools but also function as a product demonstration, as most of our tools are quite low-lever it can be hard to demonstrate the functionality of these tools without multiple devices interacting together as can be seen in *figure 15*.

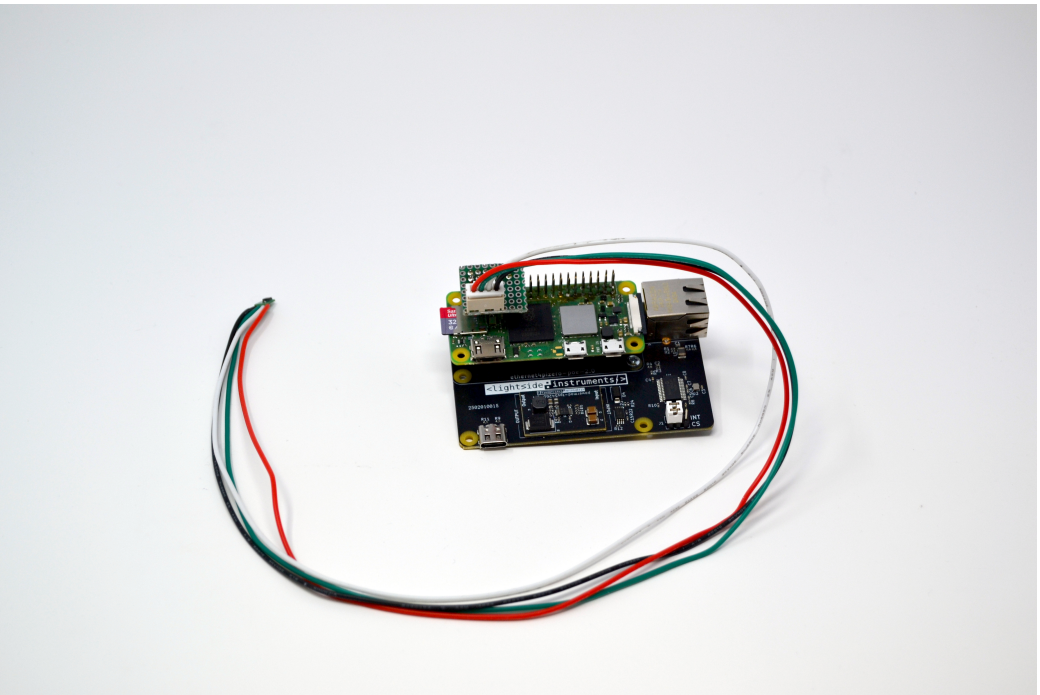

Figure 16: Temperature sensor one, consisting of a Raspberry Pi zero and an Ethernet4pizero-poe shield.
Note: the shield does not contain a transformer, and therefore it is not galvanically isolated, this limits the potential use cases of the shield as there might be large differences in voltage potentials between ground and the devices ground.

We built three different NETCONF thermometers. Two of the thermometers were built on top of a yet to be released product, the Ethernet4pizero-poe shield. This shield adds power over Ethernet to the Raspberry Pi zero creating a cheap and portable solution for NETCONF hardware development. This shield was purpously created in the hope that we would be able to build a temperature probe on top of it. It has been in development by Vassilev and Heimonen since the summer of 2024. An image of our first thermometer can be seen in *figure 16*.

Our third thermometer is quite experimental and not yet finished. Due to its experimental nature we have decided to include it anyway. Small form Factor Pluggable (SFP) is a type of module often used in network switches to make Network Interface Cards (NIC) easily replacable and the swithces easily

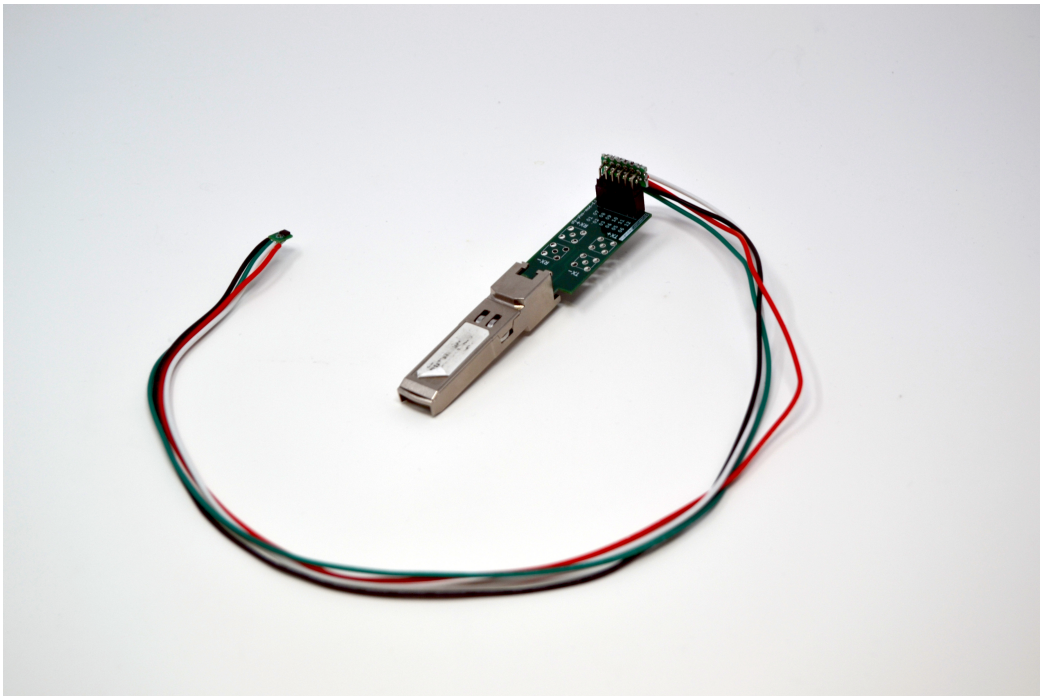

Figure 17: A Small form Factor Pluggable temperature sensor.

configurable. An SFP port features two Serializer Deserializers (SerDes) lanes and most importantly for our purposes an Inter Chip Connect (I2C) lane. Using this I2C lane one can in theory create an SFP sensor probe. This allows for the utilization of empty SFP ports for the monitoring of temperature, humidity etc. We created such a module as can be seen in *figure 17* an example of how the SFP is used can be seen in *figure 18*.

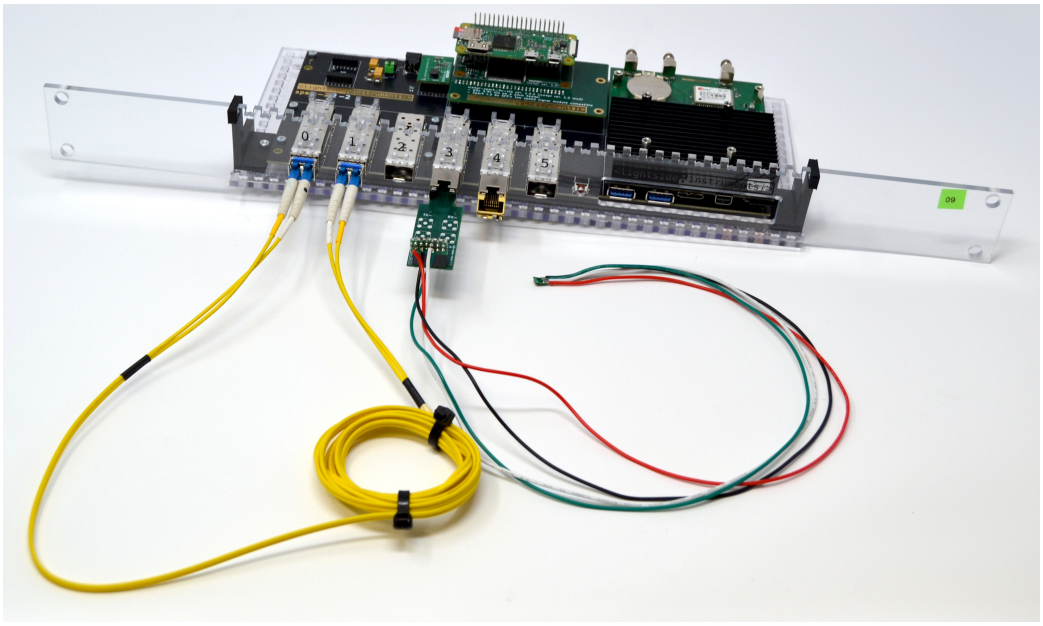

Figure 18: A Small form Factor Pluggable temperature sensor plugged into the SPARK network tester.

5.7 NETCONF Versus SNMP

Simple Network Management Protocl (SNMP) [38] is a protocol proposed in 2006 that aimed to standardize network monitoring and management. Like NETCONF and YANG SNMP is also model based, the difference comes in how the models are acquiered. Whilst a NETCONF server serves its own YANG models to the client upon request, SNMP requires the system administrators to acquire their own Object IDentifier (OID). An Object Identifier much like a YANG model is a data structure that defines data types and locations. The Object Identifiers consists of strings of numbers. Each number represents the section in a leaf. ie: *'1.2.3.4'* would mean that *.2* is a child of *.1* but, *.2* is the second child of *.1* and *.3* is the third child of *.2* and so on. This type of hierarchical data model is quite standard and is found in almost all data modeling languages like JSON and XML. The difference is the fact that the data model is transported separately from the data itself and in the case of SNMP the data models is something that must be acquired manually.

5.8 Open Source Hardware and Software Development

All software developed by Lightside Instruments is open source, published under the Internet Software Consortium (ISC) license [9] while all hardware is licensed under the TAPR open hardware license [44]. All of Lightside Instruments hardware products are also certified or in the process of getting certified by the Open Source Hardware Association (OSHWA).

We believe that when developing secure system's transparency is key therefore it is of the utmost importance that all development happens as transparently as possible in the public eye. Open source software development also allows for third party contributions, this greatly increases the speed at which innovation occurs by allowing developers and engineers to collaborate openly. This collaboration fosters a community-driven approach to development where ideas are shared freely without any commercial incentives. Many modern companies sell Software as a Service (SaaS) products, this kind of software is fundamentally incompatible with open source development and is something that we are fundamentally opposed to. We believe that we must strive to create an ecosystem of software and hardware that is not only free in a monetary sense but also free as in freedom — unrestricted by the constraints of capitalism.

6 NETCONF Security

The security of Remote Procedure Calls (RPC) is an important part of any NETCONF setup. NETCONF is developed in a manner that allows it to function over any transport layer, where the most commonly used is secure shell (SSH) [28]. By utilizing SSH the protocol is able to outsource the security of the transport layer allowing the protocol to focus on the management of network devices instead of the security of the transport layer. This type of compartmentalization is common when it comes to IETF standards. Other notable examples of this are the Internet Protocol Security (IPsec) [42] and the Transport Layer Security (TLS) [37] protocols.

6.1 SSH hardening

A secure shell (SSH) is only as good as the configuration of the SSH server. The process of securing an SSH server is called "hardening". There are many

ways to harden an SSH server. The following are some of the more common practices.

6.1.1 Diffie-Hellman

Asymmetric encryption is one of the most common ways of securing an SSH server. This involves every user generating a public/private key pair, the public key is then passed to the server and the private key is kept on the client. The server can then use the public key to encrypt messages that can only be decrypted by the private key. A central step in initializing an SSH connection is the key exchange (KEX) between the client and server. This is done through the Elliptic-curve Diffie-Hellman (ECDH) KEX protocol. As seen in *Figure 19* the exchange starts by the two parties in the key exchange selecting a large common prime, in this case $p = 7$, after this a generator g of *g(p)* is selected this generator must also be shared between both parties. The generator is a primitive root modulo of p. A number g is a **primitive root modulo of** p if:

$$\sum_{k=1}^{p-1} \left(g^k \pmod{p}\right) = \sum_{n=1}^{p-1} n = \frac{(p-1)p}{2}$$

This means that the sum of the generator raised to the power of *1-p modulus* p will always be the same as the sum of all integers from *1* to *1-p*. Therefore the expression can also be written as:

$$\{g^1 \pmod{p},\ g^2 \pmod{p},\ g^3 \pmod{p},\ \ldots,\ g^{p-1} \pmod{p}\} = \{1, 2, 3, \ldots, p{-}1\}.$$

After the generator has been selected each client generates a secret number. Each party calculates a public key *pubk* that is to be shared using their *secret*.

$$pubk = \{g^{secret} \pmod{p}\}$$

The shared secret s can then be calculated by using the other parties public key, the clients secret and the large common prime p.

$$s = \{pubk^{secret} \pmod{p} \pmod{p}\}$$

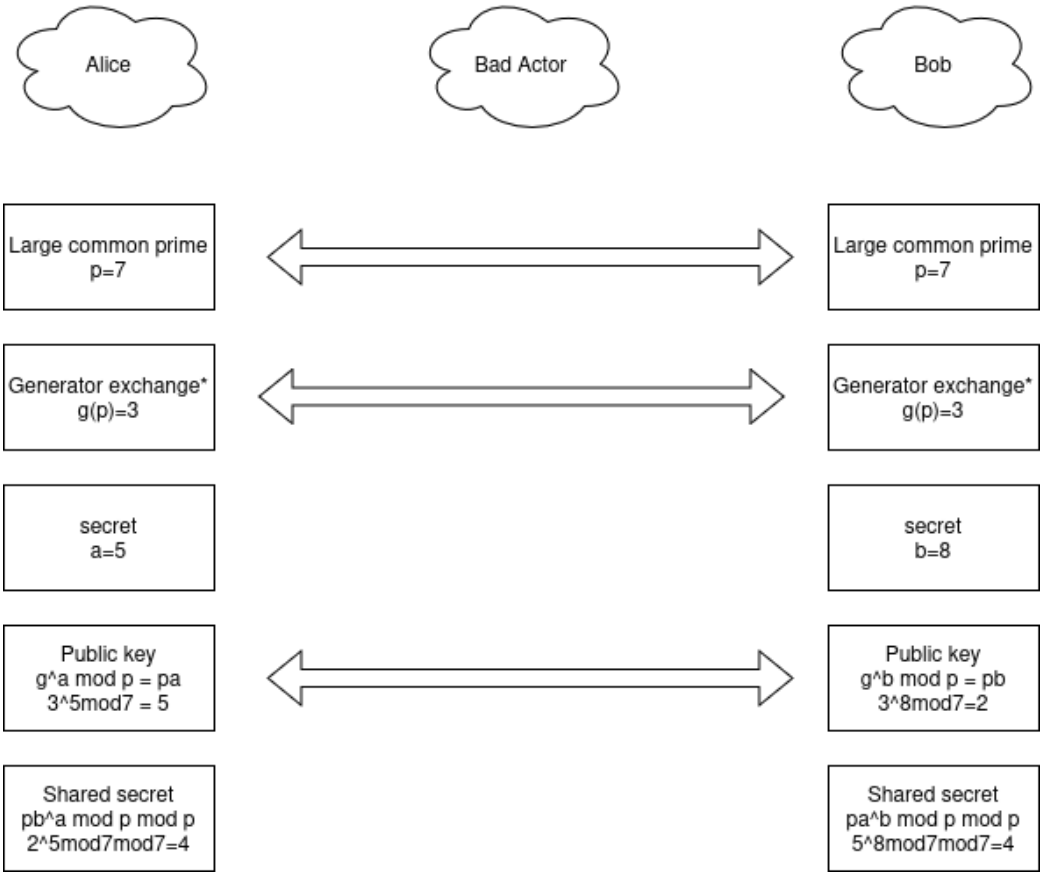

Figure 19: An illustration of the Diffie-Hellman Key Exchange Protocol. *The generator number is a primitive root modulo of p.

Now assuming that the bad actor as can be seen in *Figure 19* has intercepted all traffic it will be to mathematically complex for the bad actor to calculate the shared secret based just on the large common prime p. Assuming the bad actor is trying the General Number Field Sieve (GNFS) we can calculate the computation approximate complexity using the following big O expression where n is the current complexity of p. If $p = 127$ then $n = 2^8$

$$O(n) = \exp((\log n)2/3).$$

Assuming that p is a 1024-bit number there is then the computation complexity of solving the shared secret will be

$$O(2^{1024}) = \exp((\log 2^{1024})2/3).$$

While this is much better than the complexity of a brute force attempt $O(2^{1024}) = exp(2^{1024})$ it is still computationally impossible for a conventional computer. An illustration of the running computational complexity can be seen in *Figure 20*

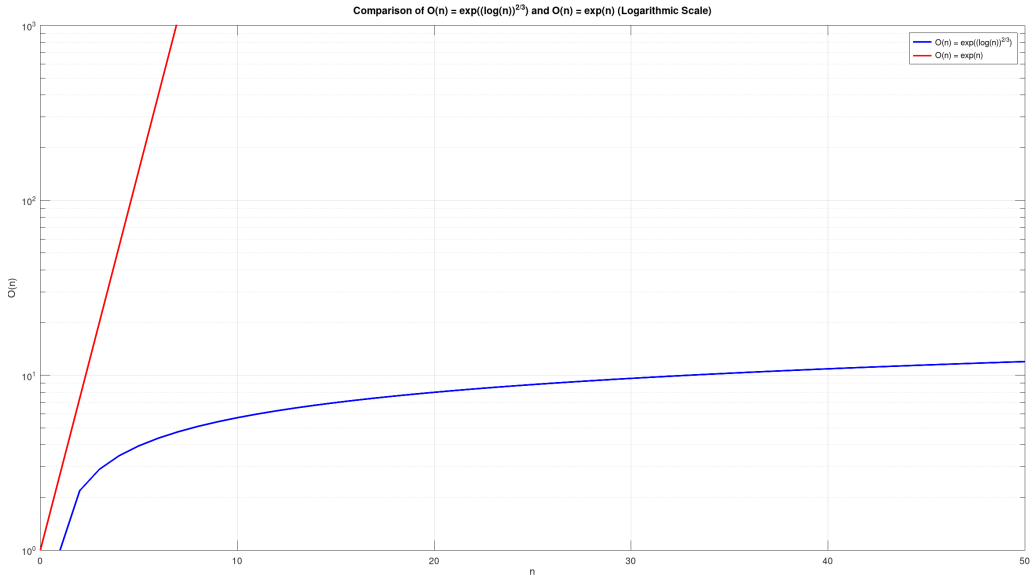

Figure 20: A plot of the running computational complexity of GNFS versus A brute force attempt on a logarithmic scale.

6.1.2 Private Public Key Authentication

One might wonder if SSH uses the Diffie-Hellman Key Exchange protocol to create a shared secret used for encryption then, what are the Public and Private keys used for? After the Diffie-Hellman Key Exchange Protocol has finished the SSH server sends the SSH client a challenge. This is called challenge-response authentication [8], the server send the client a unique string. The client creates a hash of this unqiue string. The hash is then encrypted using the clients private key and sent back to the server, this functions as a signature for the unique string. The server can then use its public key to decrypt the signature and check if the hash matches the unique string.

Why is not just all data encrypted using two pairs of private keys and two pairs of public keys in a form of dual-key encryption as seen in *figure 21*? Notice that the bad actor is storing all the data sent between Alice and Bob. This might remind you of the recent store now and decrypt later craze that has come about due to recent advances in quantum computing and cryptography. But, it is not. If Alice's system were to be compromised by the bad actor they could simply use Alice's private key to decrypt Alice's part of the

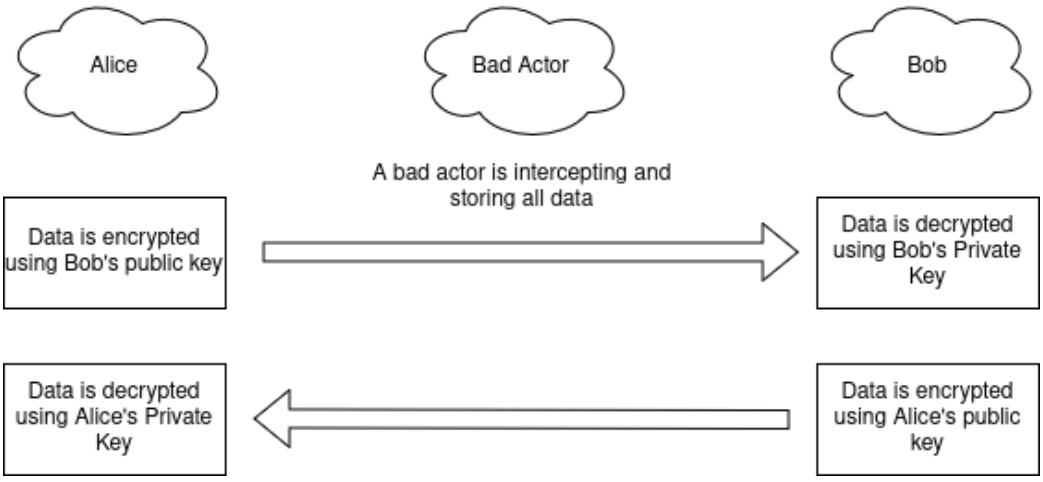

Figure 21: Dual-key encrypted communication between Bob and Alice

communication. Whilst if they were only using the key pair for authentication and using the Diffie-Hellman Key Exchange protocol for encryption this would not present an issue. The issue is not the key pairs in itself but the fact that the keys are static, the more data that is encrypted using the same key, the greater the potential damage if the key is ever compromised.

6.1.3 Fail2ban

Fail2ban is a daemon [17] that scans an SSH servers auth log for unwelcome intrusions. If an intrusion is detected it creates a system firewall rule to block the IP address. Whilst a correctly configured SSH server should be secure without fail2ban they still experience multiple brute-force attacks every hour if they are open to the internet. This can be draining on the SSH servers performance and the system resources. Therefore, it is practical to have a tool that detects and blocks the bad actors. In 2024 the CVE-2024-6387 [11] vulnerability was discovered, this vulnerability allowed a bad actor to perform a denial of service attack on all OpenSSH servers through a race condition in the OpenSSH servers authentication handling. This is a vulnerability that while not totally mitigated by fail2ban the risk of this happening would be greatly reduced.

6.2 Node-RED

The security of NETCONF becomes mute if the rest of the system is not secured. Therefore it is of the upmost importance to secure the Node-RED instance as per the Node-RED documentation [41]. Node-RED has support for both HTTPS and a user authentication system with tunable permissions out of the box. This means that different users can have different permissions, some might only have permission to read flows while others can both create and read flows. It is also worth mentioning that Node-RED can be configured to encrypt the *flows.json* file using the systems key store if one is available. The *flows.json* file is important to secure as it will contain the NETCONF authentication credentials if the connection is not using private public key authentication.

7 Conclusion

We set out to bring support for NETCONF to both Grafana and Node-RED, this involved creating custom tools and libraries that would allow for seamless integration of NETCONF devices into these platforms. Through the development of Node-Yuma123, easyNetconf, Red-Netconf, and NetconfAggregator, we have successfully achieved this goal. Our tools have been tested in various scenarios, demonstrating their versatility and reliability. The reference implementation provided a practical demonstration of the capabilities of our tools. By integrating NETCONF-enabled thermometers, a data aggregator, and Grafana as a visualization tool, we showcased how NETCONF can be used effectively for sensor management. The implementation also highlighted the flexibility of our tools in adapting to different use cases, such as the rover control flow. While the tools and implementations have proven to be effective, there are areas for improvement. For instance, the caching mechanism in NetconfAggregator could be further optimized to handle high query loads more efficiently. Additionally, we could have worked out an IEEE standard for our SFP thermometer thus making the proposal a bit less experimental. Overall, we hope that the work we have done will contribute to the popularizing of NETCONF as a protocol for network monitoring and administration. We hope that people will adapt the use of NETCONF trough both Node-RED using Red-Netconf and Grafana using the NetconfAggregator.

References

[1] *@lightside-Instruments/Red-Netconf*. http://flows.nodered.org/node/@lightside-instruments/red-netconf. (Visited on 04/21/2025).

[2] *Accenture — Let There Be Change*. https://www.accenture.com/no-en. (Visited on 04/27/2025).

[3] *ACM Digital Library*. https://dl.acm.org/. (Visited on 03/28/2025).

[4] *Agile Alliance*. https://www.agilealliance.org/. June 2015. (Visited on 03/28/2025).

[5] "Agile Software Development". In: *Wikipedia* (Apr. 2025). (Visited on 04/27/2025).

[6] Martin Björklund. *The YANG 1.1 Data Modeling Language*. Request for Comments RFC 7950. Internet Engineering Task Force, Aug. 2016. DOI: 10.17487/RFC7950. (Visited on 01/12/2025).

[7] Jeff Brehm. *AGILE TRANSFORMATION? FOR COMPLEX SYSTEMS? ...NO WAY!*. Oct. 2025. (Visited on 04/27/2025).

[8] "Challenge–Response Authentication". In: *Wikipedia* (Dec. 2024). (Visited on 05/13/2025).

[9] Internet Systems Consortium. *ISC Open Source Software Licenses*. https://www.isc.org/licenses/. Mar. 2024. (Visited on 05/13/2025).

[10] "Continuous Delivery". In: *Wikipedia* (Jan. 2025). (Visited on 04/27/2025).

[11] *CVE Record: CVE-2024-6387*. https://www.cve.org/CVERecord?id=CVE-2024-6387. (Visited on 05/13/2025).

[12] *Deloitte — Audit, Consulting, Financial, Risk Management, Tax Services*. https://www.deloitte.com/global/en.html. (Visited on 04/27/2025).

[13] *DORA — Get Better at Getting Better*. https://dora.dev. (Visited on 04/27/2025).

[14] Rob Enns et al. *Network Configuration Protocol (NETCONF)*. Request for Comments RFC 6241. Internet Engineering Task Force, June 2011. DOI: 10.17487/RFC6241. (Visited on 01/12/2025).

[15] *Extensible Markup Language (XML)*. https://www.w3.org/XML/. (Visited on 05/07/2025).

[16] "Extreme Programming". In: *Wikipedia* (Apr. 2025). (Visited on 04/27/2025).

[17] *Fail2ban/Fail2ban*. Fail2Ban. May 2025. (Visited on 05/13/2025).

[18] *Framework*. https://framework.scaledagile.com/. (Visited on 04/27/2025).

[19] *Google Scholar*. https://scholar.google.com/. (Visited on 03/28/2025).

[20] *Grafana: The Open and Composable Observability Platform*. https://grafana.com/. (Visited on 01/20/2025).

[21] Joar Heimonen. *Slenderman00/Netconfaggregator*. May 2025. (Visited on 05/07/2025).

[22] *Home — Scrum.Org*. https://www.scrum.org/index. (Visited on 03/28/2025).

[23] *IEEE Xplore*. https://ieeexplore.ieee.org/Xplore/home.jsp. (Visited on 03/28/2025).

[24] "Kanban". In: *Wikipedia* (Apr. 2025). (Visited on 04/27/2025).

[25] "Kanban (Development)". In: *Wikipedia* (Mar. 2025). (Visited on 04/27/2025).

[26] *Libuv Documentation*. https://docs.libuv.org/en/v1.x/. (Visited on 05/13/2025).

[27] *Lightside Instruments AS – YANG Model Network Managed Instruments*. (Visited on 01/20/2025).

[28] Chris M. Lonvick and Tatu Ylonen. *The Secure Shell (SSH) Transport Layer Protocol*. Request for Comments RFC 4253. Internet Engineering Task Force, Jan. 2006. DOI: 10.17487/RFC4253. (Visited on 05/14/2025).

[29] *Low-Code Programming for Event-Driven Applications : Node-RED*. https://nodered.org/. (Visited on 03/29/2025).

[30] *Manifesto for Agile Software Development*. http://agilemanifesto.org/. (Visited on 06/02/2024).

[31] *Node-Yuma123*. https://www.npmjs.com/package/node-yuma123. Apr. 2025. (Visited on 04/22/2025).

[32] *Node.Js — Run JavaScript Everywhere*. https://nodejs.org/en. (Visited on 03/29/2025).

[33] *Organisations-*. https://www.peoplecert.org/Organisations. (Visited on 03/28/2025).

[34] "Pair Programming". In: *Wikipedia* (Nov. 2024). (Visited on 04/27/2025).

[35] *Phased Project Planning Guidelines*. Aug. 1968. (Visited on 04/27/2025).

[36] *PRISMA Statement*. https://www.prisma-statement.org. (Visited on 03/28/2025).

[37] Eric Rescorla and Tim Dierks. *The Transport Layer Security (TLS) Protocol Version 1.2*. Request for Comments RFC 5246. Internet Engineering Task Force, Aug. 2008. DOI: 10.17487/RFC5246. (Visited on 05/13/2025).

[38] Jürgen Schönwälder and Tony Jeffree. *Simple Network Management Protocol (SNMP) over IEEE 802 Networks*. Request for Comments RFC 4789. Internet Engineering Task Force, Nov. 2006. DOI: 10.17487/RFC4789. (Visited on 05/14/2025).

[39] "Scrum (Rugby Union)". In: *Wikipedia* (Apr. 2025). (Visited on 04/27/2025).

[40] *Scrum Alliance - Find Courses for Scrum and Agile Certifications*. https://www.scrumalliance.org. (Visited on 03/28/2025).

[41] *Securing Node-RED : Node-RED*. https://nodered.org/docs/user-guide/runtime/securing-node-red. (Visited on 05/14/2025).

[42] Karen Seo and Stephen Kent. *Security Architecture for the Internet Protocol*. Request for Comments RFC 4301. Internet Engineering Task Force, Dec. 2005. DOI: 10.17487/RFC4301. (Visited on 05/13/2025).

[43] "The New New Product Development Game". In: *Harvard Business Review* (). ISSN: 0017-8012. (Visited on 04/27/2025).

[44] *The TAPR Open Hardware License – TAPR.* https://tapr.org/the-tapr-open-hardware-license/. (Visited on 05/13/2025).

[45] Vladimir Vassilev. *Vlvassilev/Yuma123.* Mar. 2025. (Visited on 04/27/2025).

[46] Christiaan Verwijs and Daniel Russo. "A Theory of Scrum Team Effectiveness". In: *ACM Transactions on Software Engineering and Methodology* 32.3 (July 2023), pp. 1–51. ISSN: 1049-331X, 1557-7392. DOI: 10.1145/3571849. (Visited on 04/27/2025).

[47] "Waterfall Model". In: *Wikipedia* (Feb. 2025). (Visited on 04/27/2025).

[48] *XML Path Language (XPath) 3.1.* https://www.w3.org/TR/xpath-31/. (Visited on 05/07/2025).

[49] *Yuma Yangcli Manual - Yuma123 Wiki.* https://yuma123.org/wiki/index.php/Yuma_yangcli_Manual. (Visited on 05/13/2025).

List of Figures

© 2025 Joar Heimonen, Christian Vu, Naly Keli
This work is licensed under a Creative Commons Attribution-Sharealike 4.0 International License.