Fridolin Siegmund, Stefano Acquaviti, Bjoern Nagel, Maik Rüder, Matthias Hollick, Ralf Kundel. *Open PONAlyzer: A Software-Defined Analysis and Testing Solution for Enhanced Security and Resilience in Passive Optical Networks* To appear in the Proceedings of IEEE Conference on Network Function Virtualization and Software Defined Networks 2025, IEEE, 2025.

Open PONAlyzer: A Software-Defined Analysis and Testing Solution for Enhanced Security and Resilience in Passive Optical Networks

Fridolin Siegmund*, Stefano Acquaviti*, Bjoern Nagel‡, Maik Rüder‡, Matthias Hollick*, Ralf Kundel§
*Technical University of Darmstadt
fridolin.siegmund@tu-darmstadt.de
‡Deutsche Telekom Technik GmbH
§Darmstadt University of Applied Sciences

***Abstract*—The increasing demand for high-bandwidth, low-latency Internet access is driving the widespread adoption of cost-efficient Passive Optical Networks (PONs). However, the passive nature of PONs necessitate complex subscriber device management protocols, and their shared fiber architecture presents potential disruptions from device failures, misconfigurations, or malicious attacks. To maintain service quality and customer satisfaction, network operators require effective PON monitoring solutions. Yet, existing commercial hardware-based analyzers are prohibitively expensive, challenging to deploy in live networks, and unsuitable for nationwide deployments. This paper introduces PONAlyzer, a novel, open-source software-based PON analyzer and security dashboard, which we make available to the community. By leveraging existing Software Defined Networking (SDN) interfaces of in-field PON-hardware, *i.e.* eliminating the need for additional analyzer equipment, PONAlyzer enables the capture and analysis of ONU Management and Control Interface (OMCI) messages, and supports OMCI message injection for advanced security testing and troubleshooting.**

***Index Terms*—PON, SDN, VNF, Security, Anomalies, Monitoring, Analyzer, Access Networks, Resilience**

I. Introduction

With the ever-growing demand for reliable Internet access with high throughput and low latency, fiber networks have become a cornerstone of fixed access infrastructure for both residential and business subscribers. To meet these demands, Passive Optical Networks (PONs) are widely deployed as the "last mile" technology, offering cost efficiency, scalability, and adaptability. These advantages originate in the passive nature of the shared medium: multiple subscribers are connected via a single optical fiber, where a passive splitter distributes the downstream signal and combines time-division multiplexed upstream transmissions. Besides home and business Internet access, PONs also can be used as an enabling technology for other Internet services, such as backhaul connections in future 6G access networks.

The most prominent and currently widely deployed PON technology is Gigabit Passive Optical Network (GPON) with 2.5 Gbit/s shared downstream and 1.25 Gbit/s shared upstream bandwidth. GPON includes integrated support for managing connected subscriber devices using specialized protocols for tasks such as configuration and monitoring. However, the shared fiber architecture makes this complex protocol stacks difficult to troubleshoot for operators like Internet Service Providers (ISPs). Also, monitoring these specialized protocols and general subscriber states would enhance the overall resilience of the fiber access networks as faulty devices, misconfigurations, and possible attackers could be detected. Current commercial hardware-based solutions on the market offer extended analysis capabilities but come at high prices due to low quantities in a niche market. Also, the practicability in real-world deployments is limited as a consequence of additional hardware and cabling required to integrate these solutions in existing access networks. As a result, such specialized devices are rarely used outside of laboratories or for uncommon troubleshooting in the field.

While in traditional GPONs the corresponding protocols are handled in the PON hardware itself, today's Software Defined Networking (SDN) paradigm allows the separation of these specialized protocols and the actual subscriber data path. Consequently, the separation of the control plane and data plane facilitates novel analytical approaches that deliver capabilities comparable to those of high-cost commercial analysis tools without incurring any additional expenses. However, the information available on the SDN interfaces is limited to protocols that are sent through these interfaces while, *e.g.*, physical layer provisioning is not handled by a dedicated control plane. Furthermore, PON messages can not only be analyzed but also new messages can be injected to test the connected subscriber devices for all possible scenarios regarding configuration and security.

To address these challenges, we introduce *PONAlyzer*, a novel, open-source software-based solution that leverages SDN interfaces to provide real-time visibility and control over GPON deployments. PONAlyzer combines both analytical and injection scenarios, and offers a user-friendly dashboard for network operators to monitor and test their GPONs deployments. *PONAlyzer* can be deployed within a live network without creating any disruption as it is purely software-based. This prototype is easily adaptable to future 10 Gigabit Symmetrical Passive Optical Network (XGS-PON) deployments, as a similar protocol stack is included.

Fridolin Siegmund, Stefano Acquaviti, Bjoern Nagel, Maik Rüder, Matthias Hollick, Ralf Kundel. *Open PONAlyzer: A Software-Defined Analysis and Testing Solution for Enhanced Security and Resilience in Passive Optical Networks* To appear in the Proceedings of IEEE Conference on Network Function Virtualization and Software Defined Networks 2025, IEEE, 2025.

Our key contributions are as follows:

- An investigation of (software-defined) PON protocols and interfaces for security and development analysis.
- Implementation of the software-based tool *PONAlyzer* for capturing and injecting messages in the PON control plane, which we make available to community.
- As a proof-of-principle, we demonstrate and evaluate our *PONAlyzer* open source tool.

II. Background

PON access networks are built as a point-to-multipoint tree topology with an Optical Line Terminal (OLT) access node providing L2 connectivity to each subscriber. Multiple variants of PON exists in several ITU standards, such as GPON [1], XGS-PON [2], Next-Generation Passive Optical Network 2 (NG-PON2) [3], and Higher Speed PON [4]. GPON is currently the most widely deployed PON standard, offering 2.5 Gbit/s downstream and 1.25 Gbit/s upstream shared bandwidth. Meanwhile, XGS-PON, which supports symmetrical 10 Gbit/s transmission, is actively being deployed in many countries. PONs are typically used with up to 32 subscribers sharing a single optical fiber. While the standard allows for more, such deployments are rather suitable for lower-bandwidth use-cases. As Figure 1 shows, the subscribers at one fiber receive the downstream packets as a broadcast with a passive splitter duplicating the light waves and, therefore, packets at the physical layer to each subscriber terminal, called Optical Network Unit (ONU). Behind the ONU, the Residential Gateway (RG) provides L3 connectivity as a logical termination point with routing and firewalling functionality. Some end-user hardware solutions combine both ONU and RG functionality into a single device, which are also available on the market. In the upstream direction, Time-Division Multiplexing (TDM) is used to prevent signal collisions at the passive splitter, which merges the subscriber signals.

GPON and XGS-PON share similar control plane protocols for configuration, provisioning, and monitoring. While Physical Layer Operations, Administration and Management (PLOAM) configures physical layer parameters such as the time slots for upstream TDM, ONU Management and Control Interface (OMCI) [5] is used to manage and configure the non-physical layer services of the ONUs. As Figure 1 shows, OMCI control plane messages are sent between the OLT

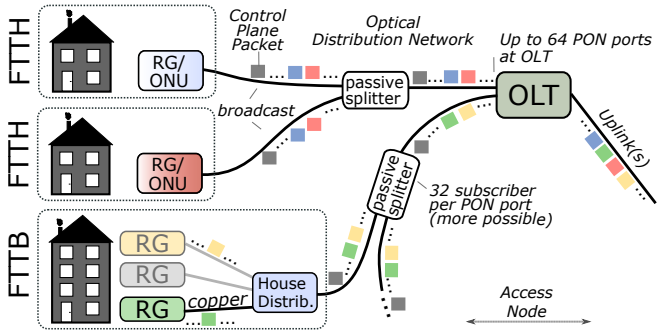

Fig. 1: PON with control plane and data plane packets.

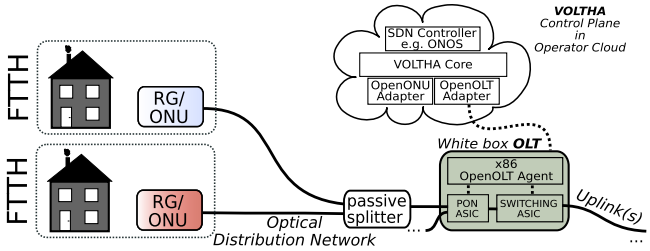

Fig. 2: White box OLT deployed with VOLTHA control plane.

and ONUs over the ONU Management and Control Channel (OMCC) through the Optical Distribution Network (ODN) with the OLT typically sending OMCI request messages to an ONU, which then replies with an OMCI response. Therefore, the OMCI communication can be considered a master-slave relationship between the OLT and the ONUs. Information and parameters of specific services are stored inside the Management Information Base (MIB), as objects, Managed Entities (ME), or more general Transmission Containers (TCONTs) [6]. The complete MIB is stored at the OLT and is partially or fully synchronized with each ONU via OMCI.

The four main purposes of the OMCI protocol are [5]:

1) Configuration Management: The OLT identifies the ONU's capabilities and configures the ports, flows, termination points, and service profiles, including Quality of Service (QoS) settings and bandwidth profiles.
2) Fault Management: The ONU sends alarm notification messages whenever a fault occurs. The faults range from physical defects to interface or software issues.
3) ONUs employ dedicated MEs to gather performance metrics, which are reported to the OLT via OMCI. Alarms can be triggered upon threshold violations.
4) Security Management: OMCI enables the authentication of ONUs at the OLT, *e.g.*, by serial number.

While conventional OLTs handle the control plane messages such as OMCI locally, the Virtual OLT Hardware Abstraction (VOLTHA) project applies the SDN paradigm by decoupling the control plane and data plane in PONs [7], [8]. The Virtual Network Functions (VNFs) of the control plane are deployed in a Kubernetes cluster in the operator cloud, as shown in Figure 2. One of the multiple goals of VOLTHA is shifting OMCI message handling from the OLT to a dedicated control plane instance running as a VNF. Also, VOLTHA abstracts the OLT as a switching device to the SDN controller to enable a flow-based data path representation of the subscriber data plane to the controller. Therefore, when a subscriber ONU is provisioned, the SDN controller adds flows and traffic queues for the subscriber. The VOLTHA core translates these operations into OLT-targeted operations. After that, the OpenONU Adapter handles the ONU-specific configuration, such as OMCI, which is forwarded from the VOLTHA control plane to the OLT and vice versa. All communication, whether from the OpenONU Adapter or the VOLTHA core itself, is sent through the OpenOLT Adapter to the OLT. The Google Remote Procedure Call (gRPC) framework is utilized for

Fridolin Siegmund, Stefano Acquaviti, Bjoern Nagel, Maik Rüder, Matthias Hollick, Ralf Kundel. *Open PONAlyzer: A Software-Defined Analysis and Testing Solution for Enhanced Security and Resilience in Passive Optical Networks* To appear in the Proceedings of IEEE Conference on Network Function Virtualization and Software Defined Networks 2025, IEEE, 2025.

message exchange between all VOLTHA VNFs and the OLT; the enabling interface for this work.

III. Design

Current PON analyzer solutions on the market (*e.g.*[*]) allow tapping into the traditional sniffing point marked in Figure 3 directly in the ODN. The traditional approach with hardware PON analyzers allows capturing all types of messages, including PLOAM and OMCI, while coming at high costs and the need for additional cabling and splitters. Typically, only one PON fiber in down- and upstream direction is supported for live analysis with commercial solutions. Also, network downtime may be induced when the necessary cabling for the hardware PON analyzer is added. Another disadvantage is the additional signal attenuation by adding another splitter, as most hardware-based analyzers are equipped with separate downstream and upstream capturing interfaces. In worst-case scenarios with exhausted cable lengths and signal levels, the measurement could disrupt the subscriber connections.

The separation of the data and control plane is one of the key paradigms in SDN-based architectures and includes segregated handling of packets belonging to these two planes. Considering such a Software Defined Passive Optical Network (SD-PON), not all types of control plane messages are handled by a dedicated controller outside the OLT. The PON application-specific integrated circuit (ASIC) inside the OLT locally handles some protocols types such as PLOAM in hardware with a low-level firmware. However, higher-layer messages such as OMCI can be processed by a dedicated controller outside the OLT, *e.g.*, as it is implemented in VOLTHA.

We propose an architecture exploiting the architectural advantages of SDN for security analysis with a new sniffing point between the SD-PON control plane, respectively, controller, and the OLT management interface as shown in Figure 3. As OMCI messages are sent over this interface, the possibility of capturing and dissecting these messages opens up. After capturing the packets, a backend processes the messages according to the OMCI specification and possible additional metadata. This approach has the advantage of being able to capture messages from all PON ports instead of only one

[*]https://www.albedotelecom.com/pages/gpon/src/gpd10k7.php

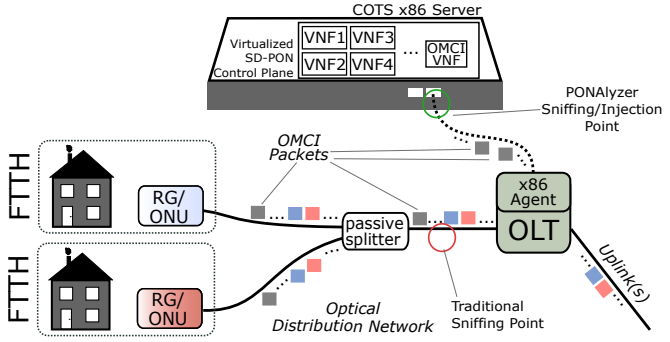

Fig. 3: OMCI packets in SD-PON with the traditional and new approach of sniffing points.

fiber as with the commercial PON analyzers. At this point, the infrastructure for sniffing also opens the possibility of injecting OMCI messages directly to the OLT, which forwards them to the respective ONUs. Both functionalities enhance the resilience of the fiber access network which are exposed to various threats. *PONAlyzer* addresses two main perimeters where threats to PONs can originate from:

1) In the PON fiber tree
2) Within the operators control plane network

Possible adversaries originating in 1) can access different vulnerabilities in typical ODNs. Due to the unprotected PON splitters, *e.g.*, in the basement of buildings, non-residents can access the PON tree physically and receive the broadcasts, inject packets, or jam the connection physically by inserting light waves. Also, malicious or faulty ONUs may send data outside their configured time slot in the TDM domain and degrade service for other subscribers. If an adversary has access to the fiber between the splitter and the OLT, malicious configurations can be injected to the ONUs. Additionally, interoperability issues between different vendors of ONUs and OLTs as well as aging or disrupted fibers can result in influenced connections. Threats originating from 2) are mostly misconfigurations or faulty software, as the operator's control plane networks are typically isolated. However, if an attacker penetrates the operator's control plane network, many threats, such as unauthorized configuration changes, leaking of sensitive data, service degradation, and Denial of Service (DoS), are possible.

To address these issues, *PONAlyzer* monitors the live OMCI control plane traffic, which allows detecting anomalies indicating a rogue ONU and wrong configured ONUs in the PON and, thus, increases the resilience reactively. OMCI anomalies are in a broad range of events, *e.g.*, out-of-sequence messages, increased occurrence of alarm messages, decoding errors, and duplicates. For example, steadily increasing decoding errors may indicate an ongoing attack or a partly jammed or physically broken fiber connection. Also, attacks from within the provider network based on the OMCI protocol are captured by *PONAlyzer* and can be detected this way by the operators. The injection capability allows extensive testing of ONUs regarding their OMCI specification conformity and edge-case testing for developers and customers to enhance resilience proactively. Additionally, the injection functionality allows for proactive penetration tests as the role of an adversary inside the operator control plane network.

Therefore, *PONAlyzer* offers a set of functions similar to hardware-based commercial PON analyzers, coming at zero cost and without any additional cabling as it is purely software-based. Also, commercial analyzers typically do not include crafting and injecting OMCI messages for testing purposes; they are proprietary solutions with no possibility of adapting the software for future or individual requirements and analysis features. Compared to commercial solutions, *PONAlyzer* exclusively dissects OMCI messages and additional control plane metadata, while commercial solutions also support physical

Fridolin Siegmund, Stefano Acquaviti, Bjoern Nagel, Maik Rüder, Matthias Hollick, Ralf Kundel. *Open PONAlyzer: A Software-Defined Analysis and Testing Solution for Enhanced Security and Resilience in Passive Optical Networks* To appear in the Proceedings of IEEE Conference on Network Function Virtualization and Software Defined Networks 2025, IEEE, 2025.

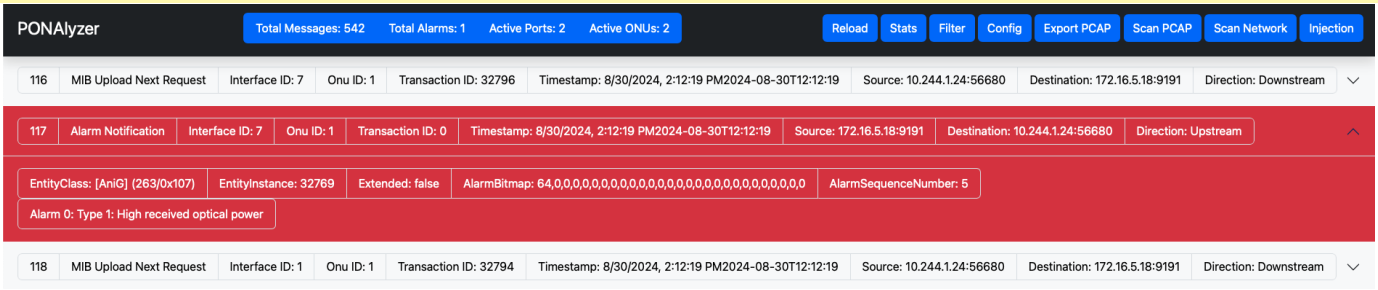

Fig. 4: PONAlyzer WebUI with an exemplary anomaly (high received optical power) marked red.

layer provisioning messages in their analysis. This originates in how both solutions work, as *PONAlyzer* is capturing at the control plane interfaces of the OLT, whereas commercial measurement solutions tap directly into a single fiber. Hence, the measurement scope of *PONAlyzer* exceeds the scope of traditional analyzers massively, as all fibers of multiple OLTs can be analyzed simultaneously.

IV. Implementation

The *PONAlyzer* prototype is implemented for the SD-PON project VOLTHA [7], currently the most popular open-source solution on the market. As a straightforward solution, the GoLang-based backend of *PONAlyzer* can run on the same host where the VOLTHA Kubernetes cluster is deployed. As illustrated in Figure 5, the WebUI, accessible from any browser on an arbitrary host, is served by the *PONAlyzer* backend. The prototype is split up into two main functionalities: 1) Analyzing the OMCI messages in the access network to monitor for, *e.g.*, rogue ONUs and 2) injecting OMCI messages for, *e.g.*, extended ONU testing. In both cases, the serial data stream must be decoded or encoded to OMCI messages as defined in the standard [5]. For this, the open-source library *omci-lib-go* [9] is utilized.

A. Analyzer

The analyzer functionality captures the Ethernet packets, containing gRPC messages between the OLT and the VOLTHA control plane stack. Alternatively, the user can select a recorded pcap file of this connection for analysis. As gRPC batches multiple messages into one packet, there is no fixed offset in the packet for parsing the OMCI message header. Therefore, each packet is inspected to determine if an OMCI message is present and at which offset. VOLTHA transmits serialized OMCI messages in gRPC with additional fields, such as the ONU identifier and the PON port at the OLT. When an OMCI message is detected in a gRPC packet, the OMCI message is decoded and preprocessed for the WebUI. The front end offers several analysis capabilities with statistics and out-of-order detection. For example, the statistics include information about the control plane network, discovered ONUs, the total number of messages, the average and maximal response time in milliseconds. Moreover, the WebUI is capable of detecting potential anomalies; however, a human consultant remains necessary, as the fully automated detection and classification of anomalies in OMCI has not yet been studied in depth. Figure 4 shows a capture with an occurring anomaly marked red. In this example, the anomaly is a *high received optical power* from the ONU, indicating a high signal power caused by the short deployed fiber in our testbed. Another feature of *PONAlyzer* is exporting a measurement into a pcap file, which, *e.g.*, can be shared and loaded again in a different *PONAlyzer* installation if desired.

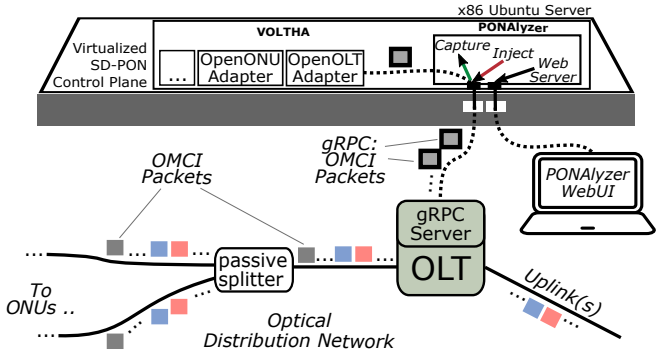

Fig. 5: *PONAlyzer* architecture with a white box OLT.

B. Injection

To inject an OMCI message, the user can generate multiple types of OMCI messages in the WebUI by selecting checkboxes. Also, it is possible to paste arbitrary encoded OMCI messages as bytestream. As a first step, the *PONAlyzer* backend initiates an additional gRPC connection to the OLT by the IP and port the user provided, parallel to the legacy connection by the VOLTHA control plane. Additionally, the targeted ONU must be specified. As the next step, the user-defined OMCI message is serialized, and a VOLTHA-compliant gRPC message is crafted and sent to the OLT, which forwards the message to the selected ONU. The crafted OMCI message can be seen in the analyzer in the live view. Several scenarios for injecting messages are possible. For example, knowledge about each ONU in the PON network can be obtained by facilitating an entire *MIB Upload* process. Another example among many would be the injection of purposely malformed OMCI messages to test ONUs.

V. Evaluation

Our evaluation testbed consists of a VOLTHA-based Radisys RLT-1600X OLT and a Huawei EchoLife HG8010H ONU. The *PONAlyzer* backend is running in an Ubuntu 24.04

Fridolin Siegmund, Stefano Acquaviti, Bjoern Nagel, Maik Rüder, Matthias Hollick, Ralf Kundel. *Open PONAlyzer: A Software-Defined Analysis and Testing Solution for Enhanced Security and Resilience in Passive Optical Networks* To appear in the Proceedings of IEEE Conference on Network Function Virtualization and Software Defined Networks 2025, IEEE, 2025.

Virtual Machine (VM) with 16 GB memory on a Proxmox Hypervisor with an AMD Epyc 7402 CPU.

A. Reference Measurement

First, to evaluate the completeness and correctness of the *PONAlyzer* prototype, we capture a complete ONU activation process and subscriber provisioning, as defined in the OMCI specification [5], with *PONAlyzer*. At the same time, a commercial hardware-based PON analyzer, the *GPON Doctor 10000 S* by Telnet, simultaneously captures the OMCI messages in the ODN. Figure 6 shows how the hardware-based analyzer is connected to the ODN with two physical interfaces, each receiving dedicated downstream and upstream packets. For the downstream packets, the reference analyzer is connected to the left splitter receiving downstream duplicates, similar to the connected ONUs. A second splitter is added to duplicate the upstream packets to the analyzer in the upstream direction. This way, the hardware-based analyzer for the reference measurement captures all types of transmitted packets on the wire in both directions. The comparison of the *PONAlyzer* and reference measurement shows that all OMCI messages are correctly and identically captured. Therefore, our presented *PONAlyzer* solution fulfills all requirements for an OMCI protocol analyzer while eliminating the need for costly specialized hardware analyzers. However, our approach does not allow insights on physical layer provisioning and operations, *e.g.*, timing violations, key exchange or analog signal analysis.

B. Anomaly Detection

To verify the anomaly detection, we analyzed our testbed deployment without modifications and purposely injected OMCI messages to trigger anomalies. As a first occurrence, we disconnect the subscriber Ethernet port at the ONU and could subsequently observe a Loss of Signal (LOS) in our analyzer. The LOS event means that the connection to the ONU is broken on the subscriber side, yet the fiber connection itself is still operational. Also, injecting various malformed OMCI requests with *PONAlyzer* resulted in artificial anomalies. For example, one type of anomaly is the failed operation response by the ONU, indicated by the *result* field in the response greater than 0. Possible occurrences here are *instance exists*, *processing error* and *decoding error*. Especially the result type *decoding error* may reveal an ongoing faulty operation in the PON, whether by an attacker or misconfiguration. In this test scenario, the *decoding error* indicates the malformed OMCI injection. Also, our prototype detected a lack of responses by identifying missing transaction IDs. Another type of anomaly is the time analysis, as a high response time usually indicates an overloaded ONU. As described in Section IV, we could trigger a high received optical power anomaly by utilizing a short fiber with low signal attenuation.

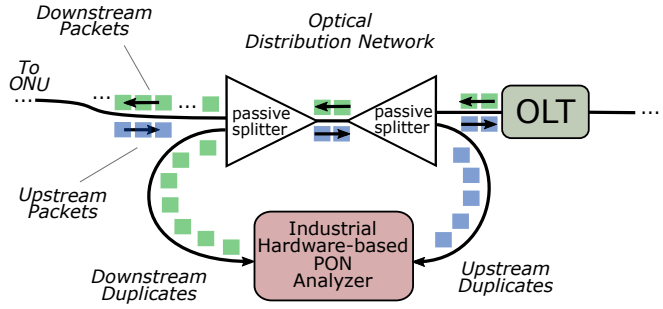

Fig. 6: Placement of reference hardware-based PON analyzer *GPON Doctor 10000 S*.

C. Exposing ONU States by Injection

To demonstrate the injection functionality of our prototype, we retrieve all stored data in an ONU by iteratively injecting requests for all MEs. At first, to execute the entire process, a *MIB Upload Request* message targeted at a default ME of each ONU is injected. On success, each ONU responds with a message containing the number of *MIB Upload Next Request* messages required for the remaining process. After this, the required number of *MIB Upload Next Request* messages is sequentially injected and then one response for each request is received. Each response contains the attributes and values of an ME stored inside the MIB of the ONU.

We were able to receive all stored information in the ONU, for example, the bandwidth allocation identifiers which leads to the bandwidth profiles as configured in the VOLTHA control plane. Some more examples of attributes obtained by injection include priority queue identifiers and the current operational status. Also, vendor information, such as the serial number of an ONU, is retrieved. Additionally, threshold values and signal levels of the deployed lasers are read from the ONU. The complete list of attributes stored in the ONU for each ME can be found in the OMCI specification [5].

D. Performance Measurements

To further evaluate the capabilities of injecting OMCI messages into the PON, we injected 20,000 *Get Request* OMCI messages within 10 s and increasing transaction IDs into the OLT, which then forwarded the messages to the ONU. Figure 7 shows, how the ONU is processing and answering the first 550 OMCI requests received while dropping the next 3425 requests for about two seconds. After that, the ONU is accepting OMCI messages again and starts responding to the next 553 requests, while dropping the subsequent 3498 ingressing requests again. Figure 7 depicts how the ONU

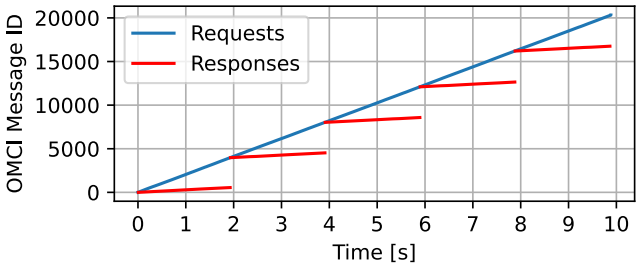

Fig. 7: Flooding the ONU buffer with OMCI request messages, with only around 500 messages processed in about 2 s.

Fridolin Siegmund, Stefano Acquaviti, Bjoern Nagel, Maik Rüder, Matthias Hollick, Ralf Kundel. *Open PONAlyzer: A Software-Defined Analysis and Testing Solution for Enhanced Security and Resilience in Passive Optical Networks* To appear in the Proceedings of IEEE Conference on Network Function Virtualization and Software Defined Networks 2025, IEEE, 2025.

overload is occurring 5 times during the test run of 10 s. Our measurement shows how the ONU processes requests in batches, indicating that the internal ONU buffer size is around 550 to 554 OMCI requests. Therefore, we can demonstrate the sensitivity of ONUs against DoS attacks on the OMCI layer, being caused by a sender on the shared PON medium, *e.g.*, *PONAlyzer*. Also, the internal processing capabilities of ONUs can be evaluated by *PONAlyzer*.

VI. Related Work

When considering resilience in PONs, multiple points in the access network are researched for proactive or reactive resilience improvements. Field Programmable Gate Array (FPGA) based solutions are researched for tapping into the ODN and capturing all transmitted messages such as OMCI and data plane traffic for security analysis [10], [11]. This tool is similar to the commercial PON analyzer utilized in this work. However, these solutions have the disadvantage of requiring additional hardware and cabling, lacking multi-fiber support, and causing downtime when splitters are added to the ODN. Another resilience improvement approach by monitoring is the measurement of the changes in reflections of the light waves in the fiber itself [12]. This way, faults such as broken fibers can be detected by the author's tool due to the missing peak reflection in case of a broken fiber. Also, the point in the fiber where the fault occurs can be measured by distance. Further, the resilience of the ODN can be increased by redundancy: As a hardware-based approach, several fiber links between subscriber ONUs and the OLT can be deployed [13]. Besides increasing the resilience in the ODN, the path of the subscriber data in the direction of the backbone can profit from improved resilience. By deploying multiple redundant links from the OLT to the termination node Broadband Network Gateway (BNG) and multiple BNG instances, the subscriber data can be steered by an SDN controller reactively to restore service after, *e.g.*, a link failure [14]. Generally, the separation of the data plane and control plane and the interfaces in between allows for new injection attacks as researched by [15]. The authors propose a framework designed to perform injection attacks targeting SDN architectures utilizing OpenFlow for security analysis.

VII. Conclusion and Future Work

In this work, we explored the use of SD-PON interfaces for the security analysis of the OMCI protocol, building upon the VOLTHA software stack as a foundation. Our prototype *PONAlyzer* demonstrates how OMCI messages can be analyzed through a user-friendly web-based interface, either via live traffic capture or by processing PCAP files at zero additional cost. Our tool enables operators and researchers to monitor PON networks and detect anomalies caused by faulty devices, misconfigurations, or malicious actors in the shared fiber infrastructure. Further, we demonstrated how subscriber ONUs can be tested in several scenarios by injecting OMCI messages from the *PONAlyzer* dashboard. The evaluation shows that *PONAlyzer* is capable of generating and injecting more than 2000 OMCI messages per second. However, *PONAlyzer* does not support the analysis of physical layer messages such as PLOAM as these messages are handled within the PON ASIC inside the OLT and are not exposed on the used interface.

To share our contribution with the community, *PONAlyzer* is publicly available as open-source software[†].

As future work, we plan compatibility tests with XGS-PON networks. Given the use of the same OMCI protocol stack, we expect only minimal adaptations to be necessary for XGS-PON and future upcoming PON generations.

Acknowledgement

This work has been supported by Deutsche Telekom through the Dynamic Networks 14 project and partly supported by the LOEWE initiative (Hesse, Germany) within the emergenCITY center [LOEWE/1/12/519/03/05.001(0016)/72]. We thank our colleagues and the anonymous reviewers for their input.

References

[1] ITU Telecommunication Standardization Sector, "G984: Gigabit-capable Passive Optical Networks (GPON): General characteristics," *ITU-T G*, vol. 984, 2003.

[2] ITU Telecommunication Standardization Sector, "G.9807.1: 10-Gigabit-capable symmetric passive optical network (XGS-PON)," *ITU-T G*, vol. 9807, 2016.

[3] ITU Telecommunication Standardization Sector, "G.989.2: 40-Gigabit-capable passive optical networks 2 (NG-PON2): Physical media dependent (PMD) layer specification," *ITU-T G*, vol. 989, 2014.

[4] ITU Telecommunication Standardization Sector, "G.9804.3: 50-gigabit-capable passive optical networks (50G-PON): Physical media dependent (PMD) layer specification," *ITU-T G*, vol. 9804, 2021.

[5] Sector, ITU Telecommunication Standardization, "G.988: ONU management and control interface (OMCI) specification," 2010.

[6] C. F. Lam, *Passive Optical Networks: Principles and Practice*. Elsevier, 2011.

[7] Open Networking Foundation and Linux Foundation, *Virtual OLT Hardware Abstraction v2.13*, 2025. [Online]. Available: https://docs.voltha.org/master/index.html

[8] N. Merayo, D. de Pintos, J. C. Aguado, I. de Miguel, R. J. Durán, P. Fernández, R. M. Lorenzo, and E. J. Abril, "Experimental validation of an SDN residential network management proposal over a GPON testbed," *Optical Switching and Networking*, vol. 42, p. 100631, 2021.

[9] Open Networking Foundation, *OMCI-Lib-Go v2.2.3*, 2023. [Online]. Available: https://github.com/opencord/omci-lib-go

[10] V. Oujezsky, T. Horvath, M. Jurcik, V. Skorpil, M. Holik, and M. Kvas, "FPGA Network Card and System for GPON Frames Analysis at Optical Layer," in *2019 42nd International Conference on Telecommunications and Signal Processing (TSP)*. IEEE, 2019, pp. 19–23.

[11] V. Oujezsky, T. Horvath, and M. Holik, "Security Incident Response Automation for xPON Networks," *Journal of Communications Software and Systems*, vol. 18, no. 2, pp. 144–152, 2022.

[12] X. Zhang, T. Yang, and X. Jia, "A PON Monitoring System Integrating Fault Detection and Localization," *IEEE Photonics Journal*, vol. 14, no. 5, pp. 1–7, 2022.

[13] C. Christodoulou and G. Ellinas, "Resilient architecture for optical access networks," *Photonic Network Communications*, vol. 41, pp. 1–16, 2021.

[14] F. Siegmund, P. J. Franz, B. Nagel, M. Rüder, L. Wernet, M. Hollick, and R. Kundel, "State-aware Subscriber Steering in Fiber Access Networks for Improved Resilience," in *2024 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. IEEE, 2024, pp. 1–6.

[15] B. E. Ujcich, U. Thakore, and W. H. Sanders, "ATTAIN: An Attack Injection Framework for Software-Defined Networking," in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2017, pp. 567–578.

[†]https://github.com/fridolinsiegmund/PONAlyzer