POET: A Self-learning Framework for PROFINET Industrial Operations Behaviour

Ankush Meshram[1(✉)], Markus Karch[2], Christian Haas[2], and Jürgen Beyerer[1,2]

[1] KASTEL Security Research Labs, Vision and Fusion Laboratory (IES), Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany
ankush.meshram@kit.edu

[2] Information Management and Production Control, Fraunhofer Institute of Optronics, System Technologies and Image Exploitation (IOSB), 76131 Karlsruhe, Germany
{markus.karch,christian.haas,juergen.beyerer}@iosb.fraunhofer.de

Abstract. Since 2010, multiple cyber incidents on industrial infrastructure, such as *Stuxnet* and *CrashOverride*, have exposed the vulnerability of Industrial Control Systems (ICS) to cyber threats. The industrial systems are commissioned for longer duration amounting to decades, often resulting in non-compliance to technological advancements in industrial cybersecurity mechanisms. The unavailability of network infrastructure information makes designing the security policies or configuring the cybersecurity countermeasures such as Network Intrusion Detection Systems (NIDS) challenging. An empirical solution is to self-learn the network infrastructure information of an industrial system from its monitored network traffic to make the network transparent for downstream analyses tasks such as anomaly detection. In this work, a *Python*-based industrial communication paradigm-aware framework, named *PROFINET* Operations Enumeration and Tracking (POET), that enumerates different industrial operations executed in a deterministic order of a *PROFINET*-based industrial system is reported. The operation-driving industrial network protocol frames are dissected for enumeration of the operations. For the requirements of capturing the transitions between industrial operations triggered by the communication events, the Finite State Machines (FSM) are modelled to enumerate the *PROFINET* operations of the device, connection and system. POET extracts the network information from network traffic to instantiate appropriate FSM models (Device, Connection or System) and track the industrial operations. It successfully detects and reports the anomalies triggered by a network attack in a miniaturized *PROFINET*-based industrial system, executed through valid network protocol exchanges and resulting in invalid *PROFINET* operation transition for the device.

Supported by topic Engineering Secure Systems of the Helmholtz Association.

S. Yu et al. (Eds.): TridentCom 2022, LNICST 489, pp. 3–19, 2023.
https://doi.org/10.1007/978-3-031-33458-0_1

Keywords: Network Security · Cyber-Physical System · Intrusion Detection

1 Introduction

The incremental advancement in technology since steam-powered manufacturing mechanization (the *first* industrial revolution) to the *third* industrial revolution of computer-driven process automation resulted in the *fourth* industrial revolution of cyber-physical systems. Industrial production systems of the 21st Century are designed for production cost reduction through efficient control of cyber-physical industrial components, realized through the adaptation of Ethernet technology in industrial networking. It blurred the separation between the *office networks* and the *industrial networks* to allow the personnel in the corporate office access a sensor in the production floor. The blurring of network separation led to cybersecurity vulnerabilities to industrial production, as demonstrated by recent cyber incidents from *Stuxnet* to *Triton* in the last decades [9]. The Advanced Persistent Threats (APT) modified the instruction exchanges within industrial processes, utilizing industrial protocols, to cause structural damage to components (*Stuxnet*) or Human, Societal and Environmental (HSE) hazards (*Industroyer/CrashOverride, Triton*). Continuous monitoring and analysis of industrial communication characteristics is required to detect the initiation of such attacks and reduce the dwell time to accelerate mitigation.

An anomaly-based Network Intrusion Detection System (NIDS) is one of the cybersecurity countermeasures that monitors the network traffic of an industrial production and learns the characteristics from its network infrastructure information to detect the deviations as *anomalies*. Germany's Federal Office for Information Security (BSI) in its cybersecurity recommendations on production networks [1] outlined anomalies in industrial networks and related categories of feature requirements for anomaly detection systems. The general requirements category collectively needs the anomaly detection to provide overview of all devices communicating in the network and identify the communication links along with the protocols used. Another requirement category emphasizes on the ability to detect unusual or exceptional activities in an industrial network, such as identification of new devices in the network, new protocols or changes in protocol among individual components, etc.

1.1 Problem Statement

In the absence of network infrastructure information, such as asset inventory and network policies, of an existing industrial system, designing the security policies or configuring the cybersecurity countermeasures such as NIDS is challenging. Insights into the industrial system's operation are required for efficient monitoring and timely incident, *cyber* and *physical*, response. Interpretation of industrial system operations from its communication network characteristics contributes to being vigilant of cyber threats aimed at industrial process disruption. An empirical solution is to self-learn the network infrastructure information (topology, assets

and communication links) and the characteristic behaviour from passive monitoring of industrial network traffic, in conjunction with an anomaly detection system to detect anomalies. The industrial network's topology, communication relations, assets and protocol data being exchanged during the industrial process operations are the network information to be extracted from the network traffic passively. The detection of the different industrial operations from start-up to process data exchange from the monitored network traffic in a systematic way for a self-learning approach is a challenge for *network transparency* (**Problem 1**).

In addition, the industrial operations executed in an order create the foundation for process data exchanges realizing the underlying intended process. Enumerating these operations through the analysis of multiple protocol communications observed in the traffic and tracking their executions helps to define the industrial system's operation behaviour. Monitoring the valid operations of devices, communication links or the industrial system would detect the adversarial actions in the context of protocol specifications of employed industrial networking technology such as *PROcess FIeld NETwork (PROFINET)* [17]. The representation and enumeration of a *PROFINET* system's operations from monitoring the multiple protocol communications in the network traffic for self-learning its *industrial operation behaviour* is another challenge (**Problem 2**).

There are multiple research works in the literature and commercial NIDS solutions that model the message exchanges of industrial network protocols based on corresponding protocol specifications, and the deviations are classified as anomalies [6]. In particular, Snort rules for *MODBUS* [10], Bro rules for *DNP3* [8] and specification-based IDS for *GOOSE* [4] check for the validity of packet fields and communication exchanges. However, the effect of protocol exchange on the industrial system is not modeled. A valid protocol exchange could have adverse effect on the industrial operations which hasn't been modeled in any of the reported works.

1.2 Proposed Solution

Different industrial networking operations of a *PROFINET*-based industrial system executed with different industrial protocols are mapped to corresponding industrial operations from the start up to the process data exchange operation. *PROFINET*'s specifications are followed to correctly map network protocols to detect networking operations and their constituent stages in a Python-based framework that passively captures the network traffic, and extracts relevant information to make the industrial network transparent for analysis. The network information made available through the developed network transparency solution are utilized to enumerate different industrial operations whenever they occur. An industrial system's operation behaviour is considered at device-level, connection-level and system-level to track operational state changes in devices, established process exchange communication links and the overall system. Graph-represented Finite State Machines (FSM) are conceptualized for each device, connection and industrial system, where nodes represent the stages of industrial operations and the edges are the transitions that are triggered by the events observed in the extracted network information from the traffic. A *Python*-based

framework, named *PROFINET* Operations Enumeration and Tracking (POET), that extracts the network information from *PROFINET*-based industrial system's network traffic to instantiate appropriate FSM models (Device, Connection or System) and track the industrial operations is developed. On a miniaturized *PROFINET*-based industrial demonstrator (*Festo Demonstrator*), the POET is successfully employed to detect anomalies triggered by a network attack targeted at an industrial component, executed through valid *PROFINET Discovery and Configuration Protocol (PN-DCP)* exchanges and resulting in invalid *PROFINET* operation transition for the device.

In the next section, a brief overview of *PROFINET* technology followed with information on miniaturized *PROFINET*-based industrial system under consideration and simulated industrial attack scenario is provided. In Sect. 3, the enumeration of different networking operations executed through different network protocols in *PROFINET* systems is provided. The proposed framework to enumerate and track the *PROFINET* operations from the analysis of traffic data is summarized in Sect. 4. The Sect. 5 presents the framework's usage for anomaly detection along with brief discussion, and conclusion in Sect. 6.

2 Preliminaries

2.1 PROFINET

Proprietary fieldbus protocols were developed to satisfy the strict requirements for real-time data transmission and deterministic communication for industrial network operations such as *PROFIBUS*, *Modbus*, etc. *PROFINET* is the result of adapting *PROFIBUS* to real-time technology and standardized in IEC 61158 & IEC 61784. *PROFINET* has 18% market share in the industrial networks that are installed globally in 2021 as compared to 17% *EtherNet/IP* [11]. Additionally, *PROFINET* is the leader of Industrial Ethernet technology in the European market which concluded its selection as the industrial system under consideration for the presented work.

There are two real-time properties of *PROFINET* communication: (a) non-synchronized real-time communication (RT), and (b) synchronized real-time communication (IRT). Within *PROFINET*, process data and alarms are transmitted with RT communication with bus cycle times in the range of 50–100 ms. Isochronous data transfer with IRT communication is used in applications such as motion control requiring bus cycle times in range of microseconds, $<$1 ms. In addition, *PROFINET* defines different classes of components characterized by their functionality and participation at different stages of industrial communication - IO Controller, IO Supervisor and IO Device. An IO Controller is the component with *master* functionality that executes the automation program, typically a Programmable Logic Controller (PLC). It participates in parametrization, cyclic/acyclic data exchange and alarm processing with connected field devices. An IO Supervisor is used for the commissioning and diagnostic purposes, generally a programming device, personal computer or Human Machine Interface (HMI). An IO Device is a field device in the vicinity of process with

slave functionality that sends process data and critical statuses (alarms & diagnostics) to connected IO Controller(s) via *PROFINET*. The transmission of data from an IO Controller/Supervisor to an IO Device is designated as *output data* whereas IO Device to IO Controller/Supervisor is *input data*. *PROFINET* utilizes the provider-consumer model of communication for I/O data exchange between controllers and devices, as well as parametrization and diagnosis information exchange between supervisors and devices.

An IO Device comprises of an Ethernet interface for communication and physical/virtual modules to handle the process data traffic. The device model of an IO Device consists of *slots*, *subslots*, *modules*, *submodules* and *channels*. The slot and subslot designates the insert slot of a module and submodules in an IO field device, respectively. The module provides the structuring, and contains at least one submodule which always holds the process data with status information. The data within the submodule is addressed using an index. Cyclic IO data in submodule are accessed through slot/subslot combinations, whereas, acyclic read/write services utilize slot, subslot and index.

2.2 System Under Consideration

The quality and characteristics of dataset employed in the development of ICS cyber threat detection methods play an important role in driving the Industrial Cybersecurity research. In the quest for finding the solutions to the aforementioned challenges, a *PROFINET*-based industrial system is used for developing and evaluating the proposed solutions. A *PROFINET*-based scaled-down industrial system with real industrial components and fully functional networking infrastructure, labeled as *Festo Demonstrator* is employed for the reported analysis. The network attacks targeted at the *Festo Demonstrator*'s underlying network and process operations are scripted, executed and resulting anomalies are passively captured from the network traffic. A systematic Python-based framework passively captures the network traffic from *PROFINET*-based system and extracts relevant information to make the industrial network transparent for downstream analyses such as anomaly detection.

Network Communication. The process scenario realized in the *Festo Demonstrator* is a simplified painting process. It is controlled through network communications between PLCs, I/O devices, actuators and process-associated sensors, and PLCs with Manufacturing Execution Systems (MES) and HMI. In Fig. 1, the network infrastructure of the *Festo Demonstrator* is shown. All the components are connected in STAR topology with the *Network Switch* at the center. The PLCs communicate to bus couplers and motors through *PROFINET* protocol, whereas PLCs to *HMI* communication is through *S7Comm* protocol. The process information is relayed to *MES* through *OPC UA* protocol from *OPC UA*-compatible PLCs. In case of non *OPC UA*-compatible *PLC-3*, an *OPC UA* gateway collects information from *PLC-3* through *S7Comm* and relays it to *MES*. *RDP* protocol is used to connect a *Tablet* to *MES* server to visualize the process execution.

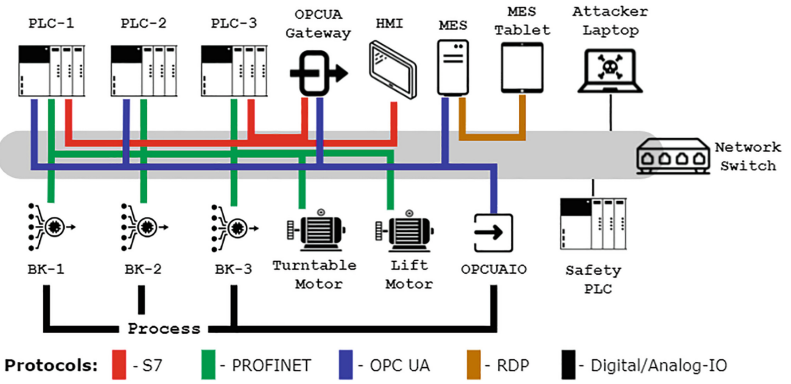

Fig. 1. The networking infrastructure of the Festo Demonstrator.

2.3 Industrial Network Attack Scenarios

An *adversary* is assumed to have gained access to the *Festo Demonstrator*'s network infrastructure. Within *PROFINET* networks, the components are addressed through their *logical names* for process data exchange via the unencrypted *PROFINET* protocol. An *adversary* exploits the *PROFINET* protocol design flaw and changes the *logical name* of *Turntable-Motor* to *"ufo"* via the *PN-DCP* protocol as shown in Fig. 2. As a result, the other industrial components are not able to identify the component with the name *"Turntable-Motor"* and the process stops.

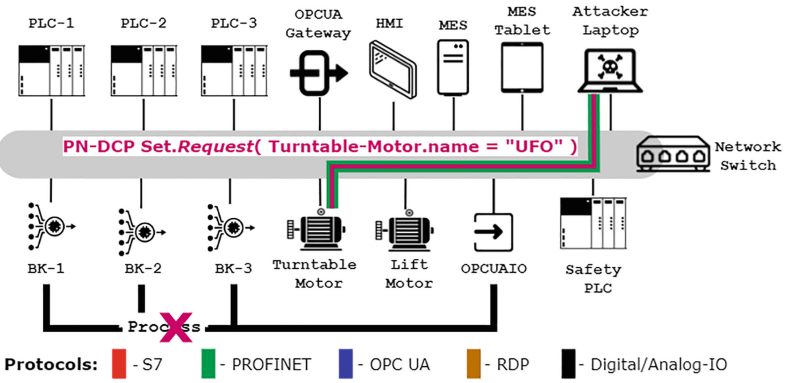

Fig. 2. The Rename Attack on the Festo Demonstrator.

3 Network Operation Enumeration for PROFINET

Configuration and commissioning of *PROFINET*-based automation systems must follow certain mode of operations in a strict order. It begins with the System Engineering operation where an automation project is configured in an engineering tool. General System Description (GSD), an *XML* file provided by every device manufacturer, contains configuration information for parametrizing the devices for real systems. In addition, each device is assigned a logical name to address it within the *PROFINET* communication. Within the System Engineering mode, an IP address is assigned to each device for communication. Transmission intervals are defined for cyclic data exchange between controller and devices. After system engineering is completed, the configuration information is downloaded to the controller. As soon as the automation system is powered on (or reset), Neighbourhood Detection, Address Resolution and System Startup are the operations followed in the same order, as shown in Fig. 3. With Address Resolution, the controller uses the system configuration information to assign the IP addresses to the devices identified through their pre-assigned logical names. System Startup operation mode is initiated by the controller to establish connection with devices and configure their I/O parameters. When the I/O parametrization ends successfully, the controller and devices step into Data Exchange mode to transmit process data, alarms and diagnostic information throughout the network.

Every mode of operation in *PROFINET*-based automation system, from configuration to commissioning, is accomplished through a complementary network operation involving specific network protocols. The networking operations and the specific network protocols exchanges driving the operations are summarized.

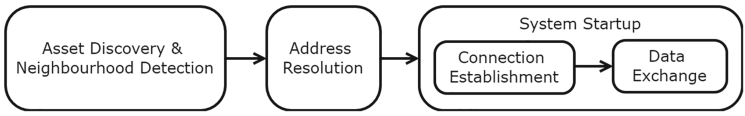

Fig. 3. Networking operations of an industrial system.

Asset Discovery and Neighbourhood Detection. After the automation system is powered on, the field device's MAC interface and its Physical Device Management (PDev) gets activated to start transmitting the parameters. PDev contains hardware-level information such as interface name, switch port data, interface and Port MAC addresses and retentively stores IP address and logical name assigned to the device. Port information is used by devices to determine their neighbours on port-by-port basis. Neighbourhood Detection is accomplished through *Link Layer Discovery Protocol (LLDP)* services. *LLDP*-capable devices communicate with their connected neighbours to cyclically exchange addressing information and consequently determine their physical location. *LLDP* frames are dissected to identify device names, number of switch ports

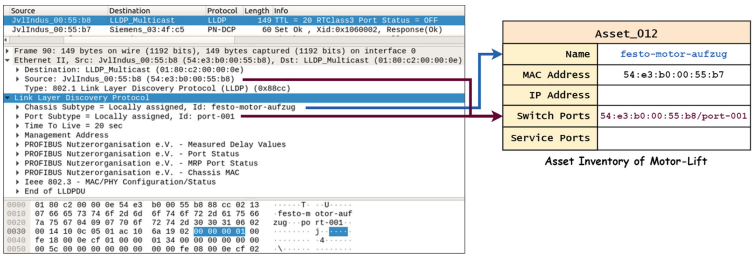

Fig. 4. The demonstration of *Lift-Motor* device's Asset Inventory filled with information from *LLDP* frame (*Wireshark* snippet).

and their MAC addresses for the Automated Asset Inventory, developed for the analysis in the reported work. Figure 4 shows an example of attributes added for the device *Lift-Motor* of the *Festo Demonstrator*.

Address Resolution. Before the *PROFINET*-based automation system starts up and field devices start communicating, an IP address needs to be assigned to all devices by the controller. An IO Device is identified through its '*NameOfStation*' information stored in its PDev and an IP address defined during System Engineering mode is assigned to it. Address Resolution networking operation for every device takes place step-by-step as follows: *(1)* Controller starts with name resolution and checks for device with configured name through '*DCP Identify*' service of *PN-DCP*, *(2)* Address Resolution begins with checking if the IP address already exists to avoid assigning same IP address twice through *ARP*, and *(3)* At the end of networking operation, the IP address is assigned to configured device through '*DCP Set*' service of *PN-DCP*. The schematic communication order for Address Resolution between *PLC-3* and *Lift-Motor* is shown in Fig. 5. The information dissected from an *Address Resolution Protocol (ARP) request* packet is used to add the IP address to controller assets. Information dissected from *PROFINET Discovery and Configuration Protocol (PN-DCP) Set* is used to add IP address information to configured devices. The Address Resolution networking operation performed by *PN-DCP* and *ARP* execute Address Resolution *PROFINET* operations.

Connection Establishment. System Startup operation begins with establishment of communication relationships between the controller and devices via *PROFINET Context Manager (PN-CM)* protocol communication exchanges. Through these established communications the controller transmits all the parameters for process data exchange to the internal module of devices. The process model and associated parameters for devices participating in the process are engineered & defined during System Engineering operation.

The '*connection*' between an IO Controller and an IO Device is established in an 'Application Relationship (AR)' uniquely identified by an `ARUUID`. Within

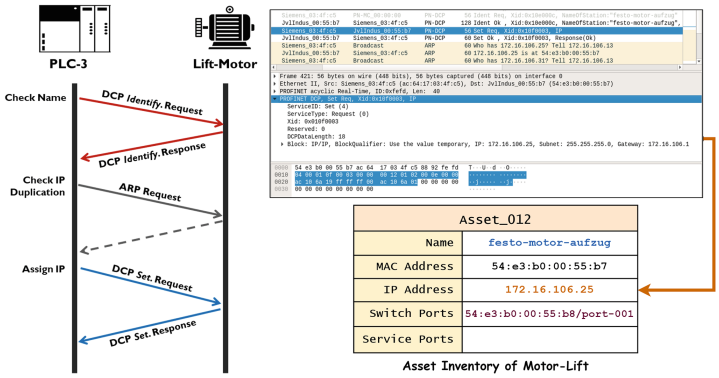

Fig. 5. The demonstration of *Lift-Motor* device's Asset Inventory from *PN-DCP* frame.

this *AR*, different 'Communication Relationship (CR)' are established for different data exchanges. An application can access data only through *CR*s established in an *AR*. *PROFINET* offers *PN-CM* protocol to handle the Connection Establishment network operation. The *PN-CM* network operation uses *UDP/IP* channel to transmit following frames in the strict order for establishing '*connection*' between controller and device:

- *Connect* frames establish *AR* and *CR*s channels.
- *Write* frames parametrize the device *submodules*.
- *DControl* frames mark the end of parametrization from the controller.
- *CControl* frames mark the validation check of parameters, data structure build up and application readiness from the device.

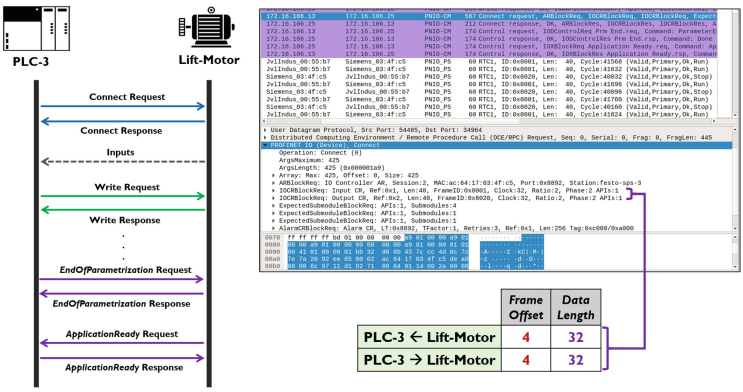

Fig. 6. The Connection Establishment handshake between *PLC-3* and *Lift-Motor*.

The first successful exchange of I/O data after *CControl* frames mark the end of *PROFINET*'s System Startup operation mode. The schematic

communication order for Communication Establishment between *PLC-3* and *Lift-Motor* is shown in Fig. 6. The GSD of a device contains modules and submodules information reflecting the position of process data within the payloads. Since in a self-learning analysis from network traffic the GSD isn't accessible, the aforementioned information is extracted through reading and interpreting *Connect* frames. *PN-CM* frames are dissected to extract the input and output data specifications for device's *submodule*s. It contains the data format type, order of data (endianness), length of data and the position of data within the payload bytes.

These specifications are used by the Data Exchange network operation's *PROFINET Input/Output (PNIO)* frames to extract process bytes. The '*connection*' between controller and device following a Connection Establishment network operation guides building the logical network topology of the system. These separated ISO layer connection state information between network assets is maintained throughout the automation system's runtime and deviations are reported.

Data Exchange. Once the System Startup operation establishes *AR* and data specific *CR*s between devices and controller, the connection-oriented communication channel is set for exchange of cyclic process data, acyclic diagnostic data and alarms. *PNIO* protocol defines the format and context for data exchange. Cyclic *PNIO* frames are sent unacknowledged between controller and devices. After *CControl* frames are acknowledged by the controller, the first valid exchange of I/O data with `IOPS=GOOD` ends Connection Establishment operation. Data Exchange operation begins with cyclical exchange of process data at configured/parametrized fixed intervals. *PNIO* cyclic data frame is transmitted in real-time with `Ethertype=0x8892` The process data bytes from `Data` field are extracted using information from *Connect* frame dissection, as shown in Fig. 7.

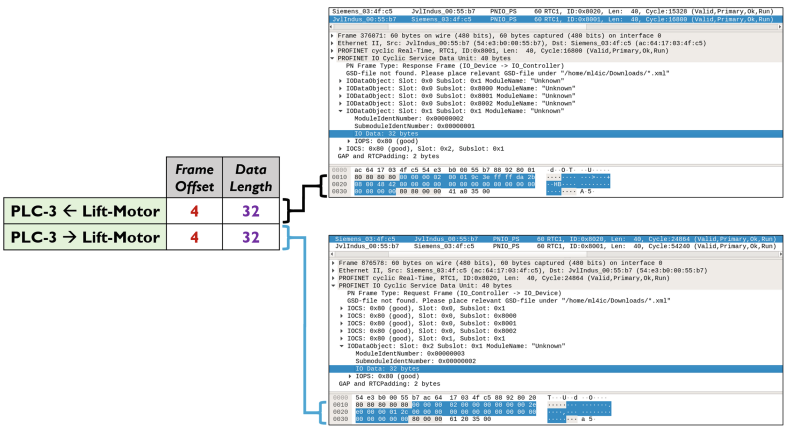

Fig. 7. The demonstration of extracting process data between *PLC-3* and *Lift-Motor* from *PNIO* frames (*Wireshark* snippet), using parameters extracted from *Connect* frame.

Enumeration of *PROFINET* operation modes is performed by passively monitoring/analysing the network traffic and identifying the associated network operation stage-by-stage. System Engineering mode is performed offline, hence, it can't be enumerated through analysing the network traffic.

4 PROFINET Operations Enumeration and Tracking

Through monitoring an industrial system's communication network, its characteristics are observed to build normal operations behaviour baseline. Deviations from the baseline behaviour could be triggered by *physical* or *cyber* threats. A systematic framework is needed to enumerate operations extracted from network traffic and track them to report deviations.

PROFINET-based automation systems follow strict order of operations. The operation mode and corresponding network operations with associated network protocols have been outlined in Sect. 3. All of those network operation's dissected information are combined to systematically iterate over the *PROFINET* operation modes as and when there occurrences are observed through network analysis. *PROFINET* devices transit through *PROFINET* operations to establish *connection*s between them for cyclic and acyclic data exchange. These transitions also govern transitions in *PROFINET* connections, which constitutes the logical topology and industrial process behaviour of the *PROFINET* system.

Finite State Machines(FSM) [7] are widely used for protocol specification (e.g. *TCP/IP* [15]) [3], where the valid *transitions* and *states* of message exchanges are defined. For the requirements of capturing the transitions between industrial operations triggered by the communication events, the FSMs are modelled to enumerate the *PROFINET* operations of the device, connection and system. *PROFINET* standard, the informative handbook on *PROFINET* [13] and empirical information collected from analysing real-world *PROFINET*-based system communications are interpreted to model the operations in FSMs.

In the next subsections, each FSM is described with the overview of states and events triggering the transitions outlined in its state diagram. The transitions which are modelled based on empirical information are distinguished by dashed edges and details are presented.

4.1 FSM *PROFINET* Device

States and Transitions. FSM Device enters with *Active* state as soon as the system is powered on, shown in Fig. 8. It transits either to *Neighbourhood Detection* or *Name Resolution* state depending on the event triggered. FSM follows through the transitions as and when the triggering event is detected in network traffic.

LLDP frames are periodically transmitted by *PROFINET* device as per their `Time-To-Live` value for consistent *LLDP* information validation. Hence, *Neighbourhood Detection* state can be arrived from any other state whenever *detect_neighbours* event is triggered by *LLDP* frame. Consequently, all the

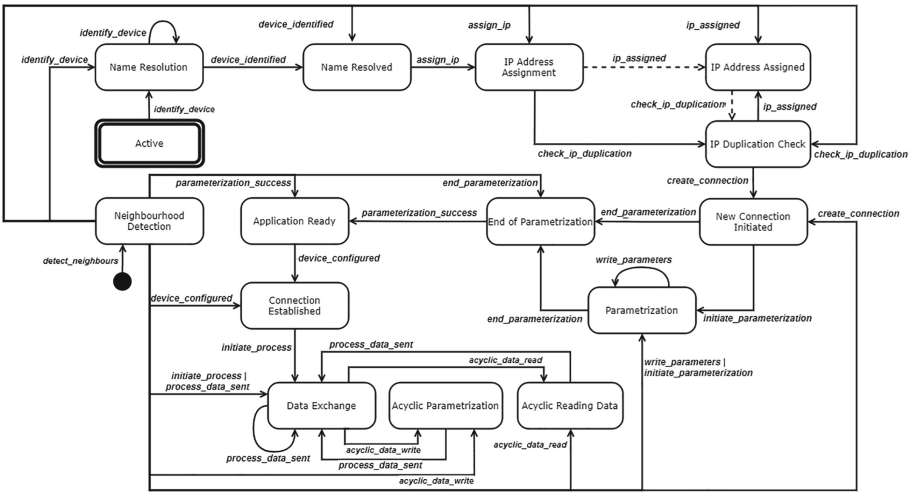

Fig. 8. *PROFINET* Device State Machine.

states are reachable with corresponding triggering events from *Neighbourhood Detection* state.

Through network traffic analysis of *PROFINET*-based systems with Siemens PLC, transitions - *IP Address Assignment* to *IP Address Assigned* and *IP Address Assigned* to *IP Duplication Check* - have been modelled. Deviating from transitions mentioned in the literature, the *PROFINET* devices checked for IP duplication with *Gratuitous ARP* after the IP address has been assigned to them. These transitions are also verified on different *PROFINET*-based systems.

Relationship Setween States and *PROFINET* Operations. *Neighbourhood Detection* state constitutes Asset Discovery & Neighbourhood Detection *PROFINET* operation. States *Name Resolution*, *Name Resolved*, *IP Address Assignment*, *IP Address Assigned* and *IP Duplication Check* constitute Address Resolution *PROFINET* operation. *PROFINET*'s Connection Establishment operation consists of states *New Connection Initiated*, *Parametrization*, *End Of Parametrization*, *Application Ready* and *Connection Established*. States *Data Exchange*, *Acyclic Parametrization* and *Acyclic Reading Data* reflect Data Exchange *PROFINET* operation.

4.2 FSM *PROFINET* Connection

States and Transitions. A *PROFINET* connection is established between *PROFINET* devices through *PROFINET*'s *PN-CM* protocol handshake. The cyclic and acyclic data exchange takes place through this connection. Hence, each connection is identified through MAC addresses of participating *PROFINET* devices. FSM Connection enters with *Connection Creation* state as soon as *Connect request* frame is sent by the controller, shown in Fig. 9. FSM follows through

the transitions as and when the triggering event is detected in network traffic. In particular, events *output_process_data_sent* and *input_process_data_sent* are triggered by transmission of *PNIO* frames from controller to device and vice versa, respectively.

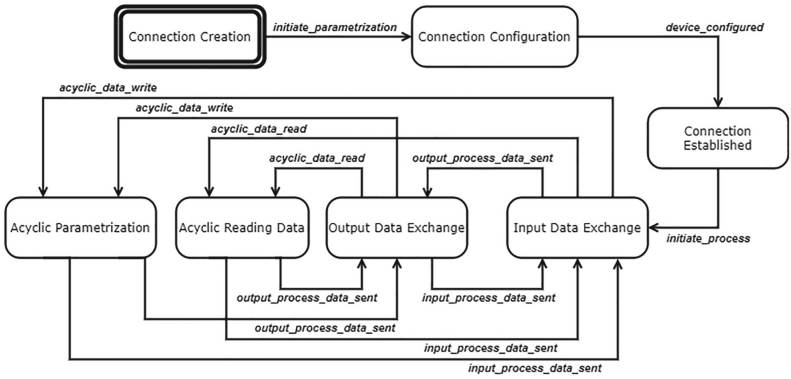

Fig. 9. *PROFINET* Connection State Machine.

Relationship Between States and *PROFINET* Operations. States *Connection Creation*, *Connection Configuration* and *Connection Established* constitute Connection Establishment *PROFINET* operation. *PROFINET*'s Data Exchange operation are reflected in states *Input Data Exchange*, *Output Data Exchange*, *Acyclic Parametrization* and *Acyclic Reading Data*.

4.3 FSM *PROFINET* System

States and Transitions. FSM System initializes with *Inactive* state and transits into *Powered On* as soon as *PROFINET* traffic triggers event *pn_traffic_detected*, show in Fig. 10. FSM follows through the transitions as and when the triggering event is detected in network traffic. Event *all_connections_established* is triggered when all the FSM *PROFINET* Connection instances have arrived in state *Connection Established*.

Relationship Between States and *PROFINET* Operations. State *Powered On* reflects either *PROFINET* operation Asset Discovery & Neighbourhood Detection or Address Resolution if the event *pn_traffic_detected* is triggered by *LLDP* or *DCP Identify request* frames, respectively. State *Asset Configuration & System Startup* reflects Connection Establishment *PROFINET* operation. *PROFINET*'s Data Exchange operation of cyclic and acyclic data transmission is reflected in state *Data Exchange*.

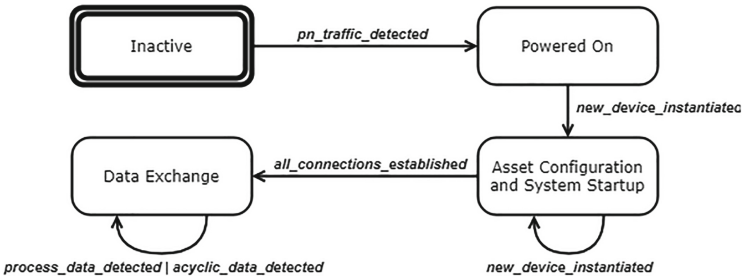

Fig. 10. *PROFINET* System State Machine.

4.4 Framework

PROFINET system, connection and device FSMs are realized in a *python*-based framework, termed as PROFINET Operations Enumeration and Tracking (POET), for enumeration and tracking *PROFINET*-based industrial network communication. It is implemented with *pytransitions* [14] and logs transitions in FSM instances of *PROFINET* System, Connection and Device continuously. Each FSM *PROFINET* System instance is identified by a name given while initialization, whereas the device name extracted from network traffic (*DCP Identify request*/*LLDP* frame) is used for identifying FSM *PROFINET* Device instance. FSM *PROFINET* Connection instance is identified by the connection identifier created from concatenating MAC addresses of devices.

POET clearly satisfies the BSI's general requirements category for an anomaly detection system to identify communicating devices, protocols and communication links (modeled as *PROFINET* Connection) in the industrial network.

5 Anomaly Detection with POET

Industrial networks are vulnerable to different threat behaviours, each utilizing different techniques to exploit industrial network characteristics. *MITRE ATT&CK for ICS* [16] is a knowledge base of such industrial system targeted threat behaviours, collected through cyber threat intelligence reports of known cyber incidents. Some threat behaviours (such as *Modify Parameter*, *Denial of Service*) are targeted at the industrial network operation to disrupt the underlying industrial process.

Pfrang et. al. [12] outlined threat scenarios targeted at real-world *PROFINET*-based systems with two different techniques to take over control of a *PROFINET* device. Within *PROFINET* networks, the devices are identified through the assigned logical name for process data exchange. The first attack of [12] demonstrated how an attacker changes the name of a device utilizing *PN-DCP* protocol and disconnecting it with other devices. A similar attack, the '*Rename Attack*' was performed on the *Festo Demonstrator* as outlined in Sect. 2.3 and POET was employed to monitor network traffic. The attack triggered events which aren't

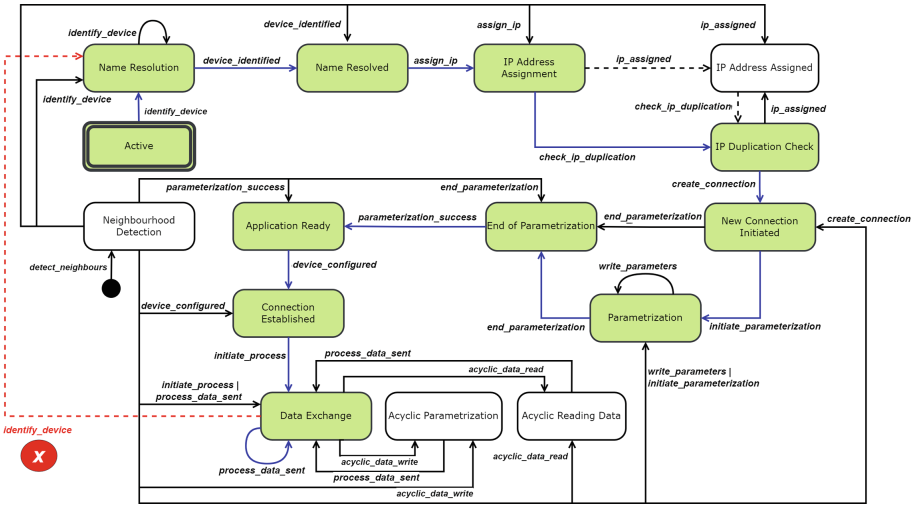

Fig. 11. The detection of Rename Attack on the *Festo Demonstrator* with the POET.

allowed for FSM Device instance of 'festo-motor-scheibe' and were reported in POET's logger, as shown in Fig. 11. The blue edges are the *valid PROFINET* operation transitions triggered by the appropriate protocol events. As per the *PN-DCP* protocol specification, the packets (x) are *valid*, however, they violate *PROFINET* device's *valid* operation transition represented as red edges, and thus detected.

The second attack of Pfrang et. al. [12] is essentially disrupting *PROFINET* network operations by establishing new connection with device. This action initiates Connection Establishment *PROFINET* operation which isn't valid transition state for FSM Device instance. Employing POET in such scenarios would also detect this attack and enhance visibility to unwarranted events in *PROFINET* networks with explanation.

POET's capability to detect the two attacks outlined in [12] satisfies the BSI's category of requirements to detect unusual or exceptional activities in an ICS network. Any other attacks that would violate the validity of an industrial operation through protocols other than *PN-DCP* and *PN-CM* would be detected by POET.

5.1 Discussion

Protocol-analysis based IDS have been proposed for industrial protocols such as *DNP3*, *Modbus/TCP*, *GOOSE*, etc. [5], where protocol specifications are utilized to build system profile and the deviations are reported. The proposed FSM-based framework, POET, can be categorized along with them as Protocol-analysis based IDS for *PROFINET*.

POET is the first-of-its-kind Protocol-analysis based IDS for *PROFINET* to incorporate empirical behaviour of *PROFINET* system collected from real-world

systems. The current version of POET uses empirical information collected from *PROFINET* systems incorporating Siemens PLC, it could vary with other PLC environments (e.g. CODESYS [2]), devices and can be adapted. The extraction of network events triggering the transitions of industrial operations would be adapted to the new protocol communication stack.

POET offers insights into the operations of *PROFINET*-based systems at ascending granularity of system, connection between devices and device. This granularity helps to specify events proficiently to be used in downstream analysis for anomaly detection. For example, Pfrang et. al. [12] outlined enhanced Snort for *PROFINET* which was used to detect the two attacks mentioned in their work. It can be integrated with POET to trigger alarms when unwarranted transitions occur.

At the end, we demonstrated a successful workflow to interpret an industrial protocol specification and the empirical information collected from its real-world industrial system usage to design a Protocol-analysis based IDS. A similar workflow could be utilized for another industrial protocol such as *EtherNet/IP* utilizing the outlined FSM models at different granularities of system, connection and device. The FSM models would have to be adapted for states and triggering events.

6 Conclusion

The *PROFINET* network traffic is mapped to different *PROFINET* operations for interpreting the underlying status of industrial communication. In Sect. 3 the solution to **Problem 1** is outlined, where different protocols associated with *PROFINET* operations are mapped to *PROFINET* network operations. For every network operation, the role it plays within *PROFINET*-based automation system communication and the network protocol utilized to achieve the goal has been presented. Thus, satisfying the BSI's general requirements for network transparency.

In addition, the protocol associated communication behaviour and their detection through protocol frame analysis has been outlined. As the solution to **Problem 2**, Sect. 4 modelled operations of *PROFINET* system, connection and device as Finite State Machines (FSM) to systematically enumerate and track *PROFINET* operations. *PROFINET* Device, Connection and System FSMs are realized in a *python*-based framework named PROFINET Operations Enumeration and Tracking (POET). Its successful usage as Protocol-based IDS in detecting cyber attack on real-world *PROFINET* demonstrator has been presented in Sect. 5. This demonstrates POET as an anomaly detection solution that satisfies the BSI's requirement to identify unusual or exceptional activities in an ICS network. In conclusion, the challenge of self-learning *PROFINET*-based industrial communication networks is solved through interpretation of network traffic to *PROFINET* operations. The workflow developed to interpret an industrial protocol's specification with the empirical information from its real-world usage to develop an anomaly detection system can be replicated further to other industrial networking technology.

References

1. BSI: Monitoring and anomaly detection in production networks. Technical report, Federal Office for Information Security (BSI) (2019)
2. codesys: Codesys profinet (2017). https://www.codesys.com/products/codesys-fieldbus/industrial-ethernet/profinet.html
3. Holzmann, G.J.: Design and validation of protocols: a tutorial. Comput. Netw. ISDN Syst. **25**(9), 981–1017 (1993)
4. Hong, J., Liu, C.C., Govindarasu, M.: Detection of cyber intrusions using network-based multicast messages for substation automation. In: ISGT 2014, pp. 1–5. IEEE (2014)
5. Hu, Y., Yang, A., Li, H., Sun, Y., Sun, L.: A survey of intrusion detection on industrial control systems. Int. J. Distrib. Sensor Netw. **14**(8), 1550147718794615 (2018)
6. Kippe, J., Karch, M.: Angriffserkennungssysteme in ics netzwerken. In: Deutschland, Digital, Sicher : 30 Jahre BSI : Tagungsband zum 17. Deutschen IT-Sicherheitskongress. pp. 1–11. SecuMedia Verlag (2021). 46.23.04; LK 01
7. Kleene, S.C., et al.: Representation of events in nerve nets and finite automata. Automata studies **34**, 3–41 (1956)
8. Lin, H., Slagell, A., Di Martino, C., Kalbarczyk, Z., Iyer, R.K.: Adapting bro into scada: building a specification-based intrusion detection system for the dnp3 protocol. In: Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop, pp. 1–4 (2013)
9. Makrakis, G.M., Kolias, C., Kambourakis, G., Rieger, C., Benjamin, J.: Vulnerabilities and attacks against industrial control systems and critical infrastructures. arXiv preprint arXiv:2109.03945 (2021)
10. Morris, T., Vaughn, R., Dandass, Y.: A retrofit network intrusion detection system for modbus rtu and ascii industrial control systems. In: 2012 45th Hawaii International Conference on System Sciences, pp. 2338–2345. IEEE (2012)
11. Nideborn, J.: (2021). https://www.hms-networks.com/news-and-insights/news-from-hms/2021/03/31/continued-growth-for-industrial-networks-despite-pandemic
12. Pfrang, S., Meier, D.: On the detection of replay attacks in industrial automation networks operated with profinet io. In: ICISSP, pp. 683–693 (2017)
13. Popp, M.: Industrial communication with PROFINET. Profibus Nutzerorganisation (2014)
14. Pytransitions: Pytransitions/transitions: A lightweight, object-oriented finite state machine implementation in python with many extensions. https://github.com/pytransitions/transitions
15. Stevens, W.R.: TCP/IP Illustrated, vol. I: The Protocols. Pearson Education India (1993)
16. Strom, B.E., Applebaum, A., Miller, D.P., Nickels, K.C., Pennington, A.G., Thomas, C.B.: Mitre att&ck: design and philosophy. Technical report (2018)
17. e. V. (PNO), P.N.: Profinet system description. Technical report, PROFIBUS & PROFINET International (PI) (2018)