http://dx.doi.org/10.14236/ewic/ICS2018.13

Bro in SCADA: dynamic intrusion detection policies based on a system model

Justyna J. Chromik
University of Twente,
the Netherlands
j.j.chromik@utwente.nl

Anne Remke
University of Twente, the Netherlands
University of Münster, Germany
anne.remke@uni-muenster.de

Boudewijn R. Haverkort
University of Twente,
the Netherlands
b.r.h.m.haverkort@utwente.nl

We present an online monitoring tool for SCADA systems based on the network monitor Bro, which can be used locally at field stations. The tool generates alerts when suspicious and erroneous commands and sensor readings are detected. It can hence been seen as a local Intrusion Detection System, as well as an safety enhancement. It maintains a model of the local system, which is updated with incoming packets containing sensor readings and commands. Focusing on the protocol IEC-104, a parser was developed and the packet content was directly fed into the system model. Adaptive policies are implemented in Bro, which formulate physical constraints and safety requirements and allow to check whether SCADA traffic complies to these rules in real time. A case study with a real IEC-104 traffic trace shows the feasibility of our approach.

Intrusion Detection System, process-aware, SCADA, IDS, power distribution

1. INTRODUCTION

Safe and reliable critical infrastructures need to be at the core of our modern society. In order to ensure their stability, they have to be continuously monitored. Geographically distributed critical infrastructures, such as electricity distribution and transmission, are monitored and controlled by SCADA (Supervisory Control and Data Acquisition) systems. By allowing remote control, operators save time and companies save money when managing the distribution grid. However, at the same time, such automation introduces new opportunities for malicious parties to disrupt the process.

Although SCADA systems include a communication network, their direct interaction with the physical process requires additional security methods specifically tailored to the safety requirements and physical constraints of the process. For example, when an attacker hacks into a control room by means of phishing (ICS-CERT (2016)), or a USB stick (ICS-CERT (2010)), and sends legitimate commands, from a legitimate source, that can easily result in disrupting the physical process. The detection of SCADA attacks aiming at disrupting the physical process is challenging if one only relies on network Intrusion Detection Systems (IDSes), implementing e.g. whitelisting or log-analysis (Barbosa et al. (2013); Hadžiosmanović et al. (2012)). Even if the mentioned IDSes are designed specifically for SCADA protocols, malicious control commands sent in a legitimate format would remain undetected. For example, the detection of a sequence attack (Caselli et al. (2015)), requires a complementary, process-aware system, which analyzes each command in the context of the physical system state.

The concept of process-aware monitoring has recently been investigated in the context of industrial control systems and electricity transmission and distribution systems (Nivethan and Papa (2016a); Lin et al. (2013); Fovino et al. (2010); Hadžiosmanović et al. (2014); Cárdenas et al. (2011)). It usually involves deep-packet inspection and processing that content, e.g., by comparing process variables to predefined thresholds ensuring system safety. Previously, we proposed a process-aware monitoring approach for field stations, which relies on a model of the directly controlled system (Chromik et al. (2016a,b)). Additionally, physical constraints, e.g., Kirchoff's law for nodes in the system, and safety requirements, e.g., maximum current allowed on the power lines, are formalized for this system.

This paper paves the way to a monitoring tool based on the network security monitor Bro (Paxson (1999)), which can be used for intrusion detection. Maintaining a model of the state of the physical system, the proposed tool checks physical constraints and safety requirements online and locally for field stations. Moreover, the tool alerts the

system operator whenever an incoming command at that location could disrupt the physical process or when a sensor measurement deviates from the model more than a predefined threshold.

The presented monitoring tool is intended for use in field stations to detect attacks coming from a corrupted control room or via a corrupted communication channel. Examples are insider attacks in the control room, and man-in-the-middle or replay attacks in the communication channel. We emphasize that our approach should be used in addition to existing centralized mechanisms, such as the Energy Management Systems, and is not as a replacement. In our local approach, the tool uses measurements taken close to their source, which are less likely to being manipulated. These measurements are then used to validate the impact of commands coming from the "less trustworthy" control center, which is more commonly attacked by hackers (ICS-CERT (2016, 2010)).

We explain how the safety and security rules are implemented as policies in Bro, such that they can run in close proximity to the RTUs in the field stations. Furthermore, we discuss how these policies need to change with the current state of the system.

The contribution of this paper is threefold: (i) A parser for the SCADA communication protocol IEC-104 is presented, which is able to interpret packet content and allow it to be processed using the Bro monitoring tool. (ii) For a specific field station, we illustrate how two constraints are turned into Bro rules and how they depend on the system state. (iii) The proposed approach is evaluated on a real traffic trace recently captured at a substation of a Dutch electricity distribution operator.

The paper is further organized as follows. Section 2 presents the related work w.r.t. process-based intrusion detection. Section 3 discusses necessary background on SCADA systems and intrusion detection. Section 4 explains how the existing physical constraints and safety requirements are written in the form of rules and Section 5 explains how such rules are translated to Bro policies and explains how those polices are state-dependent. The proposed monitoring system is evaluated in Section 6 and the paper is concluded in Section 7.

2. RELATED WORK

SCADA security and intrusion detection is a very active research area. Many proposed intrusion detection mechanisms only analyze the periodicity of packets sent in the network (Udd et al. (2016)), or investigate the structure of packets, creating a model according to the specification (Yang et al. (2013)). However, they do not incorporate process information into the decision process.

Nivethan and Papa (2016a) propose to incorporate process semantics by mapping the monitoring requirements to the respective PLC registers. The proposed mechanism issues alerts when, e.g., process variables are not within the requested bounds. While this paper does not discuss where and how this detection method could be implemented, it mostly duplicates the Energy Management System (EMS) at the central control room. Moreover, the defined rules are static, i.e., they compare process values with a desired set point. Furthermore, many popular protocols do not simply transport floating point values between the field stations and the main control room, e.g., IEC-104 uses scaled and normalized values, while Modbus transports the floating points in the form of several pairs of 2-byte registers. This complicates the use of the above approach for different protocols. Also, a method to dynamically generate rules for fine-tuned deep packet inspection has been proposed by the same authors (Nivethan and Papa (2016b)), which however, does not discuss taking the process semantics into account.

Lin et al. (2013) propose to detect malicious commands in power grids by including a model of the entire power grid into a centrally-located detection mechanism and share that information with a distributed network of Intrusion Detection Systems. These implement a Bro IDS, which triggers run-time power flow analysis that uses information obtained from network packets, to pre-calculate the physical consequences of the sent command. This approach trusts the directly connected, locally obtained data from sensing devices and does not trust the "intelligent" (controlling) devices. In their earlier work, Lin et al. (2013) explain a way to include specification-based IDS for the DNP3 protocol, also implementing a Bro IDS.

Fovino et al. (2010) aim at detecting *complex SCADA attacks* by monitoring the evolution of the state of PLC registers and creating a database of critical states, which are unsafe, to the system. Complex attacks consist of series of commands, that are licit to a traditional IDS when considered separately, but causing harm when issued together. They present examples of packet signatures for the two implemented protocols: Modbus/TCP and DNP3. To maintain an up-to-date system state, the presented solution uses a Modbus client that polls all PLCs for the values of all their registers, which introduces an additional load in the network. The approach heavily relies on the availability of an up-to-date database of the blacklisted critical states. It

remains unclear whether this database is updated automatically, or manually. Moreover, critical states may change with the current state of the system, which is not addressed in the paper. While their approach is close to one proposed here, we make use of more abstract rules to detect unsafe states and to blacklist them.

Hadžiosmanović et al. (2014) proposed an anomaly detection approach that compares register values to predictions of process variables based on historical data. This approach does not predict the outcome of an incoming command, it rather detects whether process variables deviate from their normal values.

3. SCADA AND INTRUSION DETECTION

Section 3.1 provides a brief overview on SCADA systems and IDS for industrial control systems. Section 3.2 discusses how process variables are communicated within the protocol IEC-104.

3.1. Background on SCADA and IDS

SCADA systems are often used in critical infrastructures, like power distribution, to monitor and control the underlying physical system. In the field stations, sensors measure e.g., voltage and current on the buses and power lines, and pass them to a Remote Terminal Unit (RTU) or Programmable Logic Controller (PLC). Via a communication infrastructure RTUs and PLCs forward information to the Master Terminal Unit (MTU), located in the Control Room. There, the information is processed by additional SCADA servers, which, e.g., estimate the state of the physical system, and determine if changes are required in that system. In that case, commands are sent to actuators using the same communication channels. The information is then archived in a historian server and after it is processed, the the overview of the physical system state is updated on the Human Machine Interface (HMI).

SCADA systems have been designed to provide better functionality and availability for physical systems, often lacking mechanism to ensure integrity and security of the processed information. However, even newly designed systems, which implement security mechanisms have vulnerabilities. Recent events, such as the attack on the Ukrainian power grid (ICS-CERT (2016)), or reports of new malware tailored specifically to ICS, such as CRASHOVERRIDE[1] or TRISIS[2], show that hackers are aware of vulnerabilities and are able to abuse them.

[1] https://www.dragos.com/blog/crashoverride/index.html
[2] https://www.dragos.com/blog/trisis/index.html

Many security mechanisms can be implemented in SCADA systems to reduce the risk and the (impact) of a security breach. IDSes help to analyze and secure ongoing communication, e.g., by whitelisting known IP addresses (Barbosa et al. (2013)). In contrast, this paper focuses on process-aware IDS, which requires the ability to inspect the content of packets and relate them to the physical process at hand. Among the open-source network monitoring tools available for SCADA protocols, the most popular are Snort (Roesch (1999); Yang et al. (2013)), and Bro (Paxson (1999); Lin et al. (2013); Udd et al. (2016)). Snort allows for pattern matching within packets to, e.g., determine whether their format matches their specification, or alert when variables exceed certain thresholds. Instead, Bro has a modular structure, which can be extended to include the necessary parsers for different SCADA protocols. More importantly, Bro includes a framework for rule-based evaluation of the packet content, which enables the evaluation of the proposed process-aware rules.

3.2. Process variables in IEC-104

Many of the currently used SCADA protocols, such as Modbus, IEC-104, or DNP3 do not implement security mechanisms. However, some of them, such as IEC-104, allow to transfer analog values as scaled or normalized values instead of the actual floating point values (IEC104 (2013)).

Normalized values are numbers in the range <-1;1>, where the precision depends on the number of bits available. If the resolution of the measured value is coarser than the unit of the least significant bit, then the least significant bit is set to 0 (IEC101 (2003); Clarke et al. (2004)).

Scaled values transmit values with a defined fixed decimal point. Values span <-32768; 32767>, and the range and position of the decimal point are stored in the configuration of the RTU and central server (IEC101 (2003); Clarke et al. (2004)).

For example, when using normalized values, both the main server in the control room and the RTUs in the field stations store reference values of all process variables, e.g., PV_{ref}. The transmitted value PV_{trans} of a real value PV_{real} is then defined as $PV_{trans} = \frac{PV_{real}}{PV_{ref}}$. For example, if the reference value for the current on a line is equal to 1000A and the measured value is 150A, then 0.15 is sent as normalized value to the control room. This approach is used by operators to obscure the data passed along the communication channels. However, this complicates security mechanisms as proposed e.g. by Nivethan and Papa (2016a), while process-aware

policies, as proposed in this paper, facilitates the use of normalized values.

Overall, IEC-104 passes information using Application Protocol Data Units (APDUs) which consist of Application Protocol Control Information (APCI) and Application Service Data Units (ASDUs). APCIs contain the start byte (104), the length of the APDU, and 4 bytes of control information. The content of a header of an I-frame, which transports actual system information, could be:

```
<Start=104, ApduLen=23, Type=I (0x00),
 i_send_seq=13, recv_seq=100)>
```

ADSUs contain actual system information that needs to be transmitted and can consist of multiple objects of the same Type ID, for various Information Object Addresses (IOAs). Consider the following example for an ASDU of Type ID C_SE_TA_1 (Set Point Command Normalized Value with Time Tag):

```
<typeID=C_SE_TA_1,
data_unit_identifier=(seq=0, num_ix=1),
cause_of_transmission=
(cot=Activation, negative=0, test=0),
originator_address=3,
common_address=1500,
c_se_ta_1=
[<info_obj_addr=12333,
normalized_value=0.0517578,
qos=(ql=0, se=0),
cp56time2a=<milli=7758, minute=54, hour=9,
     day=2, mon=3, year=18> >]  >
```

The command above activates a request to set a threshold of the IOA with number 12333 to the value 0.0517578.

Process variables are also stored in the IEC-104 RTU configuration. Such a configuration contains sensor and actuator information, i.e., analog or digital input and output. Sensors provide information about the system state, while actuators enable the control of that system state. Separate addresses (IOAs) are used for all analog and digital inputs and outputs in the monitoring and in the control direction using a communication infrastructure.

4. FROM RULES TO POLICIES

Previously, a set of physical constraints and a set of safety requirements have been defined, which need to be satisfied by the system in order to ensure its safety (Chromik et al. (2016a)). Both these sets can be used to improve the system security, by maintaining a model of the system state and checking new measurements and commands against this model. Any violation of these rules then indicates a possible intrusion.

The system state changes over time, and as the rules depend on the state of the system, they have to be implemented into adaptive Bro policies in order to monitor and alert potentially malicious traffic. Turning these rules into process-aware policies within Bro requires knowledge on the topology of the system, as well as on the configuration of the RTU present in the field station.

This paper explains the idea of state-dependent policies and investigates how dynamic policies need to be, i.e., when and how they need to be updated within the IDS. In the future it might be possible to automatically generate such policies for given input topologies, configurations and protocols. A complete implementation of the approach, as sketched in Figure 2, is considered future work.

Physical constraints encompass physical laws, like Kirchoff's law. They are used to determine whether the system is consistent. If the measured values do not follow the laws of physics, they must be corrupted. They can also be used to pre-compute the next system state upon receiving a command. They are summarized in Table 1.

Safety requirements are conditions which need to hold to ensure the system operates in a safe and reliable way. They are based on low and medium voltage norms, e.g., that the low voltage needs to be within 10% bound from the nominal 240V (CENELEC (1988)), as well as on the knowledge of the operators, for this particular system. We list the requirements in Table 2. Upon every reading and every command, the pre-calculated system state has to satisfy all safety requirements. For example, if the pre-calculated system state shows a current on a power line that exceeds the maximum current allowed, the operator is notified, since the power line wears out faster.

RTU configuration contains the mapping of process variables to the addresses, where the values are stored. In case of IEC-104, the address of the process variables in the monitoring direction (used for data acquisition) is usually different than in the control direction (used for commands). In the example below, the current limit value for field 1 is contained in address 12333 in the control direction.

```
RtuAddr: 1500, Address: 120,
IOA_M: 9000, IOA_Mtt: 11222, IOA_C: 12333,
IoDescription: Field_1 current limit,
LowerBound: -1000, UpperBound: 1000,
DimensionText: A
```

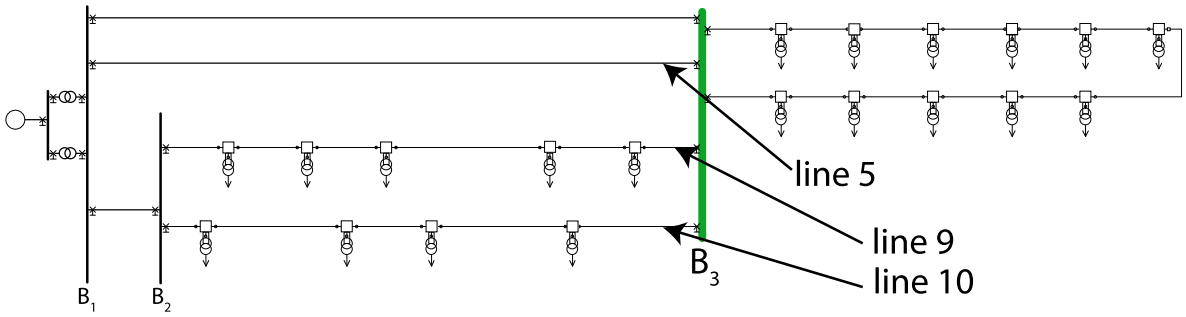

Figure 1: *The investigated topology.*

Table 1: *Physical consistency constraints*

Physical consistency constraint	Explanation
$\forall B_i \in \mathcal{B} \left(\sum_{L_j \in B_i.in} L_j.B_i.M.I = \sum_{L_k \in B_i.out} L_k.B_i.M.I \right)$	Kirchoff's current law
$\forall L_i \in \mathcal{L} \, (\forall S_j \in L_i.S \, ((S_j.st = 0) \Rightarrow (\forall M_k \in L_i.M \, ((M_k.I = 0) \wedge (M_k.V = 0)))))$	if all switches on a line are open, the values of current and voltage on this line have to be zero
$\forall T_i \in \mathcal{T}, M_x, M_y : M_x = L_x.T_i.M \wedge M_y = L_y.T_i.M \, (T_i.r = M_x.V/M_y.V = M_y.I/M_x.I)$ for $M_x.V > M_y.V$	assuming no losses, the transformer changes voltage and current according to a predefined ratio r
$P = V \cdot I$, e.g. $P_1^G.pv = P_1^G.pos.P_1^G.M.I \cdot P_1^G.pos.P_1^G.M.V$	electric power is equal to voltage times current

Table 2: *Safety requirements*

Safety requirement	Explanation
$\forall L_i \in \mathcal{L} \;\; \forall M_h \in L_i.M \;\; M_h.V \in [0.9L_i.V_{ref}; 1.1L_i.V_{ref}]$	voltage on all lines stays between the $\pm$10% boundaries of the reference voltage value
$\forall L_i \in \mathcal{L} \;\; \forall M \in L_i.M \;\; M_h.I \leq L_i.I_{max}$	current in a power line does not exceed its maximum allowed current
$\sum_{\mathcal{P}^G} P_i^G + \sum_{\mathcal{P}^C} P_j^C = 0$	power produced by the sources equals the power consumed by the loads
Interlocks (defined based on topology), e.g., $S_i.st \;\|\|\; S_j.st = 1$	some switches cannot be opened simultaneously for safety reasons
$\forall L_i \in \mathcal{L} \;\; L_i.I_{sp} \in [0.95L_i.I_{max}; 1.05L_i.I_{max}]$	the current set point of a power line should be set within the $\pm$5% boundaries of its maximum current value

Topology is the description of the system at hand. We use the format previously proposed (Chromik et al. (2016b)). This paper analyzes the topology shown in Figure 1, provided to us by the Dutch distribution system operator where we have also obtained the traffic capture. We build Bro policies for the RTU that is located at bus B_3 (marked in green). In Section 6, we investigate commands concerning line 5 and 9, also marked in this figure.

However, just the topological information about the elements listed above is not enough to define the policies for the monitoring tool. Based on the interactions between the listed elements, it is possible to develop three sets of rules: (i) *infrastructure*-based, (ii) *topology*-based, and (iii) *load*-based, that help to keep the system in a safe and secure state. We illustrate these types and their input and output dependencies in Figure 2, as explained below.

Rules type A: Infrastructure - these rules depend on the present infrastructure and change only when the physical system is changed, e.g., if a cable is replaced by one with a different maximum allowed current. Infrastructure changes are relatively rare, hence, we decided to not adjust these rules automatically within the Bro policies. Hence, a change in the infrastructure requires an explicit update by the operator, which we believe helps to keep the system secure.

Rules type B: Topology/Switching - these rules are to some extent dynamic and can be updated based on the system state. They change when the topology of the system is changed, e.g., due to a switching command. An example of such a rule is interlocking, i.e., a mutual dependency between the state of some switches. Upon receiving a command about the change of a switch state, the reference model of the switch states in Bro is updated, and the policies analyze the new, resulting model.

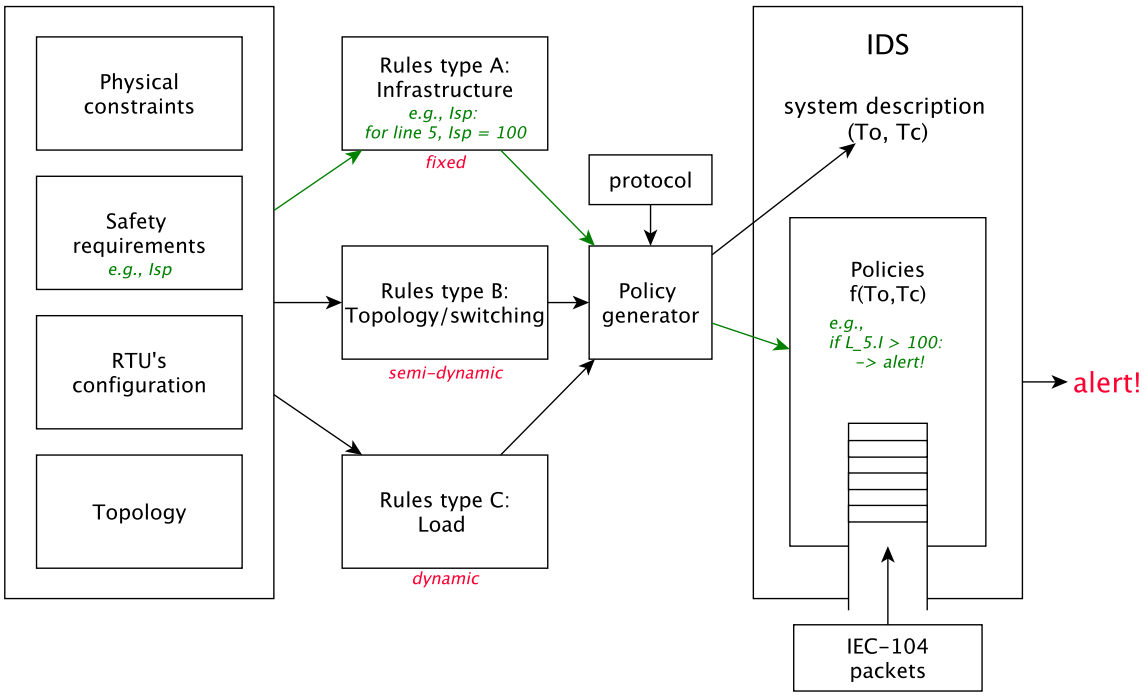

Figure 2: *Example of generating Bro policy for maximal current from the system description components.*

Rules type C: Load - these rules depend on the load and are highly dynamic. They are updated automatically in the system, e.g., a decision to open a power line depends on the load of all connected power lines. The state of the system depends on the load that is newly reported with every reading of sensor measurements, and is updated in Bro accordingly. The policies referring to the load depend on that reference state.

To illustrate our approach, we investigate the use of the following rules for the RTU located at bus B_3 (cf. Figure 1):

R1: ensures that only one of the lines L_9 and L_{10} can be open at the same time. Their states are determined by switches S_9 and S_{10}, respectively. This is a rule of Type B and ensures an interlock:

$$(S_9.st.close)\mathrm{OR}(S_{10}.st.close) == \mathrm{True}.$$

R2: The maximum allowed current in line L_5 is 160 A. This is a rule of Type A and relates to the set point defining the maximum allowed current on a line. This threshold may be set to any value within a 5% range of the maximum allowed current.

$$L_5.I_{\mathrm{sp}} == 160A \pm 5\%.$$

For example, the operator may want to change it to 165 A. However, significantly larger values, such as 200 A are suspicious.

Once, the rules for a specific RTU and a given (part of the) topology have been defined, they need to be translated into state-dependent Bro policies for a given protocol type (cf. Figure 2). Separating the definition of rules form Bro policies and the protocol ensures the extensibility of our approach, e.g., to other monitoring tools and protocols.

5. STATE-DEPENDENT POLICIES IN BRO

The developed parser is described in Section 5.1, the requirements for implementing state-dependent policies in Bro are discussed in Section 5.2 and 5.3. The implementation for the two previously discussed rules is provided in Section 5.4.

5.1. IEC-104 Parser

As the previously discussed rules might depend on the current state of the physical system, the model of the system needs to be updated and compared to the communicated system information. Hence, a parser is needed that reads the relevant information from the packets transmitted using a certain standard and directly feeds them into the monitoring tool. To analyze IEC-104 traffic in Bro, a dedicated parser for processing the packets and extracting the values representing process variables is required.

We developed a dedicated Bro parser for the IEC-104 protocol using the Spicy parser generator

(Sommer et al. (2016)) and part of the parser developed with BinPAC++ (Udd et al. (2016)). The development version of our parser is available on github[3]. We extended previous work of Udd et al. (2016) by adapting it to the Spicy parser generator, adding more Type IDs present in our capture, and by writing functions that enable content processing in Bro. Table 3 lists the implemented Type IDs.

Table 3: *Implemented Type IDs*

Type ID number	Type ID
1	M_SP_NA_1
2	M_SP_TA_1
3	M_DP_NA_1
5	M_ST_NA_1
9	M_ME_NA_1
11	M_ME_NB_1
13	M_ME_NC_1
30	M_SP_TB_1
34	M_ME_TD_1
36	M_ME_TF_1
45	C_SC_NA_1
50	C_SE_NC_1
58	C_SC_TA_1
61	C_SE_TA_1
70	M_EI_NA_1
100	C_IC_NA_1
101	C_CI_NA_1
103	C_CS_NA_1
107	C_TS_TA_1
142	proprietary
143	proprietary
200	proprietary

The Type IDs marked in green are especially interesting, because packets carrying these functions directly influence the physical system. C_SC_TA_1 is a single command function, that changes the state of a digital output on an RTU, and therefore, e.g., the state of a switch on a power line. C_SE_TA_1 is a set point command, changing, e.g., the maximum allowed current on the power line.

5.2. Construction of the monitoring tool

Based on the information about the configuration of the system under consideration, as discussed in Section 4, two main components are created for the monitoring tool: (i) a description of the system state (To), and (ii) a set of policies.

Figure 3 illustrates the dependencies of Bro policies on the current state of the system. Upon the arrival of a new packet, the monitoring tool first pre-computes (or extracts) the system state (step (i)), then checks the respective policies for the IOA in the packet on the resulting system state (step (ii)). For example, if the IOA refers to a switch state, the IDS will only check switching-related policies. If the policies detect a violation of the rules described in Section 4 for the pre-computed system state, an alert is issued to the operator. In step (iii), the pre-computed (or extracted) state is saved as the current system state.

5.3. State of the system

The state of the switches in a specific system is described as shown in Listing 1 and updated upon the arrival of a new command. The value of the maximum allowed current thresholds is shown in Listing 2; it should only change upon infrastructural changes. Changing these thresholds to a very low value would trigger many alerts, which could potentially result in operators ignoring them. However, a very high value could result in missing intrusions or dangerously high values of the current on the power line. Bro also stores the translation between the IOA address and the object name, e.g., $[4455] = \text{“}S_9.st.open''$ means that IOA 4455 refers to the object $\text{“}S_9.st.open''$.

Listing 1 Table with status of the switches.

```
global switches: table[string] of bool = {
["S_2.st.open"] = F,
["S_2.st.close"] = T,
["S_5.st.open"] = F,
["S_5.st.close"] = T,
["S_9.st.open"] = F,
["S_9.st.close"] = T,
["S_10.st.open"] = T,
["S_10.st.close"] = F,
["S_11.st.open"] = F,
["S_11.st.close"] = T,
["S_14.st.open"] = F,
["S_14.st.close"] = T };
```

Listing 2 Table with maximum allowed current on the power lines.

```
global max_current: table[string] of double = {
["L_2.I_sp"] = 0.08,
["L_5.I_sp"] = 0.08,
["L_9.I_sp"] = 0.08,
["L_10.I_sp"] = 0.08,
["L_11.I_sp"] = 0.08,
["L_14.I_sp"] = 0.08 };
```

5.4. Policies

The rules can directly be translated into policies, using the input language of the monitoring tool. This paper presents two policies, which directly translate the two rules R1 and R2 specified in Section 4. Listings 3 and 4 refer to those rules, respectively.

We implement the Bro policies to be triggered upon receiving packets with following Type IDs: C_SC_TA_1 and C_SE_TA_1 with the IEC-104 ASDU Cause Of Transmission field set to “Activation”. Once a packet containing either of those Type IDs is received by the monitoring tool, a state update is triggered, followed by a policy check. If a policy detects a violation, an alert is generated.

In the currently proposed solution, the monitoring tool only *informs* the operator about the violation.

[3] https://github.com/jjchromik/hilti-104

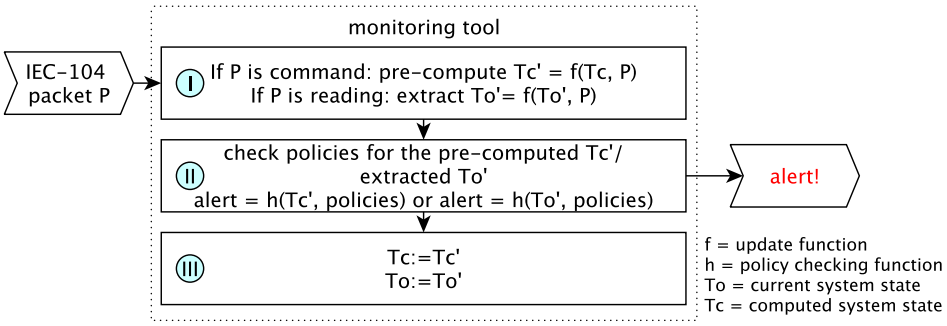

Figure 3: *Diagram representing the working of the monitoring tool.*

It is possible, however, to block the packets from reaching the RTU, therefore, preventing the command from being executed.

Listing 3 Bro policy for interlocks.

```
function check_interlock_policy(): bool{
  return (switches["S_10.st.close"] ||
    switches["S_9.st.close"]); }
```

Listing 4 Bro policy for maximum current.

```
function check_maxcurrent_policy(ioa: int): bool{
return ((max_current_precalc[name_lines[ioa]]
  >0.95*max_current[name_lines[ioa]])
  &&(max_current_precalc[name_lines[ioa]]
  <1.05*max_current[name_lines[ioa]]));
}
```

6. EVALUATION

We use a trace captured on March 2, 2018 in a Dutch electrical facility that implements the IEC-104 protocol for the communication between the field stations and the control room. Next to regular SCADA traffic, the capture contains packets resulting from the operator performing two procedures: (i) changing the state of one switch from connected to disconnected, and (ii) changing the set point (threshold) value of the maximum current on one power line, significantly.

Listing 5 shows our monitoring tool's output as provided to the operator. An alert is displayed (shown in red), when a violation of the interlock policy is detected. The tool informs which packet is being processed, therefore, the operator can directly see what caused the alert.

According to Listing 1, the initial state of all switches, except S_{10}, is closed, i.e., the power lines are connected. Due to the interlock constraint, cf. rule R1 in Section 4, line 10 and line 9 cannot be open simultaneously. Hence, the action of sending a command to open switch S_9 should trigger an alert, as can be seen in Listing 5. However, closing switch S_9 is allowed, which can be seen in Listing 5: where the command to set "$S_9.st.close$" to True results in the output "Switching allowed". These two commands (open switch S_9 and close switch S_9) were present twice in the captured file, and both times they provided the same (correct) output.

Listing 6 again shows the monitoring tool's output to the operator. As shown in Listing 2, the threshold for the normalized maximum allowed current is set to 0.08 for all power lines. Rule R2 from Section 4 mentions that the set point can be changed only within a 5% range of the original value. This is checked by the Bro policy shown in Listing 4. As can be seen in Listing 6, our tool alerts when a new set point command requests to change this value to 0.051758 or 0.030762. The set point 0.082031 is within the allowed bounds, hence, does not issue an alert.

The following observations can be made from the presented log outputs. Firstly, the proposed parser is able to extract the relevant values of the IOAs present in IEC-104 packets and feed them to the Bro monitoring tool for further analysis. Secondly, the output of the policies implemented in Bro reflect the current system state. Finally, the log output can be adjusted to be informative to the system operator, e.g., when the object name is used instead of an IOA number.

7. CONCLUSION AND FUTURE WORK

The security of field stations in power distribution requires adequate mechanisms that can operate in a distributed manner and that take the state of the physical system into account. For that purpose, this paper proposes an approach for intrusion detection in field stations, implementing state-dependent policies using the Bro monitoring tool.

We have shown the feasibility of our process-based monitoring approach using real IEC-104 traffic traces. Clearly, in order to apply the approach to other protocols, also dedicated parsers have to be developed.

Listing 5 Bro log output for interlocks.

```
2018-04-14_07:40:27 [DEBUG] New switch change command for address 4455 (S_9.st.open), value: T
2018-04-14_07:40:27 [DEBUG] Updating the system state
2018-04-14_07:40:27 [DEBUG] Alert! Switching violation!
2018-04-14_07:40:27 [DEBUG] New switch change command for address 4456 (S_9.st.close), value: T
2018-04-14_07:40:27 [DEBUG] Updating the system state
2018-04-14_07:40:27 [DEBUG] Switching allowed.
2018-04-14_07:40:27 [DEBUG] New switch change command for address 4455 (S_9.st.open), value: T
2018-04-14_07:40:27 [DEBUG] Updating the system state
2018-04-14_07:40:27 [DEBUG] Alert! Switching violation!
2018-04-14_07:40:27 [DEBUG] New switch change command for address 4456 (S_9.st.close), value: T
2018-04-14_07:40:27 [DEBUG] Updating the system state
2018-04-14_07:40:27 [DEBUG] Switching allowed.
```

Listing 6 Bro log output for set point commands.

```
2018-03-02_09:15:34 [DEBUG] New set point change command for address 12333 (line L_5.I_sp), value: 0.051758
2018-03-02_09:15:34 [DEBUG] Updating the system state
2018-03-02_09:15:34 [DEBUG] Alert! Set point deviates from the pre-defined one.
2018-03-02_09:15:34 [DEBUG] New set point change command for address 12333 (line L_5.I_sp), value: 0.082031
2018-03-02_09:15:34 [DEBUG] Updating the system state
2018-03-02_09:15:34 [DEBUG] Set point in norm.
2018-03-02_09:15:34 [DEBUG] New set point change command for address 12333 (line L_5.I_sp), value: 0.030762
2018-03-02_09:15:34 [DEBUG] Updating the system state
2018-03-02_09:15:34 [DEBUG] Alert! Set point deviates from the pre-defined one.
2018-03-02_09:15:34 [DEBUG] New set point change command for address 12333 (line L_5.I_sp), value: 0.082031
2018-03-02_09:15:34 [DEBUG] Updating the system state
2018-03-02_09:15:34 [DEBUG] Set point in norm.
```

While the presented approach is not meant as a replacement of security measures at the central control room, it has certain advantages, as it has direct access to the physical process. On the one hand, it could potentially have prevented an attack like the one that happened in Ukraine (ICS-CERT (2016)), as it would not simply execute the command sent from the corrupted control room. On the other hand, also mistakes made by the operators could be prevented with it. For example, the power outage in 2015 in the Schiphol area (Associated Press (2017)), which heavily affected the largest airport in the Netherlands, was a result of violating the interlocking policies, as discussed in this paper. Hence, process-based IDS is not only able to increase the security of the controlled process, but also its safety, as it also captures legitimate commands which jeopardize the safety of the process.

Future work will focus on the automated generation of rules for a given topology and RTU configuration, as well as their automated translation into Bro policies. With increasing system complexity, it is likely that the processing time per packet increases within the Bro monitoring tool. We plan to conduct a careful trade-off analysis between execution time and accuracy when including larger parts of the real topology into the system.

Note, that the values of IOAs and thresholds have been changed to ensure anonymity.

Acknowledgments This research is funded through the NWO project (MOre secure scada through SElf-awarenesS), grant nr. 628.001.012. We thank the anonymous Dutch DSO for providing us with the capture of the traffic from one of their field stations.

REFERENCES

Associated Press (viewed 26.06.2017). Flights cancelled at Schiphol airport as power outage hits Amsterdam. https://www.theguardian.com/world/2015/mar/27/flights-cancelled-schiphol-airport-power-outage-amsterdam.

Barbosa, R. R. R., R. Sadre, and A. Pras (2013). Flow whitelisting in SCADA networks. *International journal of critical infrastructure protection 6*(3-4), 150–158.

Cárdenas, A. A., S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry (2011). Attacks against process control systems: risk assessment, detection, and response. In *Proceedings of the 6th ACM symposium on information, computer and communications security*, pp. 355–366. ACM.

Caselli, M., E. Zambon, and F. Kargl (2015). Sequence-aware intrusion detection in industrial control systems. In *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*, pp. 13–24. ACM.

CENELEC (1988). Harmonisation Document: Nominal voltage for low voltage public electricity supply systems, HD 472 S1.

Chromik, J. J., A. Remke, and B. R. Haverkort (2016a). Improving SCADA security of a local process with a power grid model. In *Proceedings of the 4th International Symposium for ICS&SCADA Cyber Security Research, Queen's Belfast University, UK*, pp. 114–123. BCS Learning & Development Ltd. doi: 10.14236/ewic/ICS2016.13.

Chromik, J. J., A. Remke, and B. R. Haverkort (2016b). What's under the hood? Improving SCADA security with process awareness. In *Proceedings of the Joint Workshop on Cyber-Physical Security and Resilience in Smart Grids, Vienna, Austria*, pp. 1–6. IEEE. doi: 10.1109/CPSRSG.2016.7684100.

Clarke, G. R., D. Reynders, and E. Wright (2004). *Practical modern SCADA protocols: DNP3, 60870.5 and related systems*. Newnes.

Fovino, I. N., A. Carcano, T. D. L. Murel, A. Trombetta, and M. Masera (2010). Modbus/DNP3 state-based intrusion detection system. In *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pp. 729–736. IEEE.

Hadžiosmanović, D., D. Bolzoni, and P. H. Hartel (2012). A log mining approach for process monitoring in SCADA. *International Journal of Information Security 11*(4), 231–251.

Hadžiosmanović, D., R. Sommer, E. Zambon, and P. H. Hartel (2014). Through the eye of the PLC: semantic security monitoring for industrial processes. In *Proceedings of the 30th Annual Computer Security Applications Conference*, pp. 126–135. ACM.

ICS-CERT (released February 25, 2016). Alert (IR-ALERT-H-16-056-01) Cyber-Attack Against Ukrainian Critical Infrastructure. Available online: `https://ics-cert.us-cert.gov/alerts/IR-ALERT-H-16-056-01`. Accessed on: 10 Mar 2018.

ICS-CERT (released September 29, 2010). Advisory (ICSA-10-272-01), Primary Stuxnet Advisory. Available online: `https://ics-cert.us-cert.gov/advisories/ICSA-10-272-01`. Accessed on: 10 Nov 2017.

IEC101 (2003). IEC TS 60870-5-101:2003. Technical specification, TC 57 - Power systems management and associated information exchange, Geneva.

IEC104 (2013). IEC TS 60870-5-7:2013. Technical specification, TC 57 - Power systems management and associated information exchange, Geneva.

Lin, H., A. Slagell, C. Di Martino, Z. Kalbarczyk, and R. K. Iyer (2013). 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. 5. ACM.

Lin, H., A. Slagell, Z. Kalbarczyk, P. W. Sauer, and R. K. Iyer (2013). Semantic security analysis of SCADA networks to detect malicious control commands in power grids. In *Proceedings of the first ACM workshop on Smart energy grid security*, pp. 29–34. ACM.

Nivethan, J. and M. Papa (2016a). A SCADA Intrusion Detection Framework that Incorporates Process Semantics. In *Proceedings of the 11th Annual Cyber and Information Security Research Conference*, pp. 6. ACM.

Nivethan, J. and M. Papa (2016b). Dynamic rule generation for SCADA intrusion detection. In *Technologies for Homeland Security (HST), 2016 IEEE Symposium on*, pp. 1–5. IEEE.

Paxson, V. (1999). Bro: a system for detecting network intruders in real-time. *Computer networks 31*(23), 2435–2463. doi: 10.1016/S1389-1286(99)00112-7.

Roesch, M. (1999). Snort - Lightweight Intrusion Detection for Networks. In *Proceedings of the 13th USENIX Conference on System Administration, Seattle, WA, USA*, LISA '99, pp. 229–238. USENIX Association. doi: 10.1.1.105.6212.

Sommer, R., J. Amann, and S. Hall (2016). Spicy: a unified deep packet inspection framework for safely dissecting all your data. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*, pp. 558–569. ACM.

Udd, R., M. Asplund, S. Nadjm-Tehrani, M. Kazemtabrizi, and M. Ekstedt (2016). Exploiting bro for intrusion detection in a SCADA system. In *Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security*, pp. 44–51. ACM.

Yang, Y., K. McLaughlin, T. Littler, S. Sezer, B. Pranggono, and H. Wang (2013). Intrusion detection system for IEC 60870-5-104 based SCADA networks. In *Power and Energy Society General Meeting (PES), 2013 IEEE*, pp. 1–5. IEEE.