Python Plug-in Data Logging Analysis for Intrusion Detection System

Shiva Jyothi Angala and Jayasree Boya

***Abstract*—**In this paper, we discussed about the Supervisory Control and Data Acquisition (SCADA). SCADA systems are a core part of industrial systems, such as smart grid power and water distribution systems. In recent years, such systems become highly vulnerable to cyber-attack. We are using Gas Pipeline dataset for data log analysis for detecting the attacks to scale it into intrusion detection system by using `a python plugin. There aren't many IDS systems for detecting attacks on the virtual gas systems or any other similar systems Our contribution is to protect those systems from several attack types. For Example: Setpoint Attack, Pump Attack, System Mode Attack, Bad CRC Attack, etc.,

Keywords—SCADA, Intrusion Detection System, Cyber Security

I. INTRODUCTION

Supervisory control and data acquisition (SCADA) is a control system architecture that uses computers, networked data communications and graphical user interfaces for high-level process supervisory management, but uses other peripheral devices such as programmable logic controllers and discrete PID controllers to interface to the process plant or machinery. The operator interfaces which enable monitoring and the issuing of process commands, such as controller set point changes, are handled through the SCADA supervisory computer system. However, the real-time control logic or controller calculations are performed by networked modules which connect to the field sensors and actuators.
The SCADA concept was developed as a universal means of remote access to a variety of local control modules, which could be from different manufacturers allowing access through standard automation protocols. In practice, large SCADA systems have grown to become very similar to distributed control systems in function, but using multiple means of interfacing with the plant. They can control large-scale processes that can include multiple sites, and work over large distances. It is one of the most commonly-used types of industrial control systems, however there are concerns about SCADA systems being vulnerable to cyberwarfare/cyberterrorism attacks.

Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices. An intrusion detection system (IDS) is software that automates the intrusion detection process. Network-Based IDS (NIDS) monitors network traffic for network segments or devices and analyzes the network and application protocol activity to identify suspicious activity.
Security Log Analysis Systems are also known as Log-based Intrusion Detection Systems (LIDS). Log Analysis for Intrusion Detection [1]is the process or techniques used to detect attacks on a specific environment using logs as the primary source of information. LIDS is also used to detect computer misuse, policy violations and other forms of inappropriate activities.

The main argument of this paper is that NIDS and LIDS are necessary for effectively monitoring the security posture of an organization. Both techniques, network-based detection and log-based detection, complement each other in the identification and reporting of security incidents.

II. PROBLEM STATEMENT

In supervisory control and data acquisition (SCADA) systems, there are many possible signs of incidents which may go unnoticed. These events can be studied mainly by analyzing network behavior or by reviewing computer security event logs. In order to avoid or minimize the losses from an incident outcome, the events need to be analyzed as close to real-time as possible. Logging and intrusion detection systems have the potential to produce very large amount of data, and all that data must be managed, filtered and analyzed. Having a single approach and a unified platform helps with this very difficult and challenging task to monitor and report in near-real time.

Automation is needed to perform an initial analysis of the data and to alert on select events of interest for human review. Event correlation software and centralized logging can be of great value in automating the analysis process. However, the effectiveness of the process depends on the quality of the data and the data rules that goes into it.

III. PROPOSED ARCHITECTURE

Supervisory control and data acquisition (SCADA) systems should establish logging standards and procedures to ensure that adequate information is collected by logs and security software and that the data is reviewed regularly. The length of time to maintain log data is dependent on several factors, including the organization's data retention policies and the volume of data. As each business has different needs and regulatory requirements, legal counsel should be obtained to determine the appropriate retention schedule for logs.

Log analysis is an art and is geared towards narrowing down to the events of interest. Analyst needs to focus on recent changes, failures, errors, status changes, access and administration events, and other events unusual for your environment. Hence, it is important to minimize noise by removing routine, repetitive log entries from the view after confirming that they are benign. Analyst needs to correlate activities across different logs to get a comprehensive picture of the situation

IV. DESIGN AND IMPLEMENTATION

Typical legacy SCADA uses Modbus RTU open protocol.

Modbus RTU: Modbus RTU is an open, serial (RS-232 or RS-485) protocol derived from the Master/Slave architecture. It is a widely-accepted protocol due to its ease of use and reliability. Modbus RTU is widely used within Building Management Systems (BMS) and Industrial Automation Systems (IAS). This wide acceptance is due in large part to MODBUS RTU's ease of use. MODBUS RTU messages are a simple 16-bit CRC (Cyclic-Redundant Checksum). The simplicity of these messages is to ensure reliability. Due to this simplicity, the basic 16-bit MODBUS RTU register structure can be used to pack in floating point, tables, ASCII text, queues, and other unrelated data.

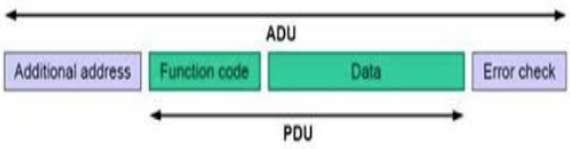

Figure 1: MODBUS Protocol

The MODBUS protocol defines a simple Protocol Data Unit (PDU) independent of the underlying communication layers. The mapping of MODBUS protocol on specific buses or networks can introduce some additional fields on the Application Data Unit (ADU).

The client that initiates a MODBUS transaction builds the MODBUS Application Data Unit. The function code indicates to the server which kind of action to perform.

Table 1: MODBUS Function Codes

	Function Code
1	Poll Controller
2	Read Discrete Input
3	Read Holding Registers
4	Read Input Registers
5	Write Single Coil
6	Write Single Holding Registers
7	Read Exception Status
8	Diagnostic
9	Program 484
10	Poll 484
11	Get Com Event Counter
12	Get Com Event Log
13	Program Controller
14	Poll Controller
15	Write Multiple Coils
16	Wr Multiple Holding Registers
17	Report Slace ID
43	Read Device Identification
128	Duplicate Station

Decode Records: MODBUS Frame contains several set of fields. As the first step these data fields should organize in to a object.

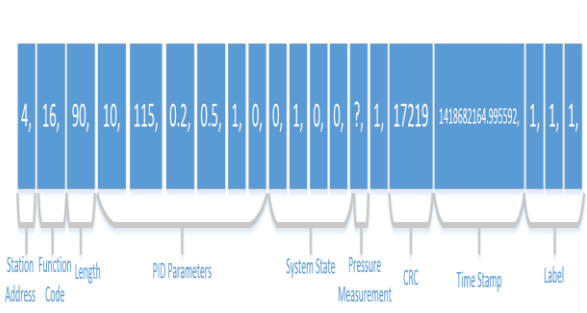

Figure 2: MODBUS Frame

Table 2: Features of ARFF Dataset

Attribute	Description
address	The station address of the MODBUS slave device. This address is the same on a query and response to a given slave device.
function	MODBUS function code.
length	The length of the MODBUS packet.
setpoint	The pressure set point when the system is in the Automatic system mode.
gain	PID gain.
reset rate	PID reset rate.
deadband	PID dead band.
cycle time	PID cycle time.
rate	PID rate.
system mode	The system's mode automatic (2), manual (1), or off (0).
control scheme	The control scheme is either pump (0) or solenoid (1). This determines which mechanism is used to regulate the set point.
pump	Pump control; on (1) or off (0). Only used in manual mode.
solenoid	Relief valve control; opened (1) or closed (0). Only used in manual mode.
pressure measurement	Pressure measurement.
crc rate	
command response	Command (1) or response (0).
time	Time stamp.
binary result	Binary class; attack (1) or normal (0).
Attack category	Category of attack (0-7).
specific result	Specific attack (0-35)

Approach:

1. Split the fields from the data and structure it into an object
2. Check for the "specific result" filed of the object.
3. If the value of "specific result" is between (1-35) identify that as an attack and process the object and get the Attack Name, Address and other information's.
4. Collect the number of attacks and its type

Identify attacks: By doing a static analysis on the structured object, it can identify the frames which contain the attacks.

```
## check for attacking vector
def check_attack(spc_rslt):
    if spc_rslt > 0 and spc_rslt < 36:
        return 1
    else:
        return 0
```

In the above code snippet is used to detect the attack.

Collect Information: After the identification of attacking frames, relevant information is interpreted into the user and number of attacks are count based on the attack and its type.

Table 3: Cyber Attacks

Attack Name	Num-bers	Categor-y	
Setpoint Attack	1-2	MPCI	Changes the pressure set point outside and inside of the range of normal operation.
PID Gain Attack	3-4	MPCI	Changes the gain outside and inside of the range of normal operation.
PID Reset Rate Attack	5-6	MPCI	Changes the reset rate outside and inside of the range of normal operation.
PID Rate Attack	7-8	MPCI	Changes the rate outside and inside of the range of normal operation.
PID Deadband Attack	9-10	MPCI	Changes the dead band outside and inside of the range of normal operation.
PID Deadband Attack	9-10	MPCI	Changes the dead band outside and inside of the range of normal operation.
PID Cycle Time Attack	11-12	MPCI	Changes the cycle time outside and inside of the range of normal operation.
Pump Attack	13	MSCI	Randomly changes the state of the pump.
Solenoid Attack	14	MSCI	Randomly changes the state of the solenoid.
System Mode Attack	15	MSCI	Randomly changes the system mode.
Critical Condition Attack	16-17	MSCI	Places the system in a Critical Condition. This condition is not included in normal activity.
Bad CRC Attack	18	DOS	Sends MODBUS packets with incorrect CRC values. This can cause denial of service.
Clean Registers Attack	19	MFCI	Cleans registers in the slave device.
Device Scan Attack	20	Recon	Scan for all possible devices controlled by the master.
Force Listen Attack	21	MFCI	Forces the slave to only listen.
Restart Attack	22	MFCI	Restart communication on the device.
Function Code Scan Attack	24	Recon	Scans for possible functions that are being used on the system. The data about the device is not recorded, but is performed as if it were being recorded.
Rise/Fall Attack	25-26	CMRI	Sends back pressure readings which create trends on the pressure reading's graph.
Slope Attack	27-28	CMRI	Randomly increases/decreases pressure reading by a random slope.
Random Value Attack	29-31	NMRI	Random pressure measurements are sent to the master.
Negative Pressure Attack	32	NMRI	Sends back a negative pressure reading from the slave.
Fast Attacks	33-34	CMRI	Sends back a high set point then a low setpoint which changes fast
Slow Attack	35	CMRI	Sends back a high setpoint then a low setpoint which changes slow

NMRI: Naive Malicious Response Injection
CMRI: Complex Malicious Response Injection
MSCI: Malicious State Command Injection
MPCI: Malicious Parameter Command Injection
MFCI: Malicious Function Code Command injection

V. Conclusion & Limitations

We worked on data logs that includes 35 cyber-attacks against SCADA control systems which uses the MODBUS communication protocol. The attacks are tabulated into 5 categories; NMRI, CMRI, MSCI, MPCI, MFCI. Structuring data into the objects because some fields doesn't have valid data. Unable to adapt into changing environment because this is created for virtual gas pipeline system. Scale the project into intrusion detection system

VI. References

[1]http://www.ece.uah.edu/~thm0009/icsdatasets/cyberhuntsvillepaper_v4.pdf.

[2]https://ai2s2pdfs.s3.amazonaws.com/5ce7/c8ee818815e96cd16fcb801dba2be5e39b1a.pdf

[3] https://docs.python.org/3/distributing/index.html

[4]https://pdfs.semanticscholar.org/ac78/cbc6b1902f3eca1c97011b12330dc58951dd.pdf

[5] Morris, T., Gao, W., Industrial Control System Network Traffic Data sets to Facilitate Intrusion Detection System Research, Critical Infrastructure Protection VIII, Sujeet Shenoi and Johnathan Butts, Eds. IFIP Advances in Information and Communication Technology, Springer Berlin Heidelberg, Volume 441, 2014, pp 65-78.

[6] https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets