I. J. Computer Network and Information Security, **2020, 6, 44-56**
Published Online December 2020 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2020.06.04

Industrial Control Systems Honeypot: A Formal Analysis of Conpot

Sheetal Gokhale
K.J. Somaiya College of Engineering, Mumbai, India
E-mail: sheetal.gokhale@somaiya.edu

Ashwini Dalvi
Veermata Jijabai Technological Institute, Mumbai, India
*corresponding author E-mail: ashwinidalvi@gmail.com

Irfan Siddavatam
K.J. Somaiya College of Engineering, Mumbai, India
E-mail: irfansiddavtam@somaiya.edu

Received: 02 August 2020; Accepted: 13 September 2020; Published: 08 December 2020

Abstract: Technologies used in ICS and Smart Grid are overlapping. The most discussed attacks on ICSs are Stuxnet and Black energy malware. The anatomy of these attacks not only pointed out that the security of ICS is of prime concern but also demanded to execute a proactive approach in practicing ICS security. Honeypot is used to implement defensive measures for security. The Honeynet group released Honeypot for ICS labelled as Conpot in 2013. Though the Conpot is low interactive Honeypot, it emulates processes of different cyber-physical systems, typically Smart Grid. In the literature, the effectiveness of Honeypot operations was studied by challenging limitations of the existing setup or proposing new variants. Similar approaches are followed for Conpot evaluation. However, none of the work addressed a formal verification method to verify the engagement of Honeypot, and this makes the presented work unique. For proposed work, Coloured Petri Net (CPN) tool is used for formal verification of Conpot. The variants of Conpot are modelled, including initial state model, deadlock state model and livelock model. Further evaluation of these models based on state space analysis results confirmed that Conpot could lure an attacker by engaging him in an infinite loop and thereby limiting the scope of the attacker from exploring and damaging the real-time systems or services. However, in the deadlock state, the attacker's activity in the conpot will be restricted and will be unable to proceed further as the conpot model incorporates deadlock loop.

Index Terms: Industrial Control System ICS, Honeypot, Conpot, Coloured Petri Net, Formal method analysis, Cyber security.

1. Introduction

Industrial Control Systems (ICSs) are a critical part of the Smart Grid. One of the types of ICS is Supervisory Control and Data Acquisition (SCADA) system. The literature covered the possibility as well as the anatomy of a range of cyber-attacks on ICS. It established the fact that ICS of Smart Grid is vulnerable to different types of cyber-attacks. Therefore, such cyber-attacks lead to the requirement of proactive security measures for ICSs.

Honeypot is one of the approaches to achieve defensive security practice. Honeypots are designed to lure attackers with the impression of the real-time system and log the attack trail. To set up Honeypot, it is essential to know what part of the system to be defense could interest attackers. Honeypot could classify depending upon its purpose, usage at the server-side or client-side, and engagement of an attacker with Honeypots. The Honeypot is used either in the Production stage or for Research purpose. Low interactive and high interactive are types of engagement of users with Honeypot.

The usefulness and effectiveness of Honeypot in IT infrastructure inspired researchers to adopt this concept in ICSs. In a further discussion in paper ICS and SCADA, terms are used interchangeably. The Honeynet group released Honeypot for SCADA labelled as Conpot in 2013. Conpot is one of the SCADA Honeypots to support various use cases of Smart Grid. It categorized under low interactive Honeypot. The attractive characteristic of Conpot is the ease of implementation along with simulated PLCs (Programmable Logic Controllers). Conpot offers support for protocols like MODBUS, HTTP, and SNMP and like other IT Honeypots, it can maintain a log of attacks. Conpot keeps the log of the attacker's attempt on setup through accessing HTTP and SNMP request and MODBUS communication. The variations of Conpot are discussed in the literature to improve the experience of a real system with Honeypot set up.

Copyright © 2020 MECS

Conpot is low interactive Honeypot with limitations such as lack of inclusion of a physical process and Telnet protocol which is used by PLCs frequently. Still, Conpot is one of the maintained projects on ICS Honeypot. It is required to analyse Conpot to grasp a fundamental understanding of its operation and to improve performance with emerging variants. Honeypot lures the attacker by simulating the real-time system. So it would be better to comprehend Honeypot in the form of transition of states to engage the attacker. In other words, modelling Honeypot is suitable to estimate likeable states for the attacker or confirming states which could prove deadlock to the attacker.

In the past few years, modelling of the system is performed to establish structure-wise behaviour of the system and further extended for verification of the system operation. In this regards the formal method based approach is adopted for verification of the system. Formal methods notations include process algebra, Petri Nets, Temporal Logic and could be analysed with tools such as Coloured Petri Nets (CPN), the Prototype Verification System (PVS), and Symbolic Analysis Laboratory (SAL) and likewise. The Petri Net is adaptable for verification of system behaviour with an upper bound on the system transaction. The Coloured Petri Nets (CPN) is the tool which specifies different tokens (i.e. representation of process/object of the system) with colour.

The research objective is to verify the behavioural analysis of conpot CPN model's when introduced with the deadlock and livelock loops which are then confirmed or evaluated using state space tool. The aim is proposed based on the knowledge that there might be some scenarios where an attacker might take over the honeypot machine or identify honeypot machine because of its shortcomings. Hence to extend the engagement capability of a honeypot, the behavioural analysis of conpot introducing deadlock and livelock is carried out. The objectives categorized in two steps are as follows:

- The first objective is to model the Conpot in CPN and then introduce the deadlock and livelock loop to study the behavioural analysis of conpot under deadlock and livelock state.
- The second objective is to evaluate all those three models which are initial conpot model, deadlock model and livelock model using the state-space tool to verify the correctness of the model and to determine the presence of deadlock and livelock respectively.

The novel contributions of the work of modelling the Conpot - Honeypot for ICS with CPN are as follows:

- Model to represent a deadlock state and infinite firing of a sequence of the token depicting in the finite loop in Honeypot systems.
- Model for engaging an attacker in an infinite cycle to stop him/her from exploring the real-time systems but at the same time make the attacker believe that his attacks are successful.
- To assess the behavioural properties of Honeypot models using a state space tool of CPN.

The rest of the paper is classified as follows: Section II covers discussion on Honeypots in ICS and an overview of the usage of Coloured Petri Net based Formal verification. Section III discusses the CPN Methodology used for the formal method work. Section IV discusses the modelling of Conpot with CPN tools. State space analysis of the Conpot CPN model is discussed in detail in section V, Results and Discussion is discussed in section VI, and Section VII concludes the paper.

2. Related works

2.1. Honeypots for Industrial Control Systems

With much discussed Stuxnet attack, the security of ICS drew the attention of researchers specifically. The smart grid is one of the cyber physical systems, where ICS plays a vital role. So many researchers attempted to gauge how Honeypot could be an expedient proactive measure to minimize or eradicate possibilities of cyber-attacks like Stuxnet. The work [1] presented the survey of Honeypots and Honeynets for Smart grid. Honeyd, first of TCP service emulating Honeypot has been modified as a SCADA honeypot project. The further variant of Honeyd is HoneydV6 to support IPv6. Further, with an introduction of Conpot, the Honeypot is designed to simulate ICS devices. Arthur Jicha et al. [2] did a detailed analysis of Conpot to evaluate its effectiveness. Literature also addressed the limitations of Conpot like Conpot is vulnerable to expose its identity by leaving other ports open during set up.

The work [3] had the preliminary assumption that existing honeypots for cyber physical systems failed to emulate the physical processes and physical nature of devices effectively. This failure alerts intruder about the presence of Honeypot. Thus authors proposed Honeypot titled HoneyPhy, which was compatible with the physics of devices of the different cyber physical system. In work [4], Honeypot-To-Go, referred to as HosTaGe mobile honeypot, was proposed for ICS to support associated protocol and emulate physical processes. The proposed low interaction honeypot claimed to be outperformed Conpot with regards to ICS protocol implementation. Different variants of Honeypots are suggested and discussed in the given work [5,6,7,8,9]. In paper [10], the authors proposed an algorithm for dynamic Honeypot to increase the engagement time of an intruder. Such dynamism was made possible by monitoring the host continuously,

having a virtual host with the re-configurable ability and monitoring of the network. Paper claimed advantages like minimum interaction of the operator, increase security awareness and an independent view of hosts and services on the network. The present approach of formal method could be the facilitator for work proposed in this paper. Referring to the CPN model, the states of Honeypot could be an estimate, and accordingly, virtual host and network service can be configured. In [11], Honeypot log data is further extended with visualisation tool support for criminal profiling on the SCADA network.

Honeypot is further extended by the concept of Honeynet where more than one honeypots are deployed to act as a decoy; one such Honeynet is SCyPH framework [12]. In [13], the authors proposed Honeynet for smart grid consisting of numerous honeypots simulated as different substations. The work claimed to be unique in the domain of smart grid honeypots. The attack vectors considered were attack from the control center and network as well as attack via VPN. Though the work proposed Honeynet, for proof of concept single substation honeypot was discussed.

Though different variants of Honeypots and Honeynets are discussed, the frequent interactions and requirements need to model to confirm verification of the behaviour of Honeypot. For verification, formal analysis is proposed in the present work. Next section discusses objective and use cases of formal method analysis using Coloured Petri Nets.

2.2. Coloured Petri Net (CPN) Based Formal Verification

The basis of the Formal method approach is strict clinging to mathematical notations and proofs. It is useful in validating dynamic state changing processes of a system. The research has used Formal methods for different domains. The formal methods include Process algebra, Petri Nets, temporal Logic, likewise. Petri Nets approach involves 'State' to represent the process, object and 'transition' to signify changes. It has the ability to confirm characteristics of the modelled system as well to verify the correctness of the system [14]. It is best suited with a concurrent system which is asynchronous. Petri Nets suffered from complexity to present large system. To minimize the complexity of representation of a large system, coloured Petri Nets based approach is used [15]. CPN uses concepts such as data type, data structure from a high level programming language.

The Coloured Petri Nets based approach is explored recently for verifying different domain use cases ranging from the Internet of Things (IoT) to critical infrastructure like a nuclear power plant. In [16], CPN based modelling was done for GIS-oriented IoT services. Security analysis of Modbus protocol with CPN based formal analysis performed in [17]. The CPN model was developed to model protocol behaviour under reasonable condition and attack condition. In [18], the MQTT protocol of IoT data communication was verified with the CPN model. The CPN model was simulated to verify the behavioural properties of the MQTT protocol. The effective sensor placement and information management in the smart factory to be simulated was the aim of work [19]. The proposed work claimed the CPN model as middleware for data sensing. The [20] proposed "Piping Possibilistic Timed Coloured Petri Net (PPTCPN)" to model the uncertainty of the instrument and control system of the nuclear plant. For discussion of the concept, the authors modelled the Digital Feedwater Control system.

The versatility of the CPN approach ranges from modelling protocol behaviour to the modelling control system of a smart grid, motivated objective of proposed work. Propose work aims to model Conpot Honeypot with CPN approach to verify its behaviour. Section IV discusses the modelling and verification of Conpot with CPN.

3. Methodology

The methodology used for formal method analysis of conpot is a CPN tool [21] which is used to model the Petri net of conpot honeypot and verify the correctness of it using state-space tool incorporated in CPN. CPN method is a combination of coloured Petri nets and Meta Language programming which is used for modelling and verification of the concurrent system. Petri nets are also called as CP-nets or Place/Transition net as it depicts the place transition graph of the system. From the related works, it is clear that CPN is best suited to model a concurrent system. It is also concluded that Petri nets suffer from complexity to present large system, and therefore the coloured Petri nets approach is used for the work.

Petri nets allow users to create states or nodes or places and declare data types for those states used in the model using Meta Language programming. CPN enables the user to create a model with places, transitions and directed arcs. Places or states means the nodes in the modelling of the system and transitions signifies the state changes or specific task and directed arcs connects places to transition or transition to places. Places are denoted by oval or circle, transitions by square or rectangle. In CPN, data types such as integer, string, float, etc., are defined and declared by using the term a color set or colset. After declaring color set or data types, variables can be declared to particular color set. Declaring color set or data types to the nodes allows us to assign values known as tokens to the places. This tokens assigned to states or places enables transitions and transitions are only enabled, or tokens are only fired if its input place contains tokens. The enabled transitions appear green.

Once the CPN model is built, the next step is to perform simulation of the model, i.e. by using a single step or multi-step debugging which let us observe the state changes and its effect on each place or state. Then lastly evaluate and verify the correctness of the model using the state-space tool for verification of the behavioural analysis of the model and to determine its working under certain conditions.

Copyright © 2020 MECS

The application of CPN methodology is presented in section IV, which explains the modelling of the conpot models, and section V discusses the verification of the models based on the evaluation framework.

4. Honeypot Model in CPN

The Coloured Petri Net [21] tool version 4.0.1 is used to model SCADA Honeypot models. The coloured Petri net tool enables modelling of complex and concurrent systems, and the coloured nets are the extended Petri nets that allow identifying tokens by "colors." Thus the coloured Petri nets are used to model the SCADA honeypot model such as conpot.

Conpot modelling in CPN represents the working of conpot model to comprehend the behaviour of it when introduced with the deadlock state as well as the livelock state. The terms of deadlock state and the infinite loop will be described in three sections, namely the Conpot Honeypot Model, Conpot Honeypot Model with Deadlock and Conpot Honeypot Model with Infinite loop.

4.1. Conpot Honeypot Model in CPN Tools

The Conpot Honeypot is designed as a formal description in CPN tools. The SCADA systems consisting of PLCs (Programmable Logic Controllers) and HMI (human-machine interface) are emulated in Conpot as part of the simulated network. PLC runs five services such as FTP, Telnet, HTTP, SNMP and Modbus [22]. The most attack attempts are carried out on HTTP, Modbus protocols by adversaries. Hence considering the mostly attacked protocols of the PLCs, the HTTP and Modbus protocols are simulated in coloured petri nets (CPN) tool. The three main Honeypot or Honeynet components, data control, data capture and data collector, are included to capture the attack attempts and keystrokes used by the intruders. The three honeypot components play the role as follows:

- Data control module logs the incoming connections (i.e. IP address and SessionID), manages the Conpot file system (Conpot VFS), and Keylogger captures keystrokes.
- Data capture module logs the command keystrokes performed by the attacker.
- Data collector module stores the keystrokes of the particular attack type performed by the intruder in the SQLite database used for Conpot.

The Simulated PLCs that are emulating the HTTP and Modbus services store their respective data in the database. HTTP protocol stores the "request type" that is the request method used by an attacker. Modbus stores the "function code" and "slave id" for the service performed as "FnC" and "SlaveID".

In CPN, the term colorset or colset is used to declare data types. Fig.1 shows the Meta Language (ML) programming supported by the CPN Tools. So here in Fig.1, string data type or color set is assigned to the place "AttackType" which enables us to give string type tokens to the place "AttackType". Here tokens are "Nmap", "DoS" and "Get" (Fig.2).

The data types assigned to the state, which is termed as colorset in ML is shown in Fig.1 as "colset AttackType = string". The string color set is defined for the place "AttackType". The other color sets of string type are Keylogger, RecordIP, SessionID, and so on. The data type product is declared to the color set "Trap" which is a combination of "Attacker" and "Attacktype" in such a way to manage the flow of tokens or data. Refer Fig.1, "colset Trap = product Attacker*AttackType". The variable "a" of the color set "Attacker" is assigned a token "A", mentioned as guard inscription [a = "A"] at "AnyAttack" transition (Fig.2). The product colorset is also declared to "connlog" for logging the intruder's connection. It is the product of color set Attacker, AttackType, RecordIP, and SessionID based on their respective variables assigned such as "a", "s", "IP" and "SID". Refer Fig.1, "colset connlog = product Attacker*Attacktype*RecordIP*SessionID". Similarly, both HTTPReq and Modbus color set is the product of Attacker, AttackType, PLCs and Command color sets.

Fig.2 depicts the Conpot Honeypot process outlined as the coloured Petri net blueprint for the SCADA system's formal analysis. The two end processes are created after the "Emulated Network" place and after the "Trap" place before redirecting the traffic to simulated PLCs and HMI. It is designed based on the knowledge that an attacker might exit from the system even before performing any activity or entering into the system or may terminate the window after entering the system.

The ML (Meta Language) [23,24] programming is added to the Honeypot model (Fig.1), and values known as tokens are assigned at the transition "keystroke" as "ks" and "CMD" to store command keystrokes. The "accept and store" transition is assigned tokens for the IP address and session id as "IP" and "SID" of the specific attack types launched by the attacker. From Fig.2, it is concluded that the SQLite database stores the tokens passed from Simulated PLCs and HMI, respectively. The central databases stores HTTP requests, Modbus function code and slave id, as well as attacker's actions in conpot file system such as keystrokes of commands, including session or connection log such as IP address and session id of the respective attacker and attack type.

ML language supports to mark place inscriptions, transition inscriptions, guard inscriptions and arc inscriptions. Guard inscriptions depicted in the model at the transition "keystroke" and "accept and store" as [c = "CMD", k = "ks"]

and [ip = “IP”, sid = “SID”] respectively. Arc inscriptions are mentioned at the directed arcs like (a,s), (a,s,ip,sid) and so on. For HTTP and Modbus logs the tokens assigned as guard inscriptions to the transition “Protocol” are “ReqType”, “FnC”, “SlaveID” and “CMD”. For example, [h = “ReqType”, m = “FnC”, m1 = “SlaveID”, c = “CMD”] in Fig. 2. These variables h, m, m1, and c are declared to the “PLCs” color set, as shown in Fig.1.

Fig.1. Declaration for CPN Model

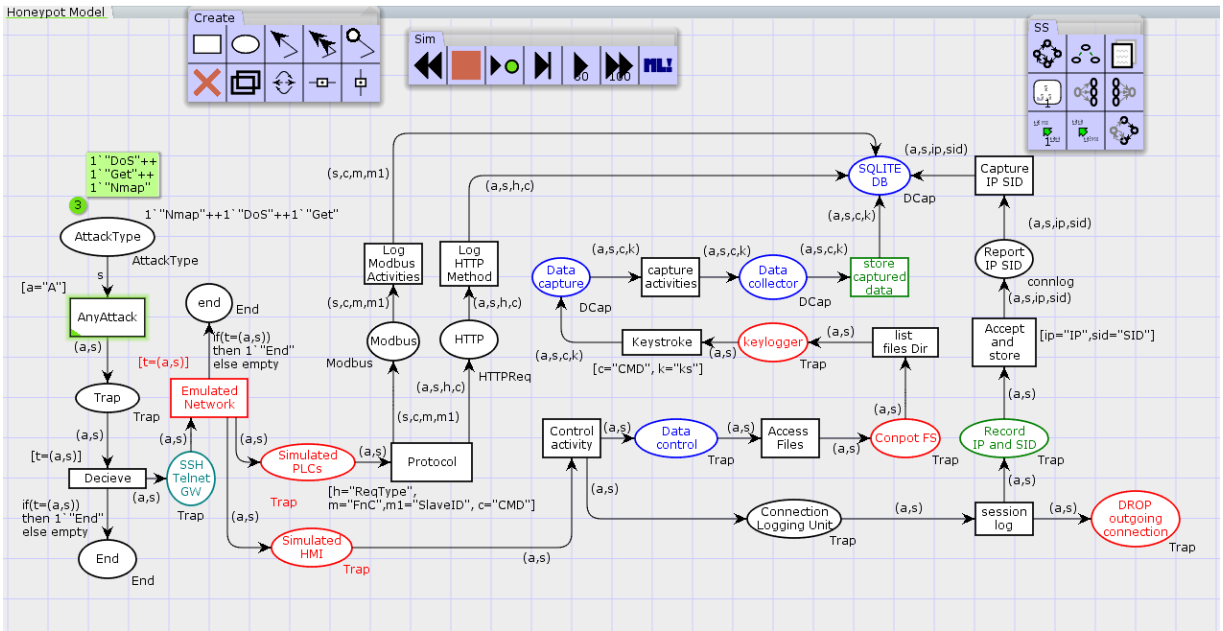


Fig.2. Conpot Honeypot Initial Model

4.2. Conpot Honeypot Model with Deadlock in CPN

The Conpot Honeypot Initial Model shown in Fig.2 is introduced with deadlock (refer Fig.4) to determine the working or behaviour of it under deadlock condition. The cyclic deadlock concept used in Fig.4 presents the deadlock process to trap an attacker if it enters the Honeypot system and performs some malicious activity. Suppose the attacker enters into the Conpot file system (Conpot FS) to gain information. In that case, the action or process of accessing and processing files or directories gets stuck in a deadlock state, thereby not able to navigate to the further processes or system or services. In other words, the attacker’s malicious activity does not progress, and the process gets stuck at the

"fetch data" place. This cause the activity of the attacker to be caught in a wait state for an infinite time, whereas at the same time, the keystrokes, IP address, session id, services for which the intruder requested for are recorded in the SQLite database. It is concluded from Fig.4 that the transition "deceptive data" is not enabled even after its input place "fetch data" contains tokens. Hence deadlock is detected, and intruder's malicious activity is in a deadlock state. Fig.3 represents the deadlock loop created with two states and one transition and bidirectional arcs directing back and forth between state and transitions are used in the Conpot Honeypot Model with Deadlock.

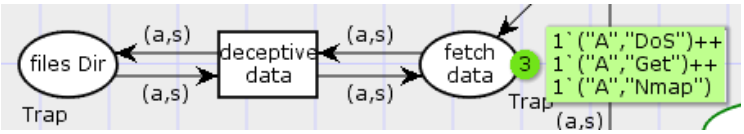


Fig.3. Cyclic process creating Deadlock in Conpot Honeypot Model

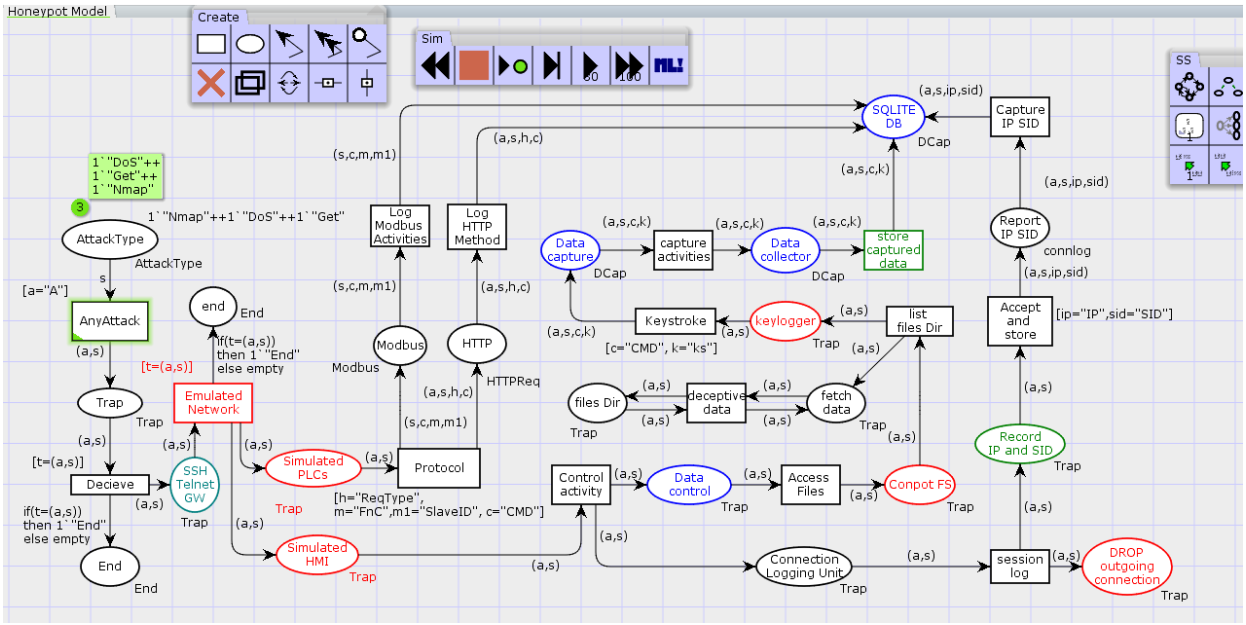


Fig.4. Formulation of Conpot Honeypot Model with Deadlock

A. *Deadlock and Livelock in CPN*

Petri net is said to be free from the deadlock if at least one transition is enabled at every reachable marking. Deadlock described in [25] states that under a deadlock state, no one statement is executed. In the livelock state, the transitions are executed repetitively, or tokens are fired infinitely (in a loop or a cycle), and they cannot quit (Fig.5). A transition of a Petri net is said to be dead if it is not enabled by any reachable state as seen from Fig.3 where transition "deceptive data" is not enabled.

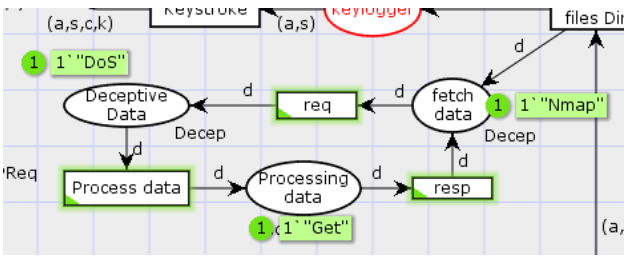


Fig.5. Cyclic process creating Infinite Firing Sequence (livelock) in Conpot Honeypot Model

4.3. *Conpot Honeypot Model with Infinite Loop in CPN Tools*

In this section, the Conpot Honeypot Initial Model shown in Fig.2 is introduced with Livelock loop and is discussed. This part deals with the behavioural analysis of Conpot under livelock loop or infinite firing sequence of tokens. The concept presented in Fig.6 presents the infinite firing sequence of tokens that engages an attacker for a long time if it enters the honeypot system (Fig.5).

Suppose the attacker enters into the Conpot file system (Conpot FS) to gain something of his interests or tries to

access files or directories, the attacker's action of processing files or attack process gets repeated consecutively, thereby engaging an attacker in an infinite loop. It prevents the attacker from navigating to further processes or system or services, i.e., states when engaged in the loop. The expected behaviour is that the attacker's malicious activity of retrieving information by using "ls" to list files and directories and to use the "rm" command to delete or remove files and directory from UNIX like operating system activity gets processed repetitively or chained in the loop. The activity keeps on repeating from "fetch data" place to "Deceptive data" and "Processing data". This repetitive loop causes the transitions "req", "resp" and "process data" to be detected as impartial transition instances where transition "t" is enabled for an infinite time which is reachable from any reachable state and home marking. It causes the activity of the attacker to be looped in a cycle that never ends and engages the attacker in an infinite loop for an endless time.

The keystrokes, IP address, session id, request type, function code and slave id are recorded as expected. The malicious tokens or actions performed by intruder keeps firing infinitely from "fetch data" place to "Deceptive data" and "Processing data" through the transitions incorporated in the cycle causing an infinite loop running in the Conpot Honeypot system. Hence this endless cycle stops an attacker from exploring the real-time systems. It leaves the attacker with the thought that the malicious actions are taking place, but it is caught in the livelock process. Note from Fig.5 that the transition "req", "resp" and "process data" is impartial because it occurs infinitely often in every IFS (infinite firing sequence) in fairness properties described by coloured Petri net tools.

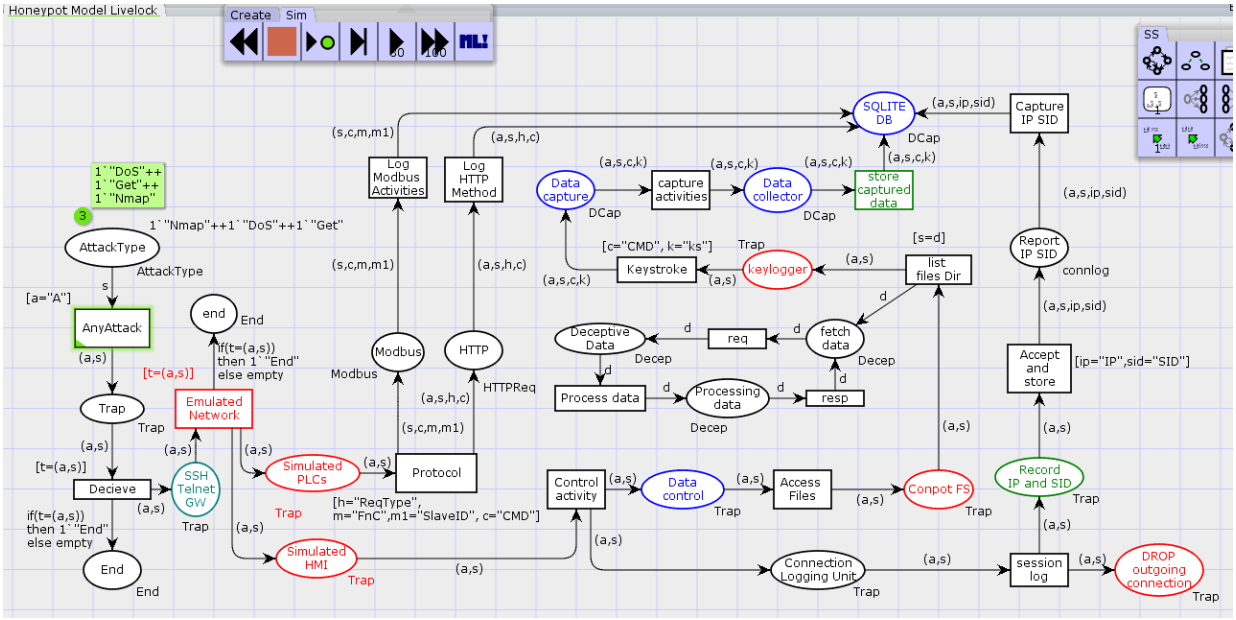


Fig.6. Formulation of Conpot Honeypot Model with Livelock

5. State Space Analysis Result

This section presents the state space report that assesses the behavioural properties of Honeypot models using a state space tool that is integrated with CPN. The states space tool calculates the state space of the model and saves the report. It calculates the reachable states (nodes) and state changes (arcs). It also states behavioural properties such as the size of the state space, home marking, dead marking, dead and live transition instances, etc. The CPN model shown in Fig.2 represents the initial model of Conpot SCADA emulation that includes the emulation of SCADA networks, PLCs and HMI which is used for users to interact with these devices and services of the industrial processes taking place. The PLCs shows the emulation of services such as HTTP (hypertext transfer protocol) and Modbus protocols. The HTTP protocol is a preferred transmission protocol that accepts requests and responds to those requests, i.e. it acts as request response protocol between a client and a server. The Modbus communication protocol is used to transfer the communication between electronic devices. The protocols activities are stored at the SQLite database, and HMIs activities at the client side are navigated through data control, data capture and data collection modules. These data handling modules describe the flow of the Honeypot as per the industrial perspective of working of SCADA systems (Fig.2). It controls the Conpot file system activities and records the interactions made to them. The Keylogger captures the commands keystrokes, be it manual or automated attacks, and a data capture module captures it. The Data collection module collects the data and passes it to the SQLite database. The data control module also consists of connection login unit node that records the session and IP (internet protocol) address of the incoming source in the database.

5.1. Evaluation

After the system operations are modelled and analysed, the next objective is to evaluate the formal model and behavioural properties of the Conpot CPN model to verify the correctness of the Petri net modelled system. To evaluate behavioural properties of the system, (i.e., boundedness properties, home properties, liveness properties to verify dead transitions and live transitions, and fairness properties) the dynamic state-space analysis is performed to obtain the required significant information about the CPN model. The evaluation process is the one-time execution process of the CPN models of initial, deadlock and livelock models. The approach for the model evaluation to observe behavioural properties is shown in Fig.7 as the evaluation framework.

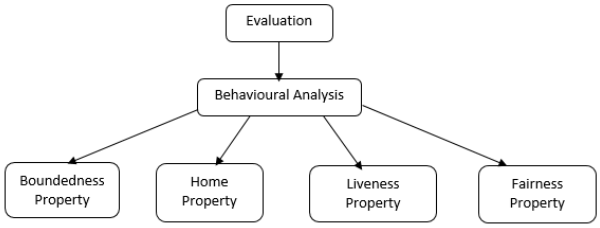


Fig.7. Evaluation Frameworks of the CPN Models

A. Behavioural Properties

Boundedness: The state-space reports (Fig.8, 9, 10) generated for all the CPN models (Fig.2, 4, 6) outlines the boundedness properties consisting of upper and lower bound places. It indicates that the "n" number of tokens placed and present in a place buffer. The proposed model of deadlock and livelock is bounded (safe) determining the Conpot model represents a finite list of operations from which the elements are taken one by one. It also indicates the reasoning enhances decision making without conflicts under uncertain circumstances.

Home marking: A home marking M_0 is said to be a home marking of CPN model if M_0 is said to belong to all the new markings (M'), i.e. reachable markings resulted from firing a transition t at home M.

In other words, in any circumstance, during the process model execution, it is always possible to reach the marking (i.e., reachable marking) in the conpot petri net system or CPN model where the transmission of all the data (tokens) is done successfully and makes a suitable decision. There is no home marking (Fig.2, 4) present in home properties of the state space report, which indicates that the CPN model terminates somewhere and does not enter the infinite loop. The reachable markings are shown in Fig.12 in the form of the reachability graph.

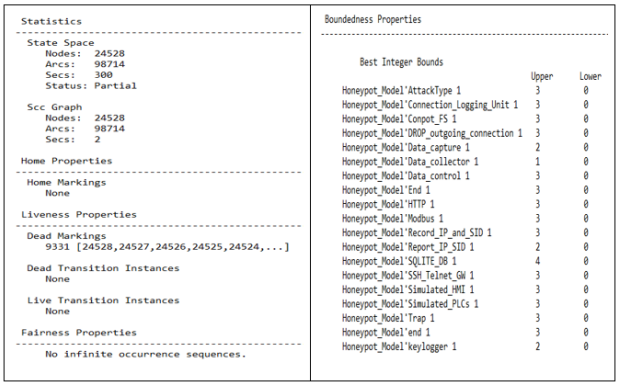
Dead marking: A dead marking M_{dead} is said to be the dead marking if no transition is enabled in the CPN model or system. For the CPN Model of the Conpot Honeypot Initial Model: Dead Marking is M_{9331}. Similarly for Conpot Honeypot with Deadlock: Dead marking is M_{7839}. And for the CPN model of the Conpot Honeypot with Livelock: Dead marking is M_{10260}.

Dead transition instance: The state-space analysis generated report shows (Fig.9) that there exists only a single dead transition (T_{dead}) = deceptive_data, which determines that the transition is not enabled. It confirms that the "deceptive_data" transition is under the unsatisfied condition in the decision-making process. The report determines that the CPN model does not represent any live transition instances present meaning where no transition is enabled infinitely (i.e., the continuous firing of tokens) and the state space calculation of the model is stopped at the dead marking (M_{7839}). Hence, there are no present any infinite occurrence sequences.

Fairness: In this phase, the fairness property of the Conpot Honeypot CPN model with livelock is measured. Impartial transitions present in the fairness property determines the strongest fairness property, and it indicates the presence of infinite firing sequences and in each infinite firing sequence (IFS) the transitions (process data, req, resp) occurs infinitely often. Also, the transitions present under the "transition with no fairness property" indicates that transitions are not just (Fig.10) and that there exists an IFS where transition "t" is continuously enabled from some point onward and does not fire anymore.

5.2. State Space Analysis of Conpot Honeypot Model

The state-space analysis report of "Conpot Honeypot Initial Model" (Fig.2) is presented in Fig.8. The State-space and SCC (Strongly Connected Components) graph nodes and arcs assist in identifying the reachable states and loops involved in the CPN model. The state-space report generates a partial status for a particular system or model configuration because it might be too big or takes longer than usual to calculate state spaces. Also, note that state-space and SCC graph nodes and arcs are the same and indicate that every node in the system is reachable from at least one or other place and the model has no loops incorporated. Note that home marking is none which means the model terminates somewhere and does not enter the infinite loop [26].



```
Statistics
------------------------------------------------------------------------

  State Space
     Nodes:  24528
     Arcs:   98714
     Secs:   300
     Status: Partial

  Scc Graph
     Nodes:  24528
     Arcs:   98714
     Secs:   2

Home Properties
------------------------------------------------------------------------

  Home Markings
     None

Liveness Properties
------------------------------------------------------------------------

  Dead Markings
     9331 [24528,24527,24526,24525,24524,...]

  Dead Transition Instances
     None

  Live Transition Instances
     None

Fairness Properties
------------------------------------------------------------------------
     No infinite occurrence sequences.
```

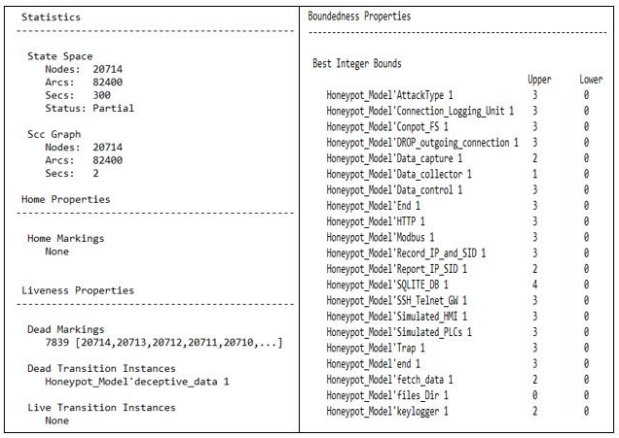
```
Boundedness Properties
------------------------------------------------------------------------

     Best Integer Bounds
                                                    Upper      Lower
     Honeypot_Model'AttackType 1                    3          0
     Honeypot_Model'Connection_Logging_Unit 1       3          0
     Honeypot_Model'Conpot_FS 1                     3          0
     Honeypot_Model'DROP_outgoing_connection 1      3          0
     Honeypot_Model'Data_capture 1                  2          0
     Honeypot_Model'Data_collector 1                1          0
     Honeypot_Model'Data_control 1                  3          0
     Honeypot_Model'End 1                           3          0
     Honeypot_Model'HTTP 1                          3          0
     Honeypot_Model'Modbus 1                        3          0
     Honeypot_Model'Record_IP_and_SID 1             3          0
     Honeypot_Model'Report_IP_SID 1                 2          0
     Honeypot_Model'SQLITE_DB 1                     4          0
     Honeypot_Model'SSH_Telnet_GW 1                 3          0
     Honeypot_Model'Simulated_HMI 1                 3          0
     Honeypot_Model'Simulated_PLCs 1                3          0
     Honeypot_Model'Trap 1                          3          0
     Honeypot_Model'end 1                           3          0
     Honeypot_Model'keylogger 1                     2          0
```

Fig.8. State-space report of Conpot Honeypot Initial Model

5.3. State Space Analysis of Conpot Honeypot Model with Deadlock

Similarly, the state space report of "Conpot Honeypot Model with Deadlock" (Fig.9) shows the same number of SCC Graph and State Space nodes and arcs. It indicates that every node in the system is reachable from at least one or another place, and the model has no loops incorporated. Note from Fig.9 that there exists a single dead transition instance as "deceptive_data" indicating the deadlock is present in the Honeypot system designed in CPN (Fig.4).

The decrease in state space arcs and nodes from the previous model of Conpot Honeypot (Fig.2, 8) indicates that there exists a deadlock and some of the nodes and transitions are not enabled and are not firing tokens.



```
Statistics
------------------------------------------------------------------------

  State Space
     Nodes:  20714
     Arcs:   82400
     Secs:   300
     Status: Partial

  Scc Graph
     Nodes:  20714
     Arcs:   82400
     Secs:   2

Home Properties
------------------------------------------------------------------------

  Home Markings
     None

Liveness Properties
------------------------------------------------------------------------

  Dead Markings
     7839 [20714,20713,20712,20711,20710,...]

  Dead Transition Instances
     Honeypot_Model'deceptive_data 1

  Live Transition Instances
     None
```

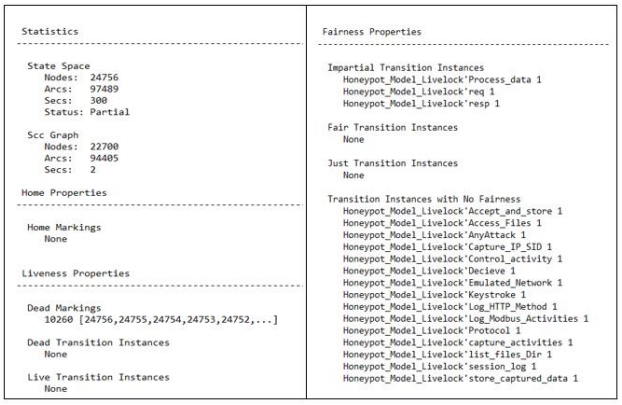
```
Boundedness Properties
------------------------------------------------------------------------

Best Integer Bounds
                                                    Upper      Lower
     Honeypot_Model'AttackType 1                    3          0
     Honeypot_Model'Connection_Logging_Unit 1       3          0
     Honeypot_Model'Conpot_FS 1                     3          0
     Honeypot_Model'DROP_outgoing_connection 1      3          0
     Honeypot_Model'Data_capture 1                  2          0
     Honeypot_Model'Data_collector 1                1          0
     Honeypot_Model'Data_control 1                  3          0
     Honeypot_Model'End 1                           3          0
     Honeypot_Model'HTTP 1                          3          0
     Honeypot_Model'Modbus 1                        3          0
     Honeypot_Model'Record_IP_and_SID 1             3          0
     Honeypot_Model'Report_IP_SID 1                 2          0
     Honeypot_Model'SQLITE_DB 1                     4          0
     Honeypot_Model'SSH_Telnet_GW 1                 3          0
     Honeypot_Model'Simulated_HMI 1                 3          0
     Honeypot_Model'Simulated_PLCs 1                3          0
     Honeypot_Model'Trap 1                          3          0
     Honeypot_Model'end 1                           3          0
     Honeypot_Model'fetch_data 1                    2          0
     Honeypot_Model'files_Dir 1                     0          0
     Honeypot_Model'keylogger 1                     2          0
```

Fig.9. State-space report of Conpot Honeypot Model with Deadlock

5.4. State Space Analysis of Conpot Honeypot Model with Livelock

If the state space and SCC graph arcs and nodes are the same, then each SCC comprises precisely one node, and there are no cycles present [27]. But note from Fig.10 that the graphs and nodes of SCC graph and state space are not same, which indicates a loop or cycle is present in the Conpot Honeypot CPN model with Livelock (Fig.6). Also, the Fairness properties show impartial transition instances such as "Process data", "req" and "resp" which are infinitely enabled when any reachable marking or home marking approaches them (Fig.10). Thus this impartial transition instances state that an infinite firing sequence (IFS) termed in CPN under fairness property occurs infinitely and loop incorporated in "honeypot CPN model with livelock" continue to execute.

Copyright © 2020 MECS



```
Statistics
----------

  State Space
     Nodes:  24756
     Arcs:   97489
     Secs:   300
     Status: Partial

  Scc Graph
     Nodes:  22700
     Arcs:   94405
     Secs:   2

Home Properties
---------------

  Home Markings
     None


Liveness Properties
-------------------

  Dead Markings
     10260 [24756,24755,24754,24753,24752,...]

  Dead Transition Instances
     None

  Live Transition Instances
     None
```

```
Fairness Properties
-------------------

  Impartial Transition Instances
     Honeypot_Model_Livelock'Process_data 1
     Honeypot_Model_Livelock'req 1
     Honeypot_Model_Livelock'resp 1

  Fair Transition Instances
     None

  Just Transition Instances
     None

  Transition Instances with No Fairness
     Honeypot_Model_Livelock'Accept_and_store 1
     Honeypot_Model_Livelock'Access_Files 1
     Honeypot_Model_Livelock'AnyAttack 1
     Honeypot_Model_Livelock'Capture_IP_SID 1
     Honeypot_Model_Livelock'Control_activity 1
     Honeypot_Model_Livelock'Decieve 1
     Honeypot_Model_Livelock'Emulated_Network 1
     Honeypot_Model_Livelock'Keystroke 1
     Honeypot_Model_Livelock'Log_HTTP_Method 1
     Honeypot_Model_Livelock'Log_Modbus_Activities 1
     Honeypot_Model_Livelock'Protocol 1
     Honeypot_Model_Livelock'capture_activities 1
     Honeypot_Model_Livelock'list_files_Dir 1
     Honeypot_Model_Livelock'session_log 1
     Honeypot_Model_Livelock'store_captured_data 1
```

Fig.10. State-space report of Conpot Honeypot Model with Livelock

```
Boundedness Properties
----------------------

  Best Integer Bounds
                                                    Upper      Lower
     Honeypot_Model_Livelock'AttackType 1               3          0
     Honeypot_Model_Livelock'Connection_Logging_Unit 1  3          0
     Honeypot_Model_Livelock'Conpot_FS 1                3          0
     Honeypot_Model_Livelock'DROP_outgoing_connection 1 3          0
     Honeypot_Model_Livelock'Data_capture 1             2          0
     Honeypot_Model_Livelock'Data_collector 1           1          0
     Honeypot_Model_Livelock'Data_control 1             3          0
     Honeypot_Model_Livelock'Deceptive_Data 1           2          0
     Honeypot_Model_Livelock'End 1                      3          0
     Honeypot_Model_Livelock'HTTP 1                     3          0
     Honeypot_Model_Livelock'Modbus 1                   3          0
     Honeypot_Model_Livelock'Processing_data 1          1          0
     Honeypot_Model_Livelock'Record_IP_and_SID 1        3          0
     Honeypot_Model_Livelock'Report_IP_SID 1            2          0
     Honeypot_Model_Livelock'SQLITE_DB 1                4          0
     Honeypot_Model_Livelock'SSH_Telnet_GW 1            3          0
     Honeypot_Model_Livelock'Simulated_HMI 1            3          0
     Honeypot_Model_Livelock'Simulated_PLCs 1           3          0
     Honeypot_Model_Livelock'Trap 1                     3          0
     Honeypot_Model_Livelock'end 1                      3          0
     Honeypot_Model_Livelock'fetch_data 1               2          0
     Honeypot_Model_Livelock'keylogger 1                2          0
```

Fig.11. Boundedness Property of Conpot Honeypot Model with Livelock

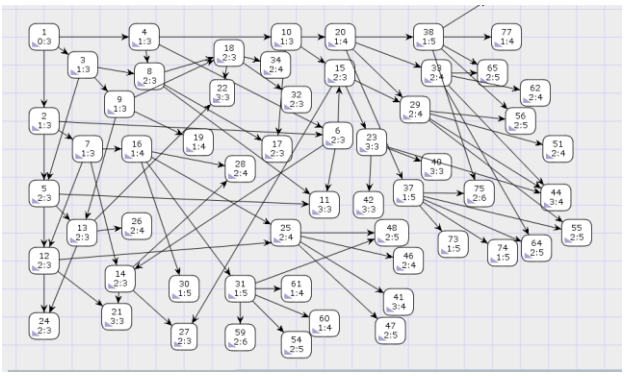


Fig.12. Reachability graph of Honeypot Model (Partial status)

To summarize the conpot initial model is introduced with deadlock and livelock condition and evaluation of those models using state-space tool determines the presence of deadlock and livelock (infinite firing of tokens) as seen in Fig.9 and 10. No work has been attempted on the topic of behavioural analysis of conpot using CPN tool by introducing deadlock and livelock conditions. Hence, this contributes to the security goal of engaging an attacker and preventing real-time systems from being explored.

6. Results and Discussion

The state-space report presents the state space size calculated for each honeypot model. The state-space report of “Conpot Honeypot Model”, and “Conpot Honeypot Model with Deadlock” show precisely the same state space and SCC graph nodes and arcs of their respective models. The “Conpot Honeypot Model with Livelock” state space

statistics show the difference in the state space, and SCC graph arcs and nodes that indicate the loops are incorporated in the system. The presence of impartial transition instances indicates that the transitions occur infinitely at every IFS (Fig.10). The state-space report of "Conpot Honeypot Model with Deadlock" realized the deadlock from the dead transition instance present in the system (Fig.9). Also, note that when home marking is none, it indicates that the model is terminating somewhere, i.e. it has end-node where all the activities of infinite cycle get stored in the database (SQLite database).

Honeypot itself is a trap for an intruder in which using deception mechanism the deadlock and IFS (livelock) process are depicted. The proposed work is a blueprint CPN model for Honeypot that states Honeypot with deadlock and livelock functionality engages the attacker in the false, deceptive process. The objective to achieve the behavioural analysis of conpot under deadlock and livelock condition the CPN tool is used first to simulate the working of conpot. Secondly, the conpot model is then incorporated with deadlock loop to analyze the behaviour or working of it and later the behavioural analysis of conpot under livelock condition is also tested. As seen in Fig.4 and 9, it is clear that the transition "deceptive data" is not enabled even if the tokens are present at its input place "fetch data" which means deadlock occurs. The activity of an attacker is collected as well as caught in deadlock process. The deceptive trap created is with the idea of two nodes and one transition forming a cycle in the process with bidirectional arcs (Fig.3).

Similarly, the trap is formed in "Conpot honeypot model with Livelock" forming a loop to represent a livelock. The livelock loop is created, including a cycle of states and transitions that act as input and output to its respective states and transitions (Fig.5).

7. Conclusions

The present work modelled Conpot using Coloured Petri Nets tools to verify its behaviour. Conpot is maintained and frequently used project for ICS Honeypot with characteristics of emulating numerous Smart Grid use-cases. The Conpot CPN models of three status (Initial, Deadlock and Livelock) are created and examined. Such formal analysis of Conpot offers better comprehension of estimating "Live" and "Dead" states of interaction. Conpot is low interactive Honeypot, so it is useful to know which interactions will drive Conpot in an infinite loop or a dead state. From the proposed model design, it is conclusive that Conpot suffers deadlock state if an attacker enters in a certain state and thus Conpot fails to generate specific attack trails.

Similarly, Conpot has the possibility of entering an infinite loop to engage the attacker in a honeypot. The proposed work suggests that the inclusion of deadlock or livelock condition in the conpot honeypot will act as the trap or engagement for the attacker and thus will prevent an attacker from exploring the real-time active systems and services. The current conpot honeypot has its limitation, and they might be identified or exploited if the attackers take over the honeypot machine. Hence to overcome certain shortcomings of the conpot, the deadlock model and livelock model is proposed for conpot and examined the behaviour analysis for understanding the working of it under deadlock or livelock conditions.

References

[1] Dalamagkas, C., Sarigiannidis, P., Ioannidis, D., Iturbe, E., Nikolis, O., Ramos, F., & Tzovaras, D. (2019, June). A survey on honeypots, honeynets and their applications on smart grid. In 2019 IEEE Conference on Network Softwarization (NetSoft) (pp. 93-100). IEEE.

[2] Jicha, A., Patton, M., & Chen, H. (2016, September). SCADA honeypots: An in-depth analysis of Conpot. In 2016 IEEE conference on intelligence and security informatics (ISI) (pp. 196-198). IEEE.

[3] Litchfield, S., Formby, D., Rogers, J., Meliopoulos, S., & Beyah, R. (2016). Poster: Re-thinking the honeypot for cyber-physical systems. In Poster at IEEE Symposium on Security and Privacy.

[4] Vasilomanolakis, E., Srinivasa, S., & Mühlhäuser, M. (2015, September). Did you really hack a nuclear power plant? An industrial control mobile honeypot. In 2015 IEEE Conference on Communications and Network Security (CNS) (pp. 729-730). IEEE.

[5] Cruz, T., Rosa, L., Proença, J., Maglaras, L., Aubigny, M., Lev, L., & Simoes, P. (2016). A cybersecurity detection framework for supervisory control and data acquisition systems. IEEE Transactions on Industrial Informatics, 12(6), 2236-2246.

[6] Serbanescu, A. V., Obermeier, S., & Yu, D. Y. (2015, July). A scalable honeynet architecture for industrial control systems. In International Conference on E-business and Telecommunications (pp. 179-200). Springer, Cham.

[7] Simões, P., Cruz, T., Proença, J., & Monteiro, E. (2015). Specialized honeypots for SCADA systems. In Cyber Security: Analytics, Technology and Automation (pp. 251-269). Springer, Cham.

[8] Pliatsios, D., Sarigiannidis, P., Liatifis, T., Rompolos, K., & Siniosoglou, I. (2019, September). A Novel and Interactive Industrial Control System Honeypot for Critical Smart Grid Infrastructure. In 2019 IEEE 24th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD) (pp. 1-6). IEEE.

[9] Belqruch, A., & Maach, A. (2019, March). SCADA security using SSH honeypot. In Proceedings of the 2nd International Conference on Networking, Information Systems & Security (pp. 1-5).

[10] Vollmer, T., & Manic, M. (2014). Cyber-physical system security with deceptive virtual hosts for industrial control networks. IEEE Transactions on Industrial Informatics, 10(2), 1337-1347.

[11] Lee, J., Jeon, J., Lee, C., Lee, J., & Cho, J. (2016). An implementation of log visualization system combined SCADA Honeypot. In International Conference on Advanced Communication Technology (ICACT).

Copyright © 2020 MECS

[12] Redwood, O., Lawrence, J., & Burmester, M. (2015, March). A symbolic honeynet framework for scada system threat intelligence. In International Conference on Critical Infrastructure Protection (pp. 103-118). Springer, Cham.
[13] Mashima, D., Chen, B., Gunathilaka, P., & Tjiong, E. L. (2017, October). Towards a grid-wide, high-fidelity electrical substation honeynet. In 2017 IEEE International Conference on Smart Grid Communications (SmartGridComm) (pp. 89-95). IEEE.
[14] Desel, J., & Reisig, W. (2015). The concepts of Petri nets. Software & Systems Modeling, 14(2), 669-683.
[15] Jensen, K., & Kristensen, L. M. (2015). Colored Petri nets: a graphical language for formal modeling and validation of concurrent systems. Communications of the ACM, 58(6), 61-70.
[16] Zhang, F., Xu, Y., & Chou, J. (2016). A novel petri nets-based modeling method for the interaction between the sensor and the geographic environment in emerging sensor networks. Sensors, 16(10), 1571.
[17] Siddavatam, I. A., Parekh, S., Shah, T., & Kazi, F. (2017). Testing and Validation of Modbus/TCP Protocol for Secure SCADA Communication in CPS using Formal Methods. Scalable Computing: Practice and Experience, 18(4), 313-330.
[18] Rodr ́guez, A., Kristensen, L. M., & Rutle, A. (2018, June). On Modelling and Validation of the MQTT IoT Protocol for M2M Communication. In PNSE@ Petri Nets/ACSD (pp. 99-118).
[19] Zhang, Y., Wang, W., Du, W., Qian, C., & Yang, H. (2018). Coloured Petri net-based active sensing system of real-time and multi-source manufacturing information for smart factory. The International Journal of Advanced Manufacturing Technology, 94(9-12), 3427-3439.
[20] kamal Kaur, R., Singh, L. K., & Khamparia, A. (2020). Modeling uncertainty of instrument and control system of nuclear power plant. Annals of Nuclear Energy, 139, 107207.
[21] Shi, L., Li, Y., & Feng, H. (2018). Performance analysis of honeypot with petri nets. Information, 9(10), 245.
[22] Buza, D. I., Juh ́asz, F., Miru, G., F ́legyh ́azi, M., & Holczer, T. (2014, February). CryPLH: Protecting smart energy systems from targeted attacks with a PLC honeypot. In International Workshop on Smart Grid Security (pp. 181-192). Springer, Cham.
[23] Yutao, F., Guiping, S., Liu, H., & Siyu, Z. (2012, December). Study on a CPN-based Auto-analysis Tool for Security Protocols. In 2012 Fourth International Symposium on Information Science and Engineering (pp. 179-182). IEEE.
[24] Boukredera, D., Maamri, R., & Aknine, S. (2012, June). A timed colored petri-net-based modeling for contract net protocol with temporal aspects. In Proceedings of the Seventh International Multi-Conference on Computing in the Global Information Technology (ICCGI 2012) (pp. 40-45).
[25] Zhu, L., Tong, W., & Cheng, B. (2010, October). CPN Tools' Application in Verification of Parallel Programs. In International Conference on Information Computing and Applications (pp. 137-143). Springer, Berlin, Heidelberg.
[26] Tare, B., Waghmare, S., Siddavatam, I., Kazi, F., & Singh, N. (2016, January). Security analysis of dnp3 using cpn model with state space report representation using lda. In 2016 Indian Control Conference (ICC) (pp. 25-31). IEEE.
[27] Caliz, E., Umapathy, K., S ́anchez-Ru ́z, A. J., & Elfayoumy, S. A. (2011, May). Analyzing web service choreography specifications using colored petri nets. In International Conference on Design Science Research in Information Systems (pp. 412-426). Springer, Berlin, Heidelberg.

Authors' Profiles

Sheetal Gokhale is pursuing her Master's degree in Information Technology (Information Security) at K.J. Somaiya College of Engineering, Mumbai. Her interests include Information security and vulnerability assessment.

Ashwini Dalvi is pursuing her PH.D. Degree from VJTI, affiliated to Mumbai University. She joined the Department of Information Technology, K. J. Somaiya College of Engineering, Mumbai in 2006 as an Assistant Professor. She has published over 25 journal and conference papers in the areas of Security, Intelligent applications.

Dr. Irfan Siddavatam received his PhD in Electronics Engineering from Veermata Jijabai Technological Institute, Mumbai in 2018. He is Associate Professor in Department of Information Technology at K.J.Somaiya College of Engineering, where he has been since 2001. His research interest includes Information and Cybersecurity, Artificial Intelligence, Machine Learning, Blockchain.

How to cite this paper: Sheetal Gokhale, Ashwini Dalvi, Irfan Siddavatam, "Industrial Control Systems Honeypot: A Formal Analysis of Conpot", International Journal of Computer Network and Information Security(IJCNIS), Vol.12, No.6, pp.44-56, 2020. DOI: 10.5815/ijcnis.2020.06.04

Copyright © 2020 MECS