Simulation of Command and Control Network Services Base on M2E2M

Pu An [1], Ying Chen [2]

[1] Science and Technology On Blind Signal Processing Laboratory, Chengdu ,610041,

[2]Sichuan International Studies University, Chongqing, 400031

Keywords: C2 networks; network simulation; Petri nets; VRNET Developer.

Abstract. With network simulation tools to simulate C2 (Command and control) network services, the simulated network will reflect the relationship between the internal units and dynamic behaviors of the real C2 network. However, the service operating mode exactly matches the organization structure of the C2 network, the service operating mode can't be extended to other C2 networks through the hard-code way. In this regard, this paper presents the M2E2M (Model to Event to Message) method. By drawing the ideas of colored, timed, object Petri nets, we propose the Petri nets based modeling method of C2 network services. By using CPN Tools to simulate the model, we generate communication events set and achieve the conversion from model to event. In the VRNET Developer simulation platform, we complete the simulation of the events set and achieve the conversion from event to message through the development of the CmdEngine and CmdApp module. Finally we complete the simulation of an early warning service of a C2 network. The simulation results match the network organization structure and the simulation method is proved to be effective.

Introduction

The C2 network geographically links decision making units, intelligence units and execution units of all layers. Through the C2 network, orders issue to lower layers and intelligence feedback to higher layers. C2 networks are featured by the following characteristics [1]:

1) The hierarchy of the structure is clear. The intelligence transmits from lower layers to higher layers and orders transmit form higher layers to lower layers.

2) The network responses quickly. All the network nodes react immediately after receiving a message from other node.

3) The service operating mode is fixed. C2 networks are built to run dedicated services. The service operating mode matches the organization structure, with a fixed information transmission process.

Using network simulation tools to simulate the C2 network, users can analyze the bottlenecks of the network, thus provide the basis for network optimization. Network simulation tools use distribution functions [2] to simulate the behavior of network users. However, using distribution functions is a statistical quantity based way. If we simulate the user behavior from the statistical quantity angle, rather than the C2 service, the simulated network can't reflect the transmission process of the information in the network, as well as the relationship between the internal units and dynamic behaviors.

Coding into the C2 node model by the hard-code way, researchers can achieve the simulation of the C2 network service. However, C2 network services match the C2 organization structure, once the structure has changed, researchers need to re-encode. To solve the poor scalability of the hard-coded method, we propose a M2E2M (Model to Event to Message) method. Through the modeling and simulation of the C2 network service, we get a structured set of communication events. By transforming events into packets which can transmit in the simulated network, we achieve the simulation of the C2 network service.

Petri Nets based Model of C2 Network Services

Modeling the C2 network service and generating communication events set by simulating the model is the first step of the M2E2M method, we complete it through Petri nets.

Petri nets [3] was proposed by Carl Adam Petri in 1962, which is a valid tool for describing distributed, concurrent, asynchronous systems. Basic Petri nets have four modeling elements: place, transition, arc and token. In the modeling process which is based on Petri nets, place is usually used to represent the state of the system, transition is usually used to represent the action, place, transition and arcs connecting place and transition are used to represent the static function and the structure of the system, the firing of transitions and the movement of tokens are used to describe the dynamics behaviors of the system. The reasons we choose Petri nets as modeling tool for C2 network services are as follows:

1) The characteristics that Petri nets can be used to describe distributed, concurrent, asynchronous systems make Petri nets suitable for describing discrete event systems. C2 networks are typical discrete event systems [1], static structures with order, concurrency and conflict features of network services can be described by places, transitions and arcs, dynamic executions of service processes can be described by the flow of tokens.

2) Using object Petri nets [4] to describe network services, and encapsulating the part of petri net belong to the same network node into an object subnet, make the model hierarchy, and combine the flows of control and information.

3) Nodes at the same layer of the C2 network play similar roles in the network service. Object subnets belong to nodes with the same social attribute can be merged into one subnet by colored Petri nets. Different individuals can be distinguished by different token colors. Through this way, we can solve the state space explosion problem of the Petri nets model.

4) Giving transition time delay makes Petri nets timed Petri nets [6], which can describe the system model from the time dimension.

5) CPN Tools [7] which is used to edit, simulate and analyze colored timed Petri nets can be used as a supporting tool for modeling and simulation.

C2 network services model based on Petri nets is a four-tuple:

$$\mathrm{CFM} = (\Sigma, O, R, M_0) \tag{1}$$

In formula (1), Σ is the set of colors, and is a four-tuple:

$$\Sigma = (N, Cmd, Div, Cpd) \tag{2}$$

In formula (2), N is the color set of C2 network nodes. Each node is corresponding with one color. Cmd is the color set of instructions. Div is a set of sub-color sets which are produced by dividing the color set N. The color of nodes with the same social attribute should belong to the same subset of colors, meeting:

$$N = Div_1 \cup Div_2 \cup \cdots \cup Div_n \tag{3}$$

$$Div_i \cap Div_j = \varnothing, 0 \le i \ne j \le n \tag{4}$$

Cpd is a product color set of these three colors.

In formula (1), O is a set of communication objects:

$$O = \{OBJ_1, OBJ_2, \cdots, OBJ_n\} \tag{5}$$

In the model of C2 network services, communication objects and the subset of Div are bijection. A communication object describes a class of network nodes with the same social attribute, relying on the token color to distinguish among different individuals. An object appears as a non-connected colored, timed Petri nets, and it can be described by an eight-tuple [8], which is used to describe a colored, timed Petri net:

$$\mathrm{CPTN} = (P, T; F, \Sigma, I_-, I_+, M_0(OBJ_i), TF) \tag{6}$$

In formula (6), P is the set of places, T is the set of transitions, F is the set of arcs, Σ is the set of colors, the same with Σ in formula (1), I_- is the set of negative functions of arcs, I_+ is the set of positive functions of arcs, $M_0(OBJ_i)$ is the initial marking of the object subnet OBJ_i, obey formula (1) in the whole nets, TF is the set of time delay on transitions. Object subnets have a special kind of transitions, as ports of the object. Object sends messages or receives messages to other objects or from other objects through ports.

In formula (1), R is the set of communication relationships among objects, which is a special kind of places with arcs and arc expressions:

$$R \subseteq (T' \times T') \tag{7}$$

meeting:

$$^{\bullet}R_k = T'_s \wedge R^{\bullet}{}_k = T'_r \tag{8}$$

In formula (8), $^{\bullet}R_k$ is the input set of R_k and $R^{\bullet}{}_k$ is the output set of R_k. $R_k \in R$, T'_s、$T'_r \in T'$, and $T'_s \neq T'_r$, T'_s is the port which initiate communication, and T'_r is the port which response communication. The color of R is a product color:

$$C(R) = Div_s \times Div_r \times \cdots \times Cmd \in Cpd \tag{9}$$

In formula (9), Div_s represents the initiator of the communication, and Div_r represents the recipient of the communication, Cmd represents the transmitted instruction.

In formula (1), M_0 is the set of initial makings of all object subnets, and also the initial making of the CFM, meeting formula (10):

$$M_0 = \sum_{i=1}^{n} M_0(OBJ_i) \tag{10}$$

The steps of using four-tuple in formula (1) to modeling C2 network services are as follows:

1) First, define the color set N. N is an enumerated color set, which is set of all nodes in the C2 network. Then get the color set Div by dividing N according to the social attributes of nodes. The elements of Div are subsets of N, meeting formula (3) and formula (4). Finally, define the color set of instructions Cmd in accordance with the instruction to transfer, which is a set of character colors.

2) Define object subnets according to Div, object subnets and the elements of Div are bijection. Use net [3] to describe the statistic structure with order, concurrency, conflict features of subnets, and define ports T'. The object subnet is not a real Petri net strictly, because it is not connected.

3) Use places R connect all ports T' according to the communication relationship among objects, meeting formula (8). Define the color type of R according to the definition of formula (9).

4) Define the initial making of the Petri net according to controlling information.

5) Define color types of all places and arc expressions according to the color type $C(R)$ of places and the initial marking M_0 of the model, and dynamically add product color set Cpd. Continuity and accessibility of the model should be ensured, and the deadlock phenomenon should not exist.

6) Give time delay to transitions. Time delay on communication ports T'_s which initiate communication and offset represent the start time of the communication activity. Time delay on communication ports T'_r which response communication and offset represent the end time of the communication activity.

7) Captures tokens in all places R which represent communication relationship, and sort the token from the time dimension. Then get the communication events set which can be converted to message packets. The format of events set is a four-tuple: (Sender, Receiver, instruction, tine).

The Simulation of C2 Network Services base on VRNET Developer

Using appropriate simulation tools, developing appropriate modules to read events form the events set, and simulating these events are the second step of M2E2M method. We choose VENET Developer as the simulation platform.

VRNET Developer [9] is a discrete network simulator, which is developed by VIRE with full Chinese development environment, with open, scalable, portability features.

VRNET Developer model consists of simple modules, which can communicate with messages. Simple modules can be nested within the composite module. Modules send message to other modules or receive messages from other modules via ports. Event is generated when any message is received by module, and the kernel run the simulation by executing events.

There are two reasons to consider in choosing VRNET Developer as the simulation platform of

C2 network services:

1) VRNET Developer is a typical discrete event simulator, and is suitable for the simulation of discrete set of communication events.

2)The modularization of network model of VRNET Developer makes it easier to simulate. When simulating C2 network services, users only concern with user-layer simulation, and the protocol-layerl simulation can be implemented by other modules.

The module of VRNET Developer is similar to a class in the object-oriented method. It encapsulates two special functions: initialize() and handleMessage(). Function initialize() is called when the simulation starts, and is used to initialize the module. Function handleMessage() is called when the module receives a message, implement the action the module should operate when receives a message.

A CmdEngine module is required to read the communication events set generated by Petri nets. In function initialize(), the CmdEngine module read the first event, package the information of source node, destination node, and instruction into a self-message, and send to itself at the corresponding communication time by function scheduleAt(). And at the same time it will receive the message, trigger the execution of function handleMessage(). In function handleMessage(), the CmdEngine module insert the self-message into events list via function msgQueue.insert(),set the sender and recipient of the message the source node, and the message becomes a self-message of the source node. Afterwards, function handleMessage() function continues to read the next event from the communication events set, package into a self-message and send to itself at the corresponding communication time. The contents of the message include the information of the source node, the destination node and the instruction. When receiving the message at appropriate time, function handleMessage() will be re-triggered.

Because the application layer protocol of C2 network services is unknowable, we only simulate the transport layer, the network layer and the data-link layer of the network protocols used by C2 network services. Assume that C2 messages were send based on UDP protocol. A C2 network node consist of four modules, of which udp module, networkLayer module and ethernetInterface module are used to implement the simulation of the transport layer, network layer and the data-link layer, while the cmdApp module is used to implement the simulation of user layer. When the cmdApp module receives the control message inserted in the events list by CmdEngine module, the function handleMessage() of the cmdApp module is called. In function handlemessage(), the cmdApp module will obtain the desired information of destination nodes and instructions. The cmdApp module will encapsulate the instruction into a new message, converts the destination node name to IP address, and send it to the lower udp module at the time it receives the self-message and the lower layers of the network will route the message to the destination node.

Simulation of Early Warning Service of C2 Network

The case is from the reference literature [10]. In this paper we expand it briefly. Suppose that a C2 network has three organization layers: a command center C has jurisdiction over three regional command posts P1, P2, P3. All the regional command post has jurisdiction over a sensor and two executors. Organization structure is shown in Figure 1:

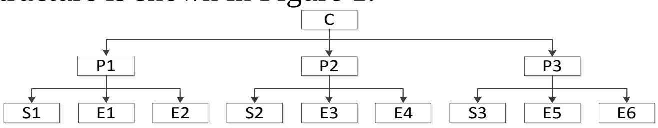

Fig .1 Organization structure of the C2 network

The early warning service of C2 network starts when any sensor of the network detects threat information, then it will send the threat information to the respective regional command post. The regional command Post will choose to handle the emergency by itself or ask for support from the adjacent units according to the situation it mastered. If it choose to handle by itself, it will send the operation task to the subordinate executors which are involved in this operation. If the regional command post consider it need support, it will send the threat information to the command center, and the command center will generate strategies according to the global situation and send strategies to executors which are involved in this operation gradually. After a period of time, all the executors will complete the mission and send execution reports to the command center gradually, then the process ends.

We choose CPN Tools as the modeling tool for the early warning service of the C2 network. The function of arc expression cmd(p, i) makes regional command post p choose a part of subordinate executors to participate in this operation through generating a random integer i. The part of subordinate executors can’t be empty. The function of arc expression Cmd (p, i) is similar to cmd(p, i), makes the command center C choose a part of regional command post to participate in this operation including the regional command post which request assistance through generating a random integer i. And the part of regional command post can’t be empty without p. Model of early warning service of C2 network base on Petri nets is shown in Figure2:

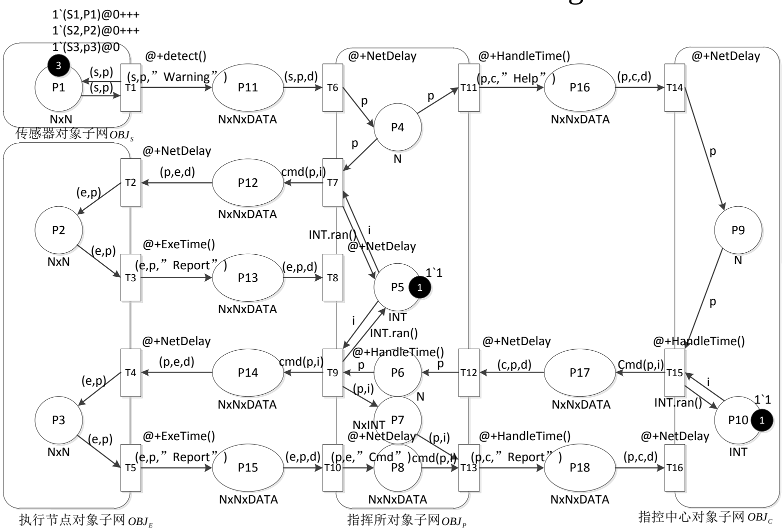

Fig.2 The Petri nets based model of early warning service of C2 network

In figure2, we know that:

$$O = \{OBJ_S, OBJ_E, OBJ_P, OBJ_C\} \tag{11}$$

$$R = \{\mathrm{P11,P12,P13,P14,P15,P16,P17,P18}\} \tag{12}$$

$$M_0(\mathrm{P1}) = 1`(\mathrm{S1,P1})\text{++}1`(\mathrm{S2,P2})\text{++}1`(\mathrm{S3,P3}) \tag{13}$$

$$M_0(\mathrm{P5}) = M_0(\mathrm{P10}) = 1`1 \tag{14}$$

Set the time sensors detect threat information uniform distributing from 50s to 100s, network transmission delay 2s, the time command post and command center generate strategies 10s, the time executors complete the mission 20s. Simulate the model, capture all tokens in place and sort in chronological sequence, get the communication events set.

Establish a C2 network, and simulate 100 pieces of communication events, set the size of package 100B, capture data in all links and count the traffic from end to end. The result is shown in the following matrix column.

As shown in Figure 3, the topology of the service and the organization structure of the network are highly matched. Thus, the early warning service reflects the relationship between the internal units and the simulation method is effective.

	C	P1	P2	P3	S1	S2	S3	E1	E2	E3	E4	E5	E6
C		600B	300B	600B									
P1	900B							700B	200B				
P2	500B									300B	200B		
P3	1K											400B	600B
S1		400B											
S2			400B										
S3				500B									
E1		700B											
E2		200B											
E3			300B										
E4			200B										
E5				400B									
E6				600B									

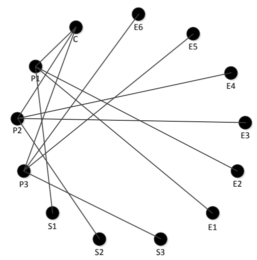

Fig. 3 Topology of the early warning service

Conclusions

The operating mode of the C2 network and the organizational structure are highly matched. It is not conductive to change the organizational structure by hard-code way to simulate the network service. In this regard, we propose a M2E2M method. It’s possible to change the organization structure and the service operating mode when using this method. Petri nets based modeling approach achieves the M2E step in M2E2M method, and successfully implement the conversion from model to communication events. By the development of CmdEngine and CmdApp module in VRNET Developer simulation platform, we achieve the E2M step in the M2E2M method, and also successfully implement the conversion from communication events to messages. Proven, simulation results and the assumption are highly matched. Thus the M2E2M methods is effective.

References

[1]Yang ruiping, Zhang zhaofeng. Simulation of C2 System[M]. Beijing: National Defense Industry Press, 2013.

[2]Long hua. OPNET Modeler and Computer Network Simulation[M]. Xinan: Xidian University Press, 2006.

[3]Murata T. Petri nets: properties, analysis and applications [J]. In: Proceedings of The IEEE, 1989, 77(4): 541-580.

[4]Zhou Chunlai, Li Zhigang, Wang Yongbin. Assembly organization model of 315ollaborative design based on object-oriented petri nets [C]. International Conference on Information Acquisition, 2006(2):770-775.

[5]K. Jensen. An Introduction to the Theoretical Aspects of Colored Petri Nets [J]. A Decade of Concurrency, Lecture Notes in Computer Science. Springer-Verlag, 1994, 803:230-272.

[6]Wu zhehui, Petri Nets Introduction, Beijing: China Machine Press, 2006.

[7]K Jensen. Computer verification tool-Colored Petri nets. http://wiki.daimi.au.dk/cpntools. 2010-10-01.

[8]Zhang zhanjun, Zhou xingqian, Network Simulation and Application base on VRNET Developer[J], Journal of Sichuan Ordnance,2013,34(2),96-99.

[9]Ding feng, Research on the Use of Colored Timed Petri Net and Stochastic Petri Net[D],Hefei University of technology master dissertation, 2009.

[10]Alberts, Garstka, Stein. Network Centric Warfare: Developing and Leveraging Information Superiority[M]． Washington DC: CCRP Publication Series, 1999, 12(3): 16-18．