Techniques for Supporting Prediction of Security Breaches in Critical Cloud Infrastructures Using Bayesian Network and Markov Decision Process

by

Vinjith Nagaraja

A Thesis Presentation in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved July 2015 by the
Graduate Supervisory Committee:

Sik-Sang Yau, Chair
Gail-Joon Ahn
Hasan Davulcu

ARIZONA STATE UNIVERSITY

August 2015

ABSTRACT

Emerging trends in cyber system security breaches in critical cloud infrastructures show that attackers have abundant resources (human and computing power), expertise and support of large organizations and possible foreign governments. In order to greatly improve the protection of critical cloud infrastructures, incorporation of human behavior is needed to predict potential security breaches in critical cloud infrastructures. To achieve such prediction, it is envisioned to develop a probabilistic modeling approach with the capability of accurately capturing system-wide causal relationship among the observed operational behaviors in the critical cloud infrastructure and accurately capturing probabilistic human (users') behaviors on subsystems as the subsystems are directly interacting with humans. In our conceptual approach, the system-wide causal relationship can be captured by the Bayesian network, and the probabilistic human behavior in the subsystems can be captured by the Markov Decision Processes. The interactions between the dynamically changing state graphs of Markov Decision Processes and the dynamic causal relationships in Bayesian network are key components in such probabilistic modelling applications. In this thesis, two techniques are presented for supporting the above vision to prediction of potential security breaches in critical cloud infrastructures. The first technique is for evaluation of the conformance of the Bayesian network with the multiple MDPs. The second technique is to evaluate the dynamically changing Bayesian network structure for conformance with the rules of the Bayesian network using a graph checker algorithm. A case study and its simulation are presented to show how the two techniques support the specific parts in our conceptual approach to predicting system-wide security breaches in critical cloud infrastructures.

Index terms- Critical cloud infrastructures, security breaches, pro-active protection, predictive defense, probabilistic reasoning, Markov decision process, Bayesian network structure conformance, and probabilistic human behaviors

ACKNOWLEDGEMENTS

I would like to thank Professor Stephen S. Yau for providing me motivation, guidance and the opportunity to work on this research project. I would also like to thank my committee members Professors Gail-Joon Ahn and Hasan Davulcu for taking the time to be on my graduate supervisory committee.

The invaluable inputs I have received for my thesis from Arun Balaji Buduru and other members of Professor Yau's research laboratory in the Information Assurance Center are very much appreciated.

I would like to thank my family and friends whose encouragement and support has motivated me throughout my research

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter 1

INTRODUCTION

1.1 Motivation

Emerging trends in cyber system security breaches in critical cloud infrastructures, such as military, finance, etc. show that major threats to critical cloud infrastructures are from large organizations and in some cases by foreign governments [1]. The sophistication of attacks reflect that the attackers are not limited by human resources and computing power. Hence, there is an urgent need to develop proactive defense approaches to protecting critical cloud infrastructures.

The security challenges in cloud infrastructure include insider threats, outsider malicious attacks, data loss, issues related to multi-tenancy, cryptographic key ownership, loss of control and service disruption [2]. Many of these challenges are addressed by existing reactive defense techniques, but require relatively large computational resources to apply all of them.

Most proactive defense existing approaches to protecting cyber infrastructures with predictive capability are based on applying game theory to generate adversarial models. All these approaches have difficulties of rationality and based on assumption that defender is always able to detect attacks. Some approaches also have difficulty of Nash equilibrium.

Additional difficulties in the approaches using game theory include the following: These approaches are not scalable with realistic sizes and complexity of the infrastructures. It is difficult for these approaches to capture the information on probabilistic human behaviors accurately. In addition, in these approaches, it is assumed that the actions of players (attackers and defenders) are synchronous, but it is not always the case.

A conceptual approach with predictive capability has been proposed [3] to enable the critical cloud infrastructures to prevent and mitigating security breaches efficiently. A framework has been proposed to use Markov Decision Process (MDP) and Bayesian network (BN) to predict system-wide security breaches based on the subsystem level security breaches and probabilistic human behaviors with respective time windows, in which security breaches are most likely to occur.

1.2 Contribution

The interactions between the dynamically changing state graphs of the MDPs of the subsystems of the critical cloud infrastructure and the dynamic causal relationships in BN of the critical cloud infrastructure is a key component to success in the approach [3]. In my thesis, two techniques are presented for supporting the prediction of potential security breaches in critical cloud infrastructures. The first technique is for evaluation of conformance of Bayesian network with the stochastic probabilistic model of each subsystem. The second technique is to evaluate the dynamically changing Bayesian network structure for conformance with the rules of the Bayesian network using

a Graph Checker Algorithm. A case study and its simulation are presented to show how the two techniques support the specific parts for the prediction of system-wide security breaches in critical cloud infrastructure.

1.3 Organization of Thesis

The thesis is organized as follows: Chapter 2 describes the current state of art for predicting security breaches in critical infrastructure. In Chapter 3 the conceptual approach for predicting security breaches in critical cloud infrastructure is discussed. In Chapter 4, the technique for evaluating the conformance of system-wide Bayesian network with the stochastic probabilistic model of each subsystem of our solution is presented. In Chapter 5, the technique for evaluation of the dynamically changing Bayesian network structure for conformance with the rules of the Bayesian network using our graph checker algorithm is presented. In Chapter 6, a case study and its simulation results are presented to show how the two techniques support the specific parts for the prediction of system-wide security breaches in critical cloud infrastructures. Finally, the conclusion and future research will be discussed in Chapter 7.

Chapter 2

CURRENT STATE OF ART

2.1 Game Theory Approaches

Most existing approaches to protecting cyber infrastructures with predictive capability are based on applying game theory to generate adversarial models. [4 - 10] All these approaches have difficulties of rationality and based on assumption that defender is always able to detect attacks. Some approaches also have difficulty of Nash equilibrium [4, 5].

The difficulties of rationalities in game theory is the assumption that all entities (user(s) and attacker(s)) make decisions rationally, in this case "rationally" means each entity wants to cause maximum damage with each decision the entity makes. This is not always the case.

In game theory, the Nash equilibrium is a solution concept of a non-cooperative game (players making independent decisions) involving two or more players, in which each player is assumed to know the equilibrium strategies of the other players, and no player has anything to gain by changing only the player's own strategy.

Additional difficulties in approaches using game theory include that the game theory approaches are not scalable with realistic sizes and complexity of the infrastructures, that it is difficult to capture probabilistic human behaviors accurately, and that they all assume the actions of players (attackers and defenders) are synchronous, but it is not always the case.

2.2 Attack Graph Approaches

Attack graph approaches are deterministic and use graph models for predicting possible attacks on IT infrastructures. MulVAL, NetSPA, GARNET and NAVIGATOR [11] are deterministic attack graph tools to model possible attacks on IT architectures using the output from network vulnerability scanners and the group policy of the infrastructure being analyzed. Each vulnerability identified by such tools is associated with a probability that represents how likely an attacker can successfully exploit it. The probability of successfully exploiting such a vulnerability by an attacker is denoted by the access-complexity value provided by the Common Vulnerability Scoring System (NVD) and the intuition by the domain expert of the IT infrastructure using the tool. The drawback of these approaches are based on the estimation of the output from vulnerability scanners, important attacks (e.g., by social engineering and discovery of novel vulnerabilities, such as zero-day attacks) and the effectiveness of defenses (e.g., antimalware and host firewalls) need to be depicted manually by the end-user. They also treat all identified vulnerabilities as directly exploitable by the attackers. This is a problematic assumption as various factors related to successful exploitation are not

gathered by network scanners, such as effectiveness of intrusion detection systems and attacker knowledge. These tools also do not include probabilistic human/operational behaviors in the IT infrastructure.

2.3 Probabilistic Modelling Approaches

The following probabilistic modelling approaches have been used to predict security and reliability issues.

- In [12 - 16], a probabilistic modelling approach for SCADA and power systems is presented to ensure that the power transmission systems remain at the acceptable reliability levels with various load by predicting failures on various generator and the random failures of system equipment.
- A thread-driven quantitative framework, called Three Tenets, was developed for secure cyber-physical system design and assessment by predicting attacks using Bayesian Network [17].
- An approach to situation assessment was developed for helping decision makers in intelligent operations and large-scale crisis management using Bayesian and Credal networks [18].

- A probabilistic reasoning modeling technique was developed using stochastic Petri Nets to understand the tradeoff between information security and operational performance in parallel distributed application environments focusing on moving target defense and deceptive defense tactics [19]. This technique generates a more secure platform by increasing the ratio of deceptive to operational nodes by changing the attack surface (the points of contact of attackers).

The difficulties encountered in these approaches [12 – 19] include that they do not capture human behaviors, they cannot detect co-related attacks, and they do not incorporate an attacker model. In addition, these approaches do not incorporate human factors, and hence the predictive capability is limited.

2.4 Approaches to Learning the Bayesian Network from Stochastic Probabilistic Models

Learning the Bayesian network from stochastic probabilistic models is required in Step 1) of our conceptual approach to construct the initial Bayesian network. There has been a number of approaches to learning the Bayesian network from data [20], and among these, a method to construct a Bayesian network from dependency network is presented. However, the generation of an optimal Bayesian network from a dependency network by is NP-hard. This method uses a heuristic technique to remove cycles in a greedy algorithm to get a Bayesian network. The generated Bayesian network uses a scoring function to calculate the Bayesian network structure with the least score, i.e. the

removal of a set of edges that has least effect on the Bayesian network prediction accuracy. The results of this approach showed experimentally that the Bayesian networks produced from dependency network has a prediction accuracy almost equaling that of Bayesian networks learned from data directly using standard datasets. Such techniques do not learn Bayesian network from more complex systems such as the MDPs, also this heuristic method cannot be applied for a dynamically changing MDPs and Bayesian network because of the computational complexity of the greedy approach.

In [21] a technique called structured policy iteration is introduced to represent stochastic actions in an MDP using Bayesian networks, together with a decision-tree representation of rewards using approximation techniques. This obviates the need for state-by state computation, aggregating states at the leaves of these trees and requiring computations only for each aggregate state. It does explain how best to prune the decision-tree and the pruning is strongly influenced by the variable ordering in the tree. Also, the classification technique applied for finding the smallest decision tree is not feasible in dynamically changing MDPs due to computational complexity. This technique learns Bayesian network only from a single MDP and not suitable for our problem of generating Bayesian network from multiple MDPs.

The above techniques have high computational complexity and do not learn Bayesian network from multiple MDPs to solve our problem. In chapter 4, we show our approach for evaluation of the conformance of the Bayesian network with multiple stochastic probabilistic models.

2.5 Approaches to Verification of the Bayesian Network

Evaluation of the dynamically changing Bayesian network is required in Step 1) and Step 3) of our approach to verify the conformance between the Bayesian network and the Bayesian properties.

In [22] a technique called the Minimum Feedback Arc Set is presented to detect and remove all cycles of a directed acyclic graph. The solution is NP-hard and hence not applicable in practical applications. Also this technique does not consider the Conditional Probability Distribution requirements of a child node in a Bayesian network with respect to the number of parent nodes for a particular child node. Hence this technique cannot be used to solve our problem of verify the conformance of a Bayesian network.

In Chapter 5, we will present an approach to verify the conformance of a Bayesian network by developing a Graph Checker Algorithm to solve our problem of evaluating the dynamically changing Bayesian network structure.

The following is a list of drawbacks of the existing approaches discussed above:

For approaches with predictive capability based on game theory,

1. Rationality is assumed, but the attacker does not always have the make the most damaging move.
2. Nash equilibrium is assumed, but the attackers and defenders can be benefited by changing strategies.
3. Not scalable with realistic sizes and complexity of the infrastructures.
4. Difficult to capture probabilistic human behaviors accurately.

5. Actions of players (attackers and defenders) are assumed to be synchronous.

For approaches using attack graphs,

6. Do not capture personalized probabilities, the probabilities in the transaction matrix of MDP subsystems after reinforcement learning for critical cloud infrastructures.
7. Vulnerability probabilities are manually input by an expert into the attack graph approach, the dynamic nature of critical cloud infrastructure forces the expert to determine all vulnerable probabilities again on the next prediction of attack graph approaches.
8. Successful attacks might not be gathered by network scanners and other vulnerability detection mechanisms which provide input to the attack graph approaches.
9. Attack graph approaches treat all identified vulnerabilities as directly exploitable by the attacker. CVSS, CVE values are not personalized for a particular cloud environment.
10. Successful exploitation is not gathered by network scanners such as effectiveness of intrusion detection systems and attacker knowledge.

Chapter 3

CONCEPTUAL APPROACH

Emerging trends in cyber system security breaches in critical cloud infrastructures, such as in military, finance, etc. show that major threats to critical cloud infrastructures are from large organizations and in some cases by foreign governments [23]. The sophistication of attacks reflect that the attackers are not limited by human resources and computing power. Hence, there is an urgent need to develop proactive defense approaches to protecting critical cloud infrastructures. In my thesis, two techniques are presented for supporting the approach presented in [3] to predict system-wide security breaches by the Bayesian network and the probabilistic human behavior in the subsystems is captured by the Markov Decision Processes of potential security breaches in critical cloud infrastructures. A case study and its simulation are presented including the above 2 techniques to show the results of the conceptual approach to predicting system-wide security breaches in critical cloud infrastructure.

The security challenges in cloud infrastructure include insider threats, outsider malicious attacks, data loss, issues related to multi-tenancy, cryptographic key ownership, loss of control and service disruption [24]. Many of these challenges are addressed by existing techniques, but require relatively large computational resources to apply all of them. In addition, security challenges for critical cloud infrastructures include:

- C1) User-centric Security Systems: Incorporating users' preferences and behaviors, including certain degrees of prediction of human behaviors, is needed for developing user-centric security systems for critical cloud infrastructures.
- C2) Emergence of Software Defined Network (SDN): From security point of view, the centralized control structure of SDN is its most serious disadvantage because moving the network from relatively decentralized structure to centralized controller environment will significantly increase the risk of potential single point of attack and causing catastrophic failure.
- C3) Overhead for Security Measures: High efficiency and speed of many tasks of critical cloud infrastructures are required for reliable and quick response. Hence, the security measures for critical cloud infrastructures must introduce little overhead, especially because current security measures need to run all the time when the critical cloud infrastructures are in service.

To address these challenges, an effective conceptual approach with predictive capability is proposed in [3] to enable the critical cloud infrastructures to prevent and mitigating security breaches efficiently. In such approaches, accurate predictive capability, accurate threat assessment and operational behavioral modelling of the critical cloud infrastructures are necessary. Probabilistic human behaviors affect the system operational behavior, a frameworks is introduced using MDP and Bayesian network to facilitate the capturing and analyses of probabilistic human behaviors accurately and efficiently.

MDPs is viewed as a stochastic automata where, an MDP state is the representation of the state of the system and actions in the MDP represent the uncertainty of inducing stochastic transitions between MDP states. The expected value of a certain course of action is a function of the transitions it induces between states, with rewards associated by moving to an MDP state. Plans can be optimized by policy iteration [25] or value iteration [26] over a fixed finite period of time. These make MDPs ideal models for capturing and analyses of probabilistic human behaviors accurately and efficiently for critical cloud infrastructures [27 - 30]

Bayesian network [31 - 33] is a framework for representing a probability distribution in factored form. It is represented by a directed acyclic graph with vertices corresponding to random variables and a directed edge between two variables indicating a direct probabilistic dependency between them. A Bayesian network also reflects implicit independencies among the variables. The Bayesian network must have a probability distribution for each immediate parent vertex conditioned on it. In addition the network must include a marginal distribution for each vertex that has no parents. The probability of any event over the Bayesian network is computed using algorithms [34] that exploit the independencies represented in the graph structure.

In this conceptual approach [3] we use probabilistic techniques, such as Bayesian network and MDP to predict system-wide security breaches based on the probabilistic inputs on subsystem level security breaches. The conceptual approach will be time based, and generate probabilistic predictions of system-wide and sub-system security breaches

with the respective time windows, in which breaches are most likely to occur. The domain expert will analyze the network of the critical cloud infrastructure accurately, and provide the information to the state construction algorithms using a data specification language.

A critical cloud infrastructure may have multiple subsystems and the number of subsystems is determined by the application (domain knowledge). Each subsystem may or may not be connected to other subsystems directly but are required to communicate with the Bayesian network, either directly or indirectly which is run on a centralized system or a subsystem connected to all the subsystems. The Bayesian network will monitor all the events which occur across subsystems, and predict system-wide security breaches using the following steps:

Step 1) Based on system state dependencies of critical cloud infrastructure (identified from application domain knowledge), construct and evaluate system-wide BN and state graphs of subsystem MDP. The probabilities in MDPs and BN can be set with arbitrary initial values and updated in next steps. The domain expert determines the threshold for probability of occurrence of each known security breach

Step 2) Monitor the critical cloud infrastructure to observe its operational behaviors, including human behaviors. Update the MDP and BN state graphs based on causal relationships among the observed operational behaviors using Bayesian probability estimation algorithm (BPEA) and MDP probability estimation algorithm (MPEA).

Step 3) Check the conformance of the BN structure with Bayesian properties, MDP state graph structure with Markovian properties and to each other. Update BN and MDP state graphs

Step 4) Estimate accuracy of probabilities in MDP and BN state graphs by deploying the MDPs and BN of critical cloud infrastructure "passively", and then insert known security vulnerabilities in the infrastructure

Repeat Steps 2) to 4) until the probabilities in the state graphs of MDPs and BN are tuned to a point where the accuracy of the prediction of system breaches is deemed "good enough" by the system administrator to be deployed in real-time.

Step 5) Run the MDPs and the BN inference engine to predict security breaches at subsystem level and system-wide level, respectively, along with a time window. If the probability of a predicted security breach exceeds the specified threshold in Step 1) for the security breach, the security breach is predicted as to occur soon

In this thesis we focus on the following:

- Step 1), evaluation of conformance of the system-wide Bayesian network with subsystem MDPs.
- Step 3), Check the conformance of the BN structure with Bayesian properties by developing a Bayesian network Graph Checker Algorithm. Also used in Step 1).

- Step 5), A case study and its simulation are presented including the above 2 techniques to show the results of the conceptual approach to predicting system-wide security breaches in critical cloud infrastructure.

Our conceptual approach intends to address these drawbacks as follows:

1. We address drawback 1 and 2 by using the personalized probabilities of transaction matrix for MDP Subsystems to make decisions, and not finding equilibrium and only predicting the best possible move by attacker.
2. Drawback 3 is addressed by divided MDP into subsystems, reducing the MDP states compared to a single system-wide MDP.
3. Not having the assumptions made in drawback 1 & 2 allows MDP to efficiently capture human actions and operational behaviors, solving drawback 4 & 5.
4. Drawback 6 is addressed in our conceptual approach by utilizing the subsystem MDP to capture human and operational behavior of a particular MDP subsystem by modified reinforcement learning in Steps 2) – 4) in our approach.
5. Drawback 7 is not completely mitigated by our conceptual approach, a domain expert still has to input data into our system but his involvement is highly reduced as we require him to only provide input in the Step 1) and selected special case instances in the subsequent steps (such as non-conformance of MDP and Bayesian network).

6. We also assume the attacks are detected and the applied defense mechanisms are efficient according to drawback 8 and 10, but if the expert is not sure about the applied defense mechanisms he can input his uncertainty by changing the probability values.
7. Drawback 9 is addressed by MDP capturing values specific to a subsystem. Here the effect of a particular global CVSS or CVE is modified by the probability values captured by the subsystem.

Other major advantages of our conceptual approach:

1. Our prediction approach uses MDP for subsystems prediction and Bayesian network for system-wide prediction. The MDP captures human/operational behavior and predicts subsystem breaches based on the current state only, while the Bayesian network predicts/infers system-wide security breaches based on the entire network of vulnerable states.
2. Anomaly based detection mechanisms integrate well into our approach as the uncertainty of anomaly based approaches can be represented by probability values of MDPs and Bayesian network conditional probabilities.

Chapter 4

EVALUATION OF CONFORMANCE OF THE SYSTEM-WIDE BAYESIAN NETWORK WITH THE STOCHASTIC PROBABILISTIC MODEL OF EACH SUBSYSTEM

This chapter explains the technique used to evaluate the conformance of system-wide Bayesian network from the subsystem MDPs. In 4.1 we present the approach to check the conformance of Bayesian network with multiple subsystem MDPs and its pseudocode. In 4.2 we present an illustration of our approach.

4.1 The Evaluation Technique

Here we show part of Step 1) and 3) of our conceptual approach for evaluation of system-wide Bayesian network with MDPs. An MDP state is a representation of the state of the critical cloud infrastructure. Ex: different possible configuration states of the firewall, different group policies etc. States with high reward values in the MDP are vulnerable states. The vulnerable states in each subsystem running MDP are used to construct the Bayesian network. Each vulnerable state is a node in Bayesian network. Due to dynamic change in the operational behavior in subsystem and the causal relationship in the Bayesian network such a conformance check is required. To check the conformance between the MDPs and the BN the following rules should be in valid in the BN between the MDPs.

- Rule 1: Every MDP vulnerable state must be a unique Node in the BN.
- Rule 2: If there exists a direct path between 2 MDP vulnerable states. The two MDP states must be in the *Markov blanket* of each other in the BN. The states in the MDP which have a relationship in the subsystem are not independent of each other in the BN, hence these vulnerable states should be in the Markov blanket of each other.
- Rule 3: BN should not contain other non-vulnerable MDP states.

Markov blanket:

- Of a node is the parents, its children, and its children's other parents.
- The nodes in the blanket are the only knowledge needed to predict the behavior of that node.

- **Stage 1:**
    - Vulnerable MDP states reward threshold T, subsystem MDPs and the BN is passed as input.
- **Stage 2:**
    - Iterate each MDP subsystem and its States,
    - if the reward value for the MDP state is greater than the threshold T, it is identified as a vulnerable state
    - Check Rule 1 by checking if MDP State is present in BN (either by Map or naming convention).
    - Store vulnerable state to check for Rule 2.

- **Stage 3:**
    - Loop each MDP subsystem vulnerable state and check for Rule 2.
    - Direct path between two vulnerable MDP states is checked by iterating over the transaction matrix of the specific MDP vulnerable states.
- **Stage 4**:
    - The BN is verified for Rule 3 by subtracting the System-wide security breaches from the BN and comparing it to the amount of MDP vulnerable states.

Pseudocode below shows the algorithm to evaluate the BN to be verified for conformance with subsystem MDPs. The validated graph is the Bayesian network.

Input for the below pseudocode are as follows,

- MDP[n] all MDP subsystems.
- BN the Bayesian network constructed but not verified
- T the threshold for vulnerable states based on reward value.

Algorithm: Conformance between BN and MDPs

```
1.  BN_MDP_ Conformance (MDP[n], BN, T)
2.     initialize vulnerableStateCount =0;
3.     Initialize systemWideCount =0;
4.     Initialize VulnerableStates[n][];
5.     Boolean isMarkovBlanket;
6.     Loop reward set R{S1:r1,  S2:r2, …} for each subsystem MDP[j]
7.       if Reward ri >= Threshold provided by Domain expert T of state Si
8.          vulnerableStateCount ++;
9.          V = Get Node MDP[j]_Si from BN
10.         if V = null                                  // Rule 1
11.            return Error Vulnerable state in MDP[j]not added in BN
12.         vulnerableStates[i][k++]    // K=0 for each new MDP subsystem
13.    End Loop
14.    Loop vulnerableStates[i] and size(vulnerableStates[i] > 1)  for each MDP subsystem
15.       isMarkovBlanket = IsMarkovBlanket(vulnerableStates[i][]) // Rule 2
16.       if isMarkovBlanket = false
17.          return Error nonconformance of Rule 2
18.    End loop
19.     Loop BN nodes for SW_[i]
20.      systemWideCount++;
21.    if (size(BN) – systemWideCount) != vulnerableStateCount)
22.       return Error BN contains states that are not vulnerable
```

Algorithm: Conformance between BN and MDPs – Rule 2

```
1.  isMarkovBlanket(vulnerableStates[j])
2.     Loop k each element in vulnerableStates array
3.          if transactionMatrix[vulnerableState[k]] contains any of vulnerableStates[0 to
    m]
4.                    // direct dependence but not self.
5.            if not markovBlanket(vulnerableState[k],vulnerableState[m]);
6.                return Error not in Markov blanket direct dependence.
```

Figure 1: The Pseudocode for Evaluating the Conformance of the Bayesian Network of a Critical Cloud Infrastructure with the MDPs of its Subsystem

End of Technique 1 generates a Graph G from the MDPs.

Time complexity of the above algorithm is O(vd)

- v is the number of vulnerable states
- d is the dependencies of each vulnerable state v given by the domain expert.

4.2 An Illustrative Example

In this section, we illustrate the process of evaluation for the conformance of Bayesian network from the stochastic probabilistic model. Consider the critical cloud infrastructure of the bank handling online personal transactions has 4 subsystems.

The MDP of subsystem 1 is constructed from the initial state dependencies from the domain expert as shown in figure 2.

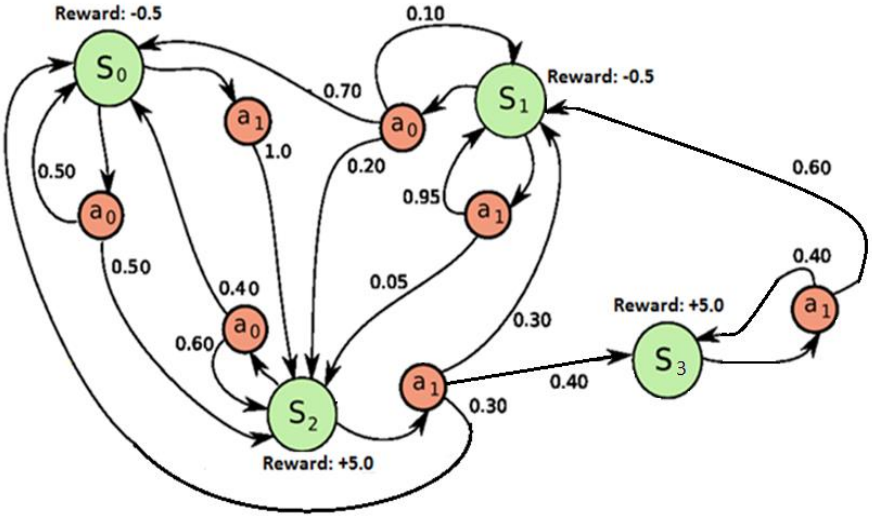

Figure 2: The MDP State Diagram of a Subsystem

A Markov decision process (MDP) is a 4-tuple: <S, A, P, R> i.e. states, actions, transaction matrix and rewards respectively

In our illustration (Figure 1),

- S is a finite set of states i.e.{S_0, S_1, S_2, S_3}
- A is a set of actions available from each state in S

  {S_0:[a_0,a_1], S_1:[a_0,a_1], S_2:[a_0,a_1], S_3:[a_1]},

  Where each state has two possible actions a_0, a_1 except S_3which only has actiona_1
- P is a transaction matrix containing the probability that action *a* in state S_i at time *t* will lead to state S_j at time *t+1*, represented by

  $P_a(S_i, S_j) = P(S_{t+1} = S_j \mid S_t = S, a_t = a)$

State\Action	**a_0**	**a_1**
S_0	[0.5, S_0], [0.5,S_2]	[1.0, S_2]
S_1	[0.1, S_0], [0.7,S_1],[0.2,S_2]	[0.05, S_2], [0.95,S_1]
S_2	[0.4, S_0], [0.6,S_2]	[0.3, S_0], [0.3,S_1],[0.4,S_3]
S_3		[0.6,S_1],[0.4,S_3]

Table 1: The Transaction Matrix of the MDP Subsystem shown in Figure 2

- *R(s)* is the immediate reward (or expected immediate reward) received after transition to state *s*

  R = [S_0 = -0.5], [S_1 = -0.5], [S_2 = +5.0], [S_3 = +5.0]

State 1:

The threshold for vulnerable states in the MDP is set to +0.3 by the Domain expert.

Stage 2:

MDP State 2 and MDP State 3 are identified as vulnerable states in MDP subsystem 1 because of high reward values. The Identified vulnerable states in MDP subsystem are verified to exist in the Bayesian network.

Vulnerable states S2 and S3 are stored to check rule 2.

There exists a direct path between MDP state S2 (BN node 1) and S3 (BN node 5) through action a1, hence they must be in the Markov Blanket of each other in the BN. Here both BN nodes 1 & 5 are parents of the same child 9 BN node; therefore they are in the Markov blanket of each other.

BN node	Description
1,2	Vulnerable states in MDP Subsystem 1
3,4	Vulnerable states in MDP Subsystem 2
5,6	Vulnerable states in MDP Subsystem 3
7,8	Vulnerable states in MDP Subsystem 4
9 - 14	Probable System-wide security breaches

Table 2: The Nodes and their Description of Bayesian Network shown in Figure 3

Stage 3: The Identified vulnerable states in the MDP are added to the graph using the algorithm shown in next section. If there is no error thrown from the algorithm i.e. all the values in the dependency map are correct, the below graph is generated shown in Figure 3.

The following is a partial dependency map from Domain expert:

Dependency map = {[Sub1_S2:SW_9, SW_10], [Sub1_S3: SW_11], …}

Here,

- Sub1_S2 is the State S2 in subsystem 1 having a dependency to System-wide breach SW_9 and SW_10.
- Sub1_S3 is the State S3 in subsystem 1 having a dependency to System-wide breach SW_11

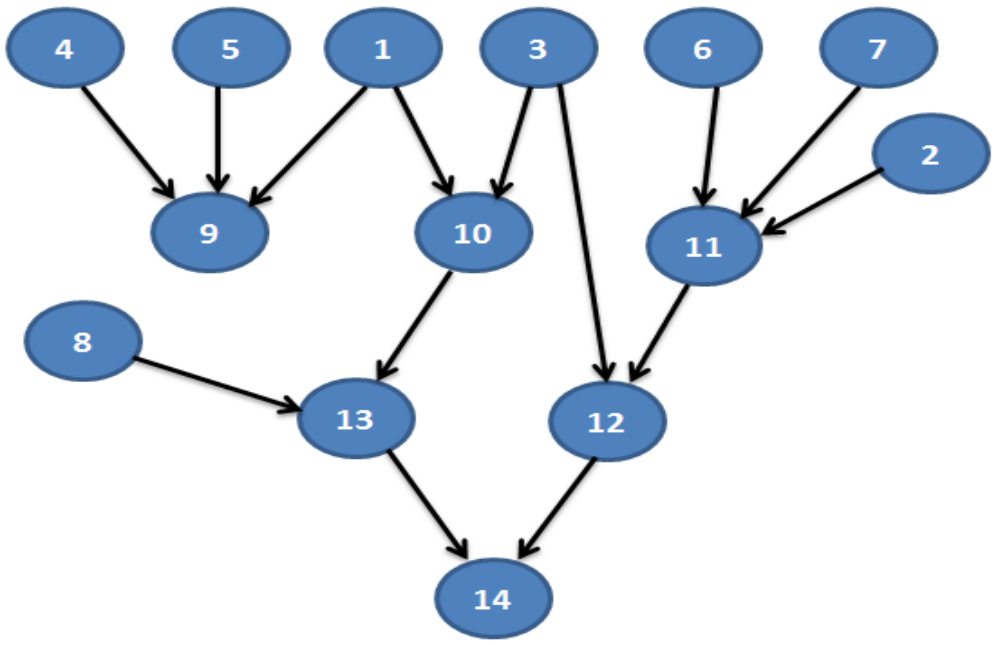

Figure 3: The State Graph of a System-wide Bayesian Network of the Critical Cloud Infrastructure

Node	conditional probability distribution (CPD)
1	[0.9, 0.1]
2	[0.9, 0.1]
3	[0.9, 0.1]
4	[0.9, 0.1]
5	[0.8, 0.2]
6	[0.9, 0.1]
7	[0.8, 0.2]
8	[0.9, 0.1]
9	[1.0, 0.9, 0.9, 0.7, 0.9, 0.8, 0.95, 0.7, 0.3, 0.05, 0.2, 0.1, 0.1, 0.3, 0.1, 0.9]
10	[1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]
11	[1.0, 0.9, 0.9, 0.7, 0.9, 0.7, 0.9, 0.95, 0.05, 0.1, 0.1, 0.3, 0.1, 0.3, 0.3, 0.9]
12	[1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]
13	[1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]
14	[1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]

Table 3: The Conditional Probability Distribution of the Bayesian Network shown in Figure 3

Chapter 5

EVALUATION OF THE DYNAMICALLY CHANGING BAYESIAN NETWORK STRUCTURE

5.1 Graph Checker Algorithm and Pseudocode

The verification of the Bayesian network/graph with the rules of the Bayesian network structure is required because of the need to verify the conformance of the initial graph generated from the MDP subsystems in Stage 3 of chapter 4 and due to the dynamically changing Bayesian network structure in Step 3) of our conceptual approach.

Rules of Bayesian Network structure are:

Rule 1: BN is a Directed Acyclic Graph.

Rule 2: Each child node of the Bayesian network should have 2 power (number of parents+1) entries in its conditional probability distribution (CPD) considering the each node has Boolean entries in CPD (The most common representation [31 - 33]). In some cases it is 2 power (number of parents) depending on optimized representation.

Rule 3: All the vulnerable states in each MDP subsystem i.e. the nodes in the Bayesian network should be connected either directly or indirectly to all nodes in the BN.

The nodes in the loops of the Bayesian network and unconnected nodes should be notified to Domain Expert.

A recursive function is used to find the cycle in the Bayesian network and to check connectivity using a Depth First Search approach. The recursive function returns both the cycle in the graph if present and also the nodes not connected in the graph.

Stage 1: Provide the graph to be verified for conformance to the isBayesian() function. Create arrays Vertex_marker and Stack_marker of the size of the graph. The Vertex_marker and Stack_marker represent the nodes connected in the graph and the current nodes in the possible cycle respectively.

Stage 2: Loop for each node in the graph in isBayesian() function for each node 'n' it returns the node 'n' if the node does not have the appropriate number of CPT entries in the node (line no. 7 in pseudocode) or if the node is not connected (line no. 9 in pseudocode by checking inDegree and outDegree of node) it the Bayesian network. Here Rule 2 and Rule 3 are checked.

The isBayesian() function then calls the recursive DFS isCycle_util() function if the node has not been visited. The isCycle_util() function performs DFS on a node and returns True if a cycle is present and also returns the possible nodes in the cycle. If no cycle is present it returns false with all the nodes visited. Here Rule 1 is checked.

Stage 3: If the above isBayesian() function returns True, It also returns a list of nodes of the graph containing all the possible nodes in the loop. DFS is performed on one of the nodes to extract the exact list of elements and their direction. Else If there is no cycle the isBayesian() function returns False and null. To satisfy Rule 3, the indegree and outdegree of the graph are considered the same to perform a Depth or Breadth first search to check for connectivity of the graph. If all the nodes are visited the graph is connected.

If there is more than one loop the algorithm needs to be run again till all loops are detected and eliminated by the domain expert.

The Time complexity of this function is O(n+e)

- Where n is the number of nodes in the Graph G
- E is the number of edges in the graph

Algorithm to check conformance of Bayesian network

```
1.  def isBayesian(Graph G):
2.      Set vertex_marker[size(G)] to False
3.      Set stack_marker[size(G)] to False
4.      Loop each node n in G:
5.          parents_count = getInDegree(n)
6.          if(parents_count != (2^parents_count+1)
7.              return False,Null,n        // CPT of node 'n' not in conformance
8.          if vertex_marker[n] is False:
9.              if isCyclic_util(G, n, vertex_marker, stack_marker) function returns True
10.                 return True, stack_marker,Null
11.     return False, Null,Null


12. def isCyclic_util(Graph G, node n, vertex_marker, stack_marker):
13.     Set vertex_marker[n] to True
14.     Set stack_marker[n] to True
15.     loop neighbors 'i' of n in G:
16.         if vertex_marker[i] is false and isCyclic_util(G, i, vertex_marker, stack_marker) is
        True:
17.             return True
18.         else if stack_marker[i] is True:
19.                 return True
20.     Set stack_marker[n] to False
21.     return False
```

Figure 4: The Pseudocode of the Graph Checker Algorithm for Evaluating the Conformance of the Bayesian Network with the Bayesian Properties

5.2 An illustrative Example

Illustration 1:

Consider Figure 5 shown below is the graph to be verified for conformance with the rules of Bayesian network.

Stage 1: Arrays Vertex_marker and Stack_marker is set to size 10 and initialized to false.

Stage 2: The isBayesian() function is called and it checks if Rule 2 and Rule 3 for each iteration of each node, if conformance to Rule 2 and Rule 3 fails it returns the node. Then the isCyclic_Util function is called and it returns True if the DFS on the node finds a cycle along with a list of nodes marked True in the Stack_marker function. The nodes marked True are Node 4, Node 8, Node 2, Node 3 and Node 7. Here the graph is connected and we assume the CPD is correct (Illustration 2 shows this test case).

Stage 3: A DFS is performed on one of the nodes marked True. When two duplicate nodes are detected by the DFS, the cycle and its direction is notified to the domain expert. The cycle retrieved here is between nodes 8 -> 2 -> 3 -> 7 ->8.

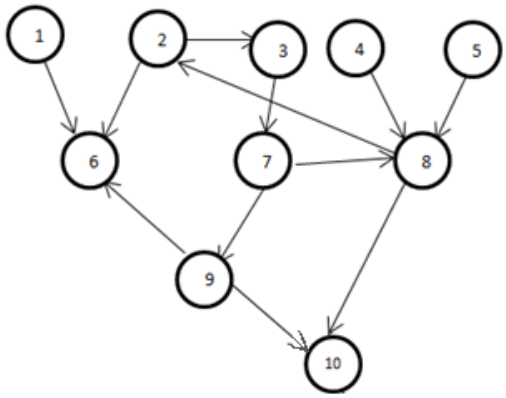

Figure 5: The Updated Bayesian Network to be Verified for Conformance with Bayesian Network

Illustration 2:

Consider Figure 6 shown below is the graph to be verified for conformance with the rules of Bayesian network.

Stage 1: Arrays Vertex_marker and Stack_marker is set to size 14 and initialized to false.

Stage 2: The isBayesian() function is called on each iteration it checks the conditional probability distribution table of each node. Here node 9 has 3 parents and should consist of 3^4 i.e. 16 CPD values. It contains only 14 values, this node is returned by the isBayesian() function.

Stage 3: Node 9 is notified to the domain expert and is requested to fix the CPD values of node 9.

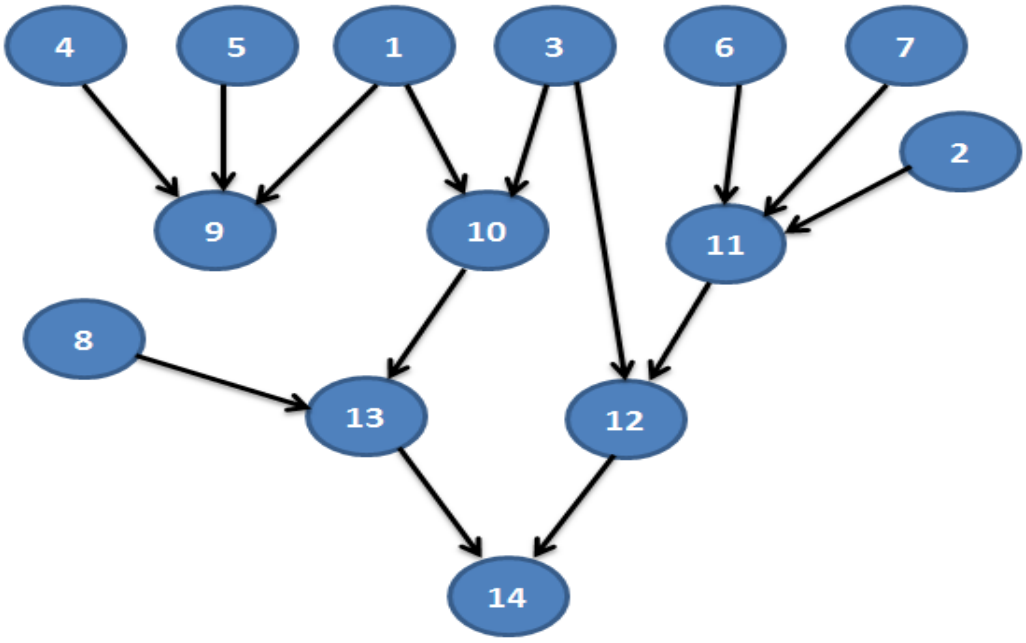

Figure 6: The State Graph of the Bayesian Network to be Verified for Conformance with Bayesian Properties

Node	conditional probability distribution (CPD)
1	[0.9, 0.1]
2	[0.9, 0.1]
3	[0.9, 0.1]
4	[0.9, 0.1]
5	[0.8, 0.2]
6	[0.9, 0.1]
7	[0.8, 0.2]
8	[0.9, 0.1]
9	[1.0, 0.9, 0.9, 0.7, 0.9, 0.95, 0.7, 0.3, 0.05, 0.1, 0.1, 0.3, 0.1, 0.9]
10	[1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]
11	[1.0, 0.9, 0.9, 0.7, 0.9, 0.7, 0.9, 0.95, 0.05, 0.1, 0.1, 0.3, 0.1, 0.3, 0.3, 0.9]
12	[1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]
13	[1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]
14	[1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]

Table 4: The Conditional Probability Distribution of the Bayesian Network shown in Figure 6

Illustration 3:

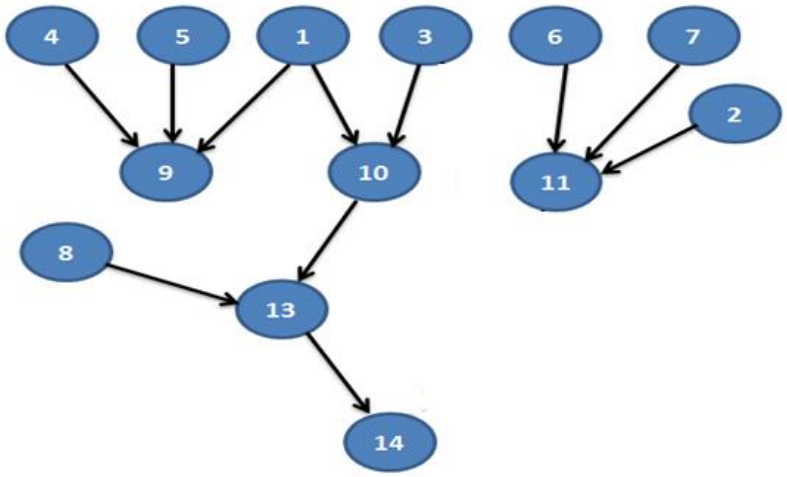

Figure 7: The Bayesian Network for the Critical Cloud Infrastructure of Illustration 3

The BN is in illustration 3 is disconnected.

Consider Stage 1 and 2 are valid as shown in previous illustration.

Stage 3: To check the connectivity of the BN we perform a DFS on any node by considering the BN to be an undirected graph.

To do this the indegree nodes and the outdegree nodes of a particular node is passed to the stack of the DFS as shown in algorithm above.

A count is made on the nodes visited and checked against the size of the BN. If there is a difference between the two then the graph is disconnected. Here there is a difference of 4.

The advantages here are you do not have to create a new undirected graph from the existing directed graph since we only check for connectivity of the network.

Chapter 6

CASE STUDY AND SIMULATION FOR THE TECHNIQUES

6.1 Overview of the Case Study

In this section a case study and simulation is conducted to show the prediction of security breaches in online personal transactions in a bank critical cloud infrastructure using the two techniques presented in chapter 4 and 5. The MDP subsystem prediction code is programmed in Python v2.7 and the Bayesian network system-wide prediction code is programmed in Matlab v2012b. Section 4.2 shows the case study of using online personal transactions in a bank critical cloud infrastructure and the simulation of the predictive capability of the critical cloud infrastructure. The simulation focuses on Step 1), the construction, Steps 1) & 3), verification of the Bayesian network and Step 5), the prediction of our approach including the two techniques presented in section 4 and 5.

6.2 Simulation Results

The case study of online personal transactions in a bank critical cloud infrastructure implementing our predictive conceptual approach is shown below. The critical cloud infrastructure consists of four subsystems (Figure 8):

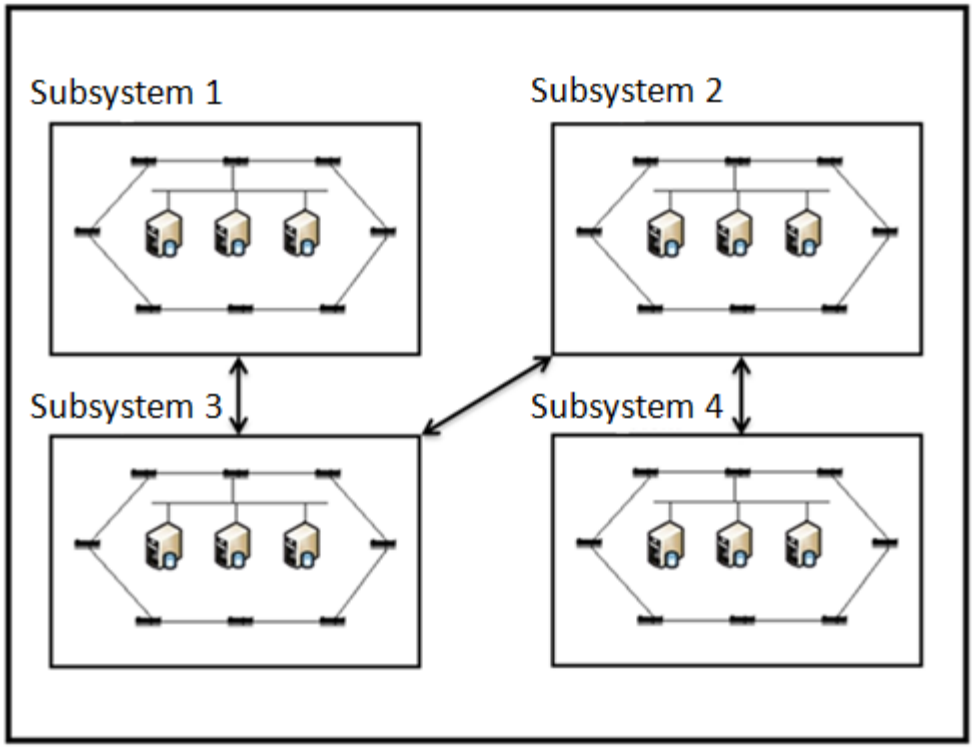

Figure 8: The Critical Cloud Infrastructure of Bank application

– ***Subsystem 1*** provides interface of the bank for personal transaction services to all its customers. It also checks possible attacks from customers' input, such as DoS, CSRF, XSS and SQL injections, but does not store customers' identity and personal transaction data. It forwards the validated requests to ***Subsystem 3***.

– ***Subsystem 2*** stores anonymized customers' transaction data received from ***Subsystem 3*** and sends acknowledgement back to ***Subsystem 3***.

– ***Subsystem 3*** revalidates customers' requests, processes transactions and sends the transaction data in anonymized form for storage in ***Subsystem 2.*** It stores customers' identity and transaction data. It also sends responses of customers' requests to ***Subsystem 1***.

– ***Subsystem 4*** performs business analysis of customers' transaction data for bank employees, using anonymized customers' data in ***Subsystem 2***

The below steps shown the simulated values of the case study according to steps of our conceptual approach shown in background section.

Step 1) Based on system state dependencies of critical cloud infrastructure (identified from application domain knowledge), construct the MDP state graphs of all subsystems.

In Subsystem 1, the following 4 features are identified by the domain experts' initial state dependencies.

Feature 1: The rate of unsuccessful login [Values: high, low]

Feature 2: The firewall configuration [Values: good, bad]

Feature 3: The IDS state [Values: running, stopped]

Feature 4: Bank application process queue [Values: good condition, stopped]

<u>The possible values of the above features of subsystem 1 and their meaning are:</u>

Feature value a: The rate of unsuccessful login attempts is low

Feature value b: The rate of unsuccessful login attempts is high

Feature value c: The firewall configuration is good

Feature value d: The firewall configuration is bad

Feature value e: The IDS is running normally

Feature value f: The IDS is unable to process data or has been stopped

Feature value g: Bank application process queue is in good condition.

Feature value h: Bank application process queue has been stalled

The 4 features of MDP subsystem 1 and their 2 possible values for each feature results in a total of 2^4 states in MDP subsystem 1. The possible states and the feature values of each state are given below in table 5

In practical implementations the number of states is reduced in MDP using approximation techniques [34 - 37] to combine similar states specific to the critical cloud infrastructure.

State	Feature value 1	Feature value 2	Feature value 3	Feature value 4
State 1	A	c	e	g
State 2	A	c	e	h
State 3	A	c	f	g
State 4	A	c	f	h
State 5	A	d	e	g
State 6	A	d	e	h
State 7	A	d	f	g
State 8	A	d	f	h
State 9	B	c	e	g
State 10	B	c	e	h
State 11	B	c	f	g
State 12	B	c	f	h
State 13	B	d	e	g
State 14	B	d	e	h
State 15	B	d	F	g
State 16	B	d	F	h

Table 5: The States and their Features in MDP Subsystem 1 of the Case Study

The MDP actions of subsystem 1 are:

A0: users attempt to login to system

A1: Change in firewall configuration

A2: Change in IDS state

A3: Change in the size of requests

The action set containing the states and the actions possible from each state is shown below:

A = {[S1: A0, A1, A2, A3], [S2: A0, A1, A2, A3], [S3: A0, A1, A2, A3], [S4: A0, A1, A2, A3], [S5: A0, A1, A2, A3], [S6: A0, A1, A2, A3], [S7: A0, A1, A2, A3], [S8: A0, A1, A2, A3], [S9: A0, A1, A2, A3], [S10: A0, A1, A2, A3], [S11: A0, A1, A2, A3], [S12: A0, A1, A2, A3], [S13: A0, A1, A2, A3], [S14: A0, A1, A2, A3], [S15: A0, A1, A2, A3], [S16: A0, A1, A2, A3]}

State Diagram of MDP subsystem 1 is shown in the figure 9:

The state diagram consists of states of MDP subsystem 1 and the possible actions from each state shown in Figure 9. For better visibility self-loops have not been shown in the state diagram. They are shown in the transaction matrix below.

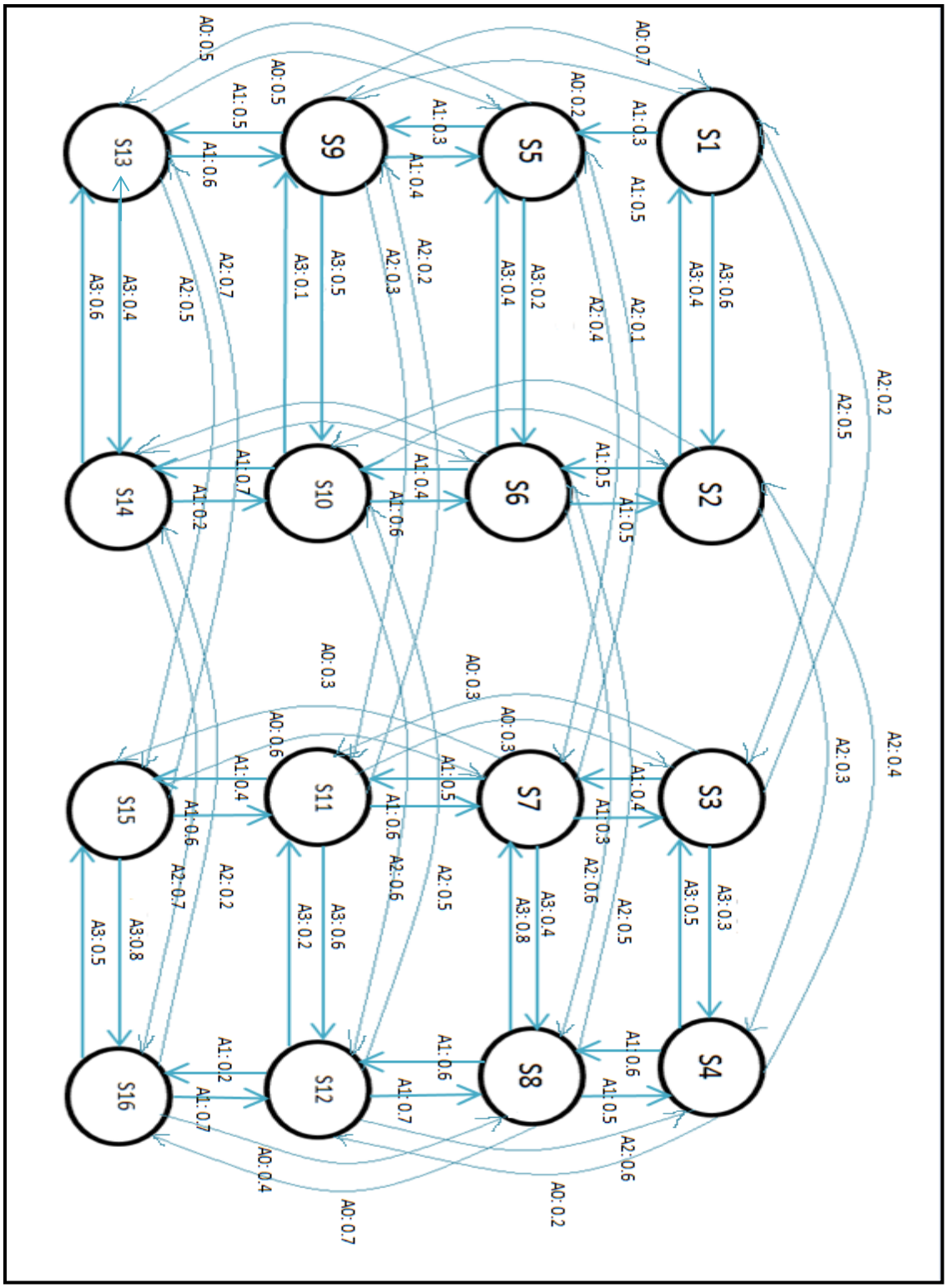

Figure 9: The State Diagram of MDP Subsystem 1 of the Case Study

State\ Action	A0	A1	A2	A3
S1	[s9:0.2][s1:0.8]	[s5:0.3][s1:0.7]	[s3:0.5][s1:0.5]	[s2:0.6][s1:0.4]
S2	[s10:0.3][s2:0.7]	[s6:0.5][s2:0.5]	[s4:0.3][s2:0.7]	[s4:0.3][s2:0.7]
S3	[s11:0.3][s3:0.7]	[s7:0.4][s3:0.6]	[s1:0.5][s3:0.5]	[s4:0.3][s3:0.7]
S4	[s12:0.2][s4:0.8]	[s8:0.6][s4:0.4]	[s2:0.3][s4:0.7]	[s3:0.5][s4:0.5]
S5	[s13:0.5][s5:0.5]	[s1:0.5][s5:0.5]	[s7:0.1][s5:0.9]	[s6:0.2][s5:0.8]
S6	[s14:0.7][s6:0.3]	[s10:0.4][s6:0.6]	[s8:0.3][s6:0.7]	[s5:0.4][s6:0.6]
S7	[s15:0.3][s7:0.7]	[s3:0.3][s7:0.7]	[s5:0.1][s7:0.9]	[s8:0.4][s7:0.6]
S8	[s7:0.4][s8:0.6]	[s6:0.6][s8:0.4]	[s6:0.5][s8:0.5]	[s7:0.8][s8:0.2]
S9	[s1:0.7][s9:0.3]	[s5:0.4][s9:0.6]	[s11:0.3][s9:0.7]	[s10:0.3][s9:0.7]
S10	[s2:0.8][s10:0.2]	[s14:0.7][s10:0.3]	[s12:0.6][s10:0.4]	[s9:0.1][s10:0.9]
S11	[s3:0.3][s11:0.7]	[s7:0.6][s11:0.4]	[s9:0.2][s11:0.8]	[s12:0.6][s11:0.4]
S12	[s4:0.6][s12:0.4]	[s16:0.2][s12:0.8]	[s10:0.5][s12:0.5]	[s11:0.2][s12:0.8]
S13	[s5:0.5][s13:0.5]	[s9:0.6][s13:0.4]	[s15:0.5][s13:0.5]	[s14:0.4][s13:0.6]
S14	[s6:0.6][s14:0.4]	[s10:0.2][s14:0.8]	[s16:0.2][s14:0.8]	[s3:0.6][s14:0.4]
S15	[s7:0.6][s15:0.4]	[s11:0.6][s15:0.4]	[s13:0.7][s15:0.3]	[s6:0.8][s15:0.2]
S16	[s8:0.4][s16:0.6]	[s12:0.2][s16:0.8]	[s14:0.2][s16:0.8]	[s15:0.5][s16:0.5]

Table 6: The Transaction Matrix of MDP Subsystem 1 of the Case Study

The transaction matrix shows the resultant state when an action is performed from a particular state.

The reward attained when you reach a certain state is shown by the reward matrix. The high rewards here represent the vulnerable states. The threshold for a reward state being a vulnerable state is provided by the domain expert, here it is set to +15.

R = {S1:-0.5, S2:-0.5, S3:-0.5, S4:-0.5, S5:-0.5, S6:-0.5, S7:-0.5, S8:-0.5, S9:-0.5, S10:-0.5, S11:-0.5, S12:-0.5, S13:+10, S14:+75, S15:+55, S16:+100}

The vulnerable states in MDP subsystem 1 are S14, S15 and S16. The terminal states defined by the domain expert are S14 and S16. The terminal states are highly vulnerable states in the MDP subsystem.

Similarly subsystem 2, subsystem 3 and subsystem 4 MDPs are constructed in Step 1) of our conceptual approach using the initial state dependencies provided by the domain expert.

Using the information from the MDPs vulnerable states and the domain experts the Bayesian network is generated, validate the Bayesian network with MDPs as shown in Chapter 4, then also use the GCA shown in chapter 5 to check the conformance of the generated Bayesian network. The initial Bayesian network is generated after executing these two steps. The domain expert sets the security breach threshold to 0.6.

Step 2) Monitor the critical cloud infrastructure of the bank to observe its operational behaviors, including human behaviors. Update the MDP and BN state graphs based on causal relationships among the observed operational behaviors using Bayesian

probability estimation algorithm (BPEA) and MDP probability estimation algorithm (MPEA).

Step 3) Check the conformance of the BN structure with Bayesian properties using the approach shown in Chapter 5. Also check the conformance of the MDP state graph structure with Markovian properties and to each other as shown in chapter 4. Update BN and MDP state graphs.

Step 4) Estimate accuracy of probabilities in MDP and BN state graphs by deploying the MDPs and BN of critical cloud infrastructure "passively", and then insert known security vulnerabilities in the infrastructure.

- Repeat ***Steps 2)*** to ***4)*** until the probabilities in the state graphs of MDPs and BN are tuned to a point where the accuracy of the prediction of system breaches is deemed "good enough" by the system administrator to be deployed in real-time.

The System-wide Bayesian network deemed good enough by the system administrator is shown below:

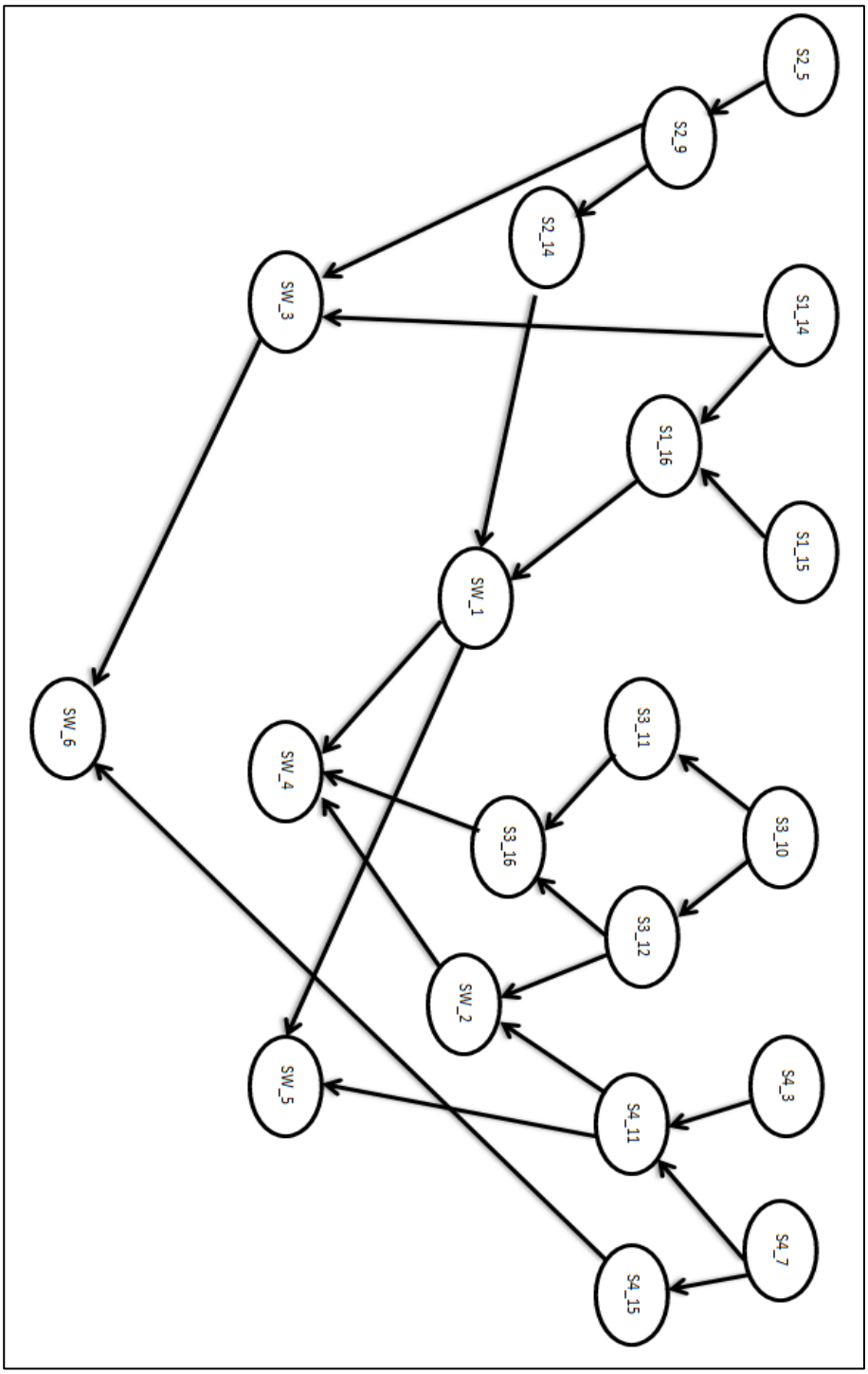

Figure 10: The System-wide Bayesian Network at the end of Step 4) of Conceptual Approach of Case Study

Node	conditional probability distribution (CPD)
S1_14	[0.9, 0.1]
S1_15	[0.9, 0.1]
S1_16	[1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]
S2_5	[0.9, 0.1]
S2_9	[0.8, 0.7, 0.3, 0.2]
S2_14	[0.9, 0.2, 0.8, 0.1]
S3_10	[0.8, 0.2]
S3_11	[0.8, 0.3, 0.7, 0.2]
S3_12	[0.8, 0.3, 0.7, 0.2]
S3_16	[0.1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]
S4_3	[0.9, 0.1]
S4_7	[0.9, 0.1]
S4_11	[0.1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]
S4_15	[0.9, 0.8, 0.2, 0.1]
SW_1	[0.1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]
SW_2	[0.1, 0.8, 0.8, 0.1, 0.05, 0.2, 0.2, 0.9]
SW_3	[0.1, 0.8, 0.8, 0.1, 0.05, 0.2, 0.2, 0.9]
SW_4	[0.1, 0.9, 0.9, 0.7, 0.9, 0.7, 0.7, 0.7, 0.05, 0.1, 0.1, 0.3, 0.1, 0.3, 0.3, 0.9]
SW_5	[0.1, 0.8, 0.8, 0.1, 0.05, 0.2, 0.2, 0.9]
SW_6	[0.1, 0.8, 0.8, 0.95, 0.05, 0.2, 0.2, 0.9]

Table 7: The Conditional Probability Distribution of the Bayesian Network of the Case Study shown in Figure 10

Here S1_14 to S1 _15 are vulnerable states in the MDP subsystems similar to data shown in table 5. These are observed nodes in the Bayesian network.

The states SW_1 to SW_6 are unobserved nodes who represent system-wide vulnerabilities involving multiple MPD subsystem such as DDoS attack as a deviation to conduct a malicious transaction involving multiple subsystems, configuration change by an internal employee (operational activity or malicious activity) leading to remote file execution etc. The probabilities of the unobserved nodes are inferred by the BN inference engine.

Step 5) To predict security breaches at subsystem level and system-wide level, first the policy iteration [37] is performed on each MDP subsystem at a certain time with the MDP agent starting at the current MDP State of critical cloud infrastructure. The MDP agent represents the action to be taken at each state in critical cloud infrastructure during policy iteration. The result of the policy iteration on a particular MDP subsystem is the predicted path the attacker takes from one MDP State to the next MDP State in each time step the attacker takes to reach his goal in the MDP subsystem. The attacker reaches his goal when the MDP agent reaches a terminal MDP state. A terminal MDP state is a highly vulnerable MDP state defined by the domain expert. For MDP subsystem 1, the terminal MDP states are S14 and S16.

Consider the critical cloud infrastructure is currently in state S7 and we begin policy iteration on each MDP subsystem. The path taken by the attacker in MDP subsystem 1 is predicted to be: S7 -> S11-> S15-> S16. This is calculated by the policy iteration [38] which maximizes the reward received according to the state diagram of the MDP subsystem 1. Here the vulnerable states activated in the Bayesian network are S15 and S16. Policy iteration gives the best action to take for each State. We use this information to go to the next state.

Similarly, the policy iteration on MDP subsystem 2, MDP subsystem 3 and MDP subsystem 4 is run to find the vulnerable states reached by the attacker in each subsystem. Table 8 shows the vulnerable states reached (Boolean value True in BN) in each MDP subsystem.

Subsystem	Vulnerable States reached
1	S15, S16
2	S5, S9
3	S10, S11, S16
4	S4, S15

Table 8: The Vulnerable States Reached in Each MDP Subsystem of the Case Study

Based on the vulnerable states reached by the attacker in each MDP subsystem the Bayesian network inference engine is run to detect system-wide security breaches. The probability of a predicted security breach exceeds the specified threshold 0.6 set in ***Step 1)*** for the security breach, the security breach is predicted as to occur soon. Figure 11 shows the system-wide security breach prediction on each node of the Bayesian network.

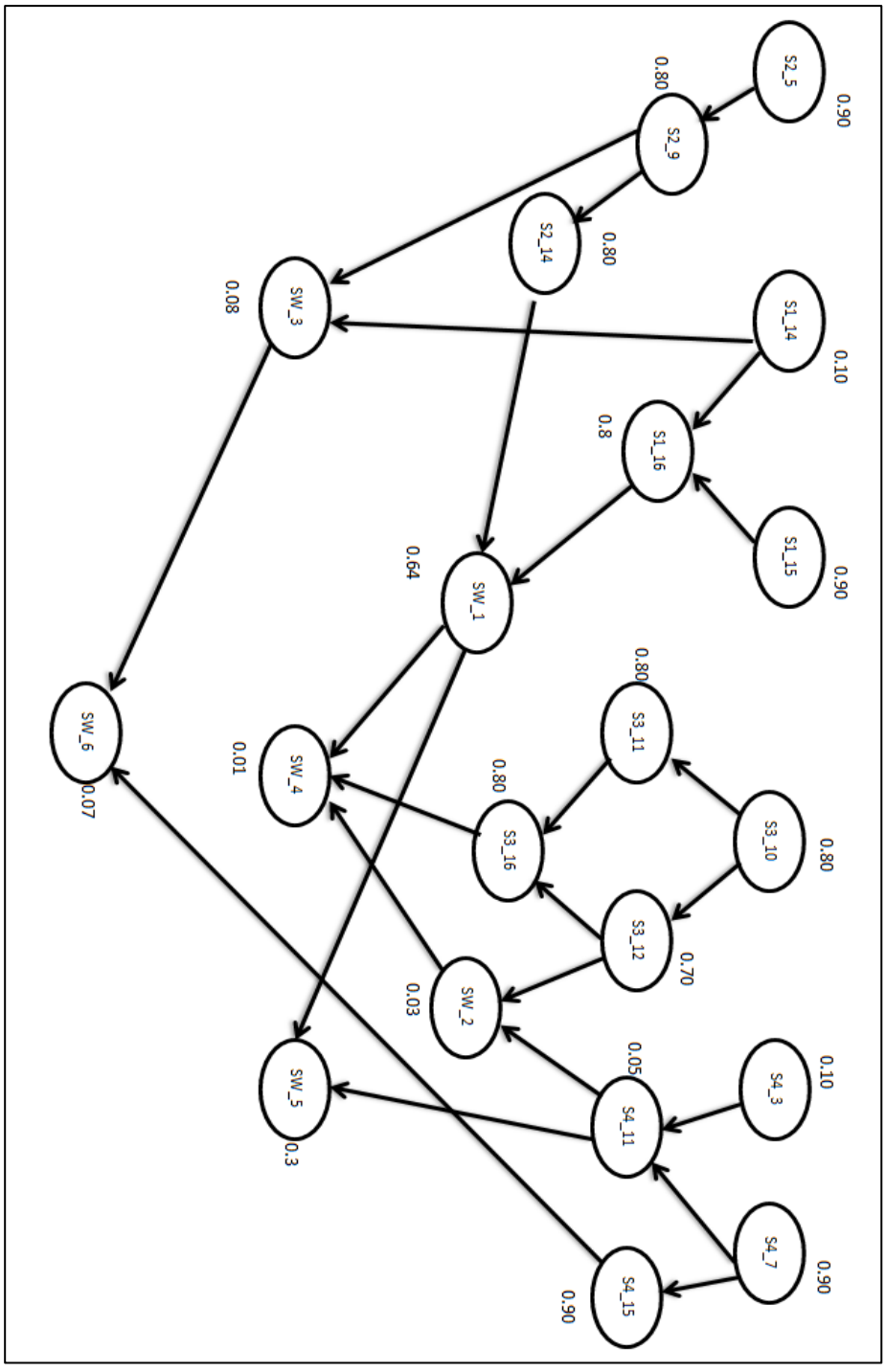

Figure 11 The Predicted System-Wide Security Breach at end of Step 5) of Conceptual Approach of Case Study

The system-wide security breach prediction alerts that System-wide vulnerability 1 has a 0.60 probability of occurring soon.

Similarly, rewards, transaction matrix and agent start location were changed and vulnerabilities were introduced in the critical cloud infrastructure of bank to predict security breaches. This simulation consists of 20 to 30 states in each subsystem and 20 – 25 nodes in the system-wide Bayesian network. The two techniques shown in Chapter 4 and 5 were implemented in each of the MDP subsystems and the system-wide BN.

Chapter 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

In this thesis, two techniques have been presented to support the conceptual approach [3] to predicting security breaches in critical cloud infrastructures. The first technique is for evaluation of the conformance of the Bayesian network with the multiple MDPs. The second technique is to evaluate the dynamically changing Bayesian network structure for conformance with the rules of the Bayesian network using a graph checker algorithm. The simulation of part of the conceptual approach presents the prediction results of the conceptual approach using the above two techniques. The conceptual approach can be applied in other applications where modelling human behaviors in large and complex systems are required. The challenges shown in Chapter 3 are addressed by:

- Incorporation of probabilistic human behaviors in ***Step 2)*** of the conceptual approach, and involvement of the domain expert in ***Step 1)*** (off-line) will help the conceptual approach address challenge ***C1)*** User-centric Security Systems
- Predictive nature of the conceptual approach will enable selectively applying certain security measures based on predicted breaches, and hence the operational overhead of using the approach may not be more than that of existing approaches without predictive capability. This will help the conceptual approach address challenge ***C3)*** Overhead for Security Measures

- Challenge ***C2)*** Emergence of Software Defined Network (SDN) is not addressed in the conceptual approach, but it can be reduced by incorporating fail safe mechanisms, such as controller hardening, and robust policy framework.

7.2 Future Research

The following research tasks need to be completed for deployment of our conceptual approach to predicting security breaches in critical cloud infrastructures.

- Develop an effective data specification language for domain experts to provide their input in our conceptual approach.
- Generate the probability metrics for the causal relationship among security breaches in Bayesian network, and state dependencies among the states in the MDP.
- Develop effective techniques to construct, check and update the state graphs of the MDP using an efficient MDP graph checker algorithm.
- Develop an efficient way for frequent updating the probabilities and the state graph structures of the MDP and Bayesian network.
- Adding global variables to the Bayesian network such as threat intelligence and external factors outside the critical cloud infrastructure to improve prediction accuracy needs to be researched.

REFERENCES

1. Sajjan Shiva, Sankardas Roy, and Dipankar Dasgupta. "Game theory for cyber security." Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research, p. 34. ACM, 2010.

2. Peng Liu, and Lunquan Li. "A Game Theoretic Approach to Attack Prediction." Technical Report, PSU-S2-2002-001, Penn State Cyber Security Group, 2002.

3. Stephen S. Yau, Arun Balaji Buduru, and Vinjith Nagaraja "Protecting Critical Cloud Infrastructures with Predictive Capability" Proceedings 15th annual IEEE Cloud 2015

4. Sankardas Roy, Charles Ellis, Shiva Sajjan, Dipankar Dasgupta, Vivek Shandilya, and Qishi Wu. "A survey of game theory as applied to network security." Proceedings of System Sciences (HICSS), 2010 43rd Hawaii International Conference on, pp. 1-10. IEEE Press2010

5. Milind Tambe, Manish Jain, James Adam Pita, and Albert Xin Jiang. "Game theory for security: Key algorithmic principles, deployed systems, lessons learned." Proceedings of Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on, pp. 1822-1829. IEEE Press, 2012.

6. Mohammad Hossein Manshaei, Quanyan Zhu, Tansu Alpcan, Tamer Bacşar, and Jean-Pierre Hubaux. "Game theory meets network security and privacy." ACM Computing Surveys (CSUR) 45, no. 3 (2013): 25.

7. Alecsandru PĂTRAŞCU, and Emil Simion. "Game theory in cyber security defence." Proceedings of Electronics, Computers and Artificial Intelligence (ECAI), 2013 International Conference on, pp. 1-6. IEEE Press, 2013.

8. Dan Shen, Genshe Chen, Jose B. Cruz Jr, Leonard Haynes, Martin Kruger, and Erik Blasch. "A markov game theoretic data fusion approach for cyber situational awareness." Proceedings Defense and Security Symposium, pp. 65710F-65710F. International Society for Optics and Photonics, 2007.

9. Dan Shen, Genshe Chen, Leonard Haynes, and Erik Blasch. "Strategies comparison for game theoretic cyber situational awareness and impact assessment." Proceedings of Information Fusion, 2007 10th International Conference on, pp. 1-8. IEEE Press, 2007.

10. Wei Jiang, Zhi-hong Tian, Hong-li Zhang, and Xin-fang Song. "A stochastic game theoretic approach to attack prediction and optimal active defense strategy decision." Proceedings of 2008 IEEE International Conference on, Networking, Sensing and Control, pp. 648-653. 2008.

11. Holm, Hannes, Khurram Shahzad, Markus Buschle, and Mathias Ekstedt. "P< sup> 2</sup> CySeMoL: Predictive, Probabilistic Cyber Security Modeling Language." (2014).

12. Ullah K.W., Ahmed A.S., Ylitalo J., "Towards Building an Automated Security Compliance Tool for the Cloud", Proceedings of Trust, Security and Privacy in 2013 12th IEEE International Conference on Computing and Communications (TrustCom), , pp: 1587- 1593

13. Teodor Sommestad, "A framework and theory for cyber security assessments." PhD diss., KungligaTekniskahögskolan (KTH), Royal Institute of Technology Stockholm, Sweden, 2012.

14. Teodor Sommestad, Mathias Ekstedt, and Lars Nordström, "A case study applying the Cyber Security Modeling Language.", Proceedings of CIGRE 2010.

15. Holm Hannes, Khurram Shahzad, Markus Buschle, and Mathias Ekstedt, "CySeMoL: Predictive, Probabilistic Cyber Security Modeling Language.", IEEE Transactions onDependable and Secure Computing,, vol. 99, 2014, pp.1,1.

16. Wenyuan Li, and P. Choudhury. "Probabilistic planning of transmission systems: Why, how and an actual example.", Proceedings of 21st IEEE Power and Energy Society General Meeting-Conversion and Delivery of Electrical Energy the 21st Century, 2008, pp. 1-8.

17. Jeff Hughes, and George Cybenko, "Three tenets for secure cyber-physical system design and assessment.", Proceedings of Society of Photo-Optical Instrumentation Engineers Defense Security, June, 2014

18. Claudine Conrado and Patrick de Oude, "Scenario-based reasoning and probabilistic models for decision support.", Proceedings of 17th IEEE Information Fusion (FUSION), July, 2014, pp. 1-9.

19. W.C. Moody, Hongxin Hu, A. Apon, "Defensive maneuver cyber platform modeling with Stochastic Petri Nets,", Proceedings of 10th International Conference on Collaborative Computing (Collaborate Com 2014), October 22-25, 2014.

20. Hulten, Geoff, David Maxwell Chickering, and David Heckerman. "Learning Bayesian networks from dependency networks: A preliminary study." In Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics. 2003

21. Boutilier, Craig, Richard Dearden, and Moisés Goldszmidt. "Stochastic dynamic programming with factored representations." Artificial Intelligence 121, no. 1 (2000): 49-107.

22. Charbit, Pierre, Stéphan Thomassé, and Anders Yeo. "The minimum feedback arc set problem is NP-hard for tournaments." Combinatorics, Probability and Computing 16, no. 01 (2007): 1-4.

23. Biggest ever cyber-attacks by Telegraph, "http://www.telegraph.co.uk/technology/internet-security/10848707/The-biggest-ever-cyber-attacks-and-security-breaches.html", Accessed on February 15th, 2015

24. Akhil Behl, "Emerging security challenges in cloud computing: An insight to cloud security challenges and their mitigation", Proceedings of IEEE World Congress on Information and Communication Technologies (WICT), 2011, pp: 217 – 222

25. R.A. Howard, "Dynamic Programming and Markov Processes", MIT Press, Cambridge, MA, 1960

26. R.E. Bellman, "Dynamic Programming", Princeton University Press, Princeton, NJ, 1957

27. C. Boutilier, T. Dean, S. Hanks, "Decision theoretic planning: Structural assumptions and computational leverage", J. Artificial Intelligence Res. 11 (1999) 1–94.

28. C. Boutilier and M.L. Puterman, "Process-oriented planning and average-reward optimality" Proceedings of IJCAI-95, Montreal, Quebec, 1995, pp. 1096–1103.

29. T. Dean, L.P. Kaelbling, J. Kirman and A. Nicholson, "Planning with deadlines in stochastic domains" Proceedings of AAAI-93, Washington, DC, 1993, pp. 574–579

30. J.A. Feldman and R.F. Sproull, "Decision theory and artificial intelligence II: The hungry monkey", Cognitive Science. 1 (1977) 158–192.

31. A. Darwiche and M. Goldszmidt, "Action networks: A framework for reasoning about actions and change under uncertainty" Proceedings of 10th Conference on Uncertainty in Artificial Intelligence, Seattle, WA, 1994, pp. 136– 144.

32. T. Dean and K. Kanazawa, "A model for reasoning about persistence and causation", Computational Intelligence 5 (3) (1989) 142–150.

33. J. Pearl, "Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference", Morgan Kaufmann, San Mateo, CA, 1988.

34. Azzedine Benameur, Nathan S. Evans and Matthew C. Elder, “Cloud resiliency and security via diversified replica execution and monitoring”, Proceedings of 2013 6th International Symposium on Resilient Control Systems (ISRCS), 2013 , pp: 150- 155

35. Roy, Nicholas, Joelle Pineau, and Sebastian Thrun. "Spoken dialogue management using probabilistic reasoning." Proceedings of 38th Annual Meeting on Association for Computational Linguistics, pp. 93-100. Association for Computational Linguistics, 2000.

36. Gordon, Geoffrey J. "Stable function approximation in dynamic programming." Proceedings of twelfth international conference on machine learning, pp. 261-268. 1995.

37. Fern, Alan, Sung Wook Yoon, and Robert Givan. "Approximate Policy Iteration with a Policy Language Bias." Proceedings of NIPS. 2003.

38. Meyn, Sean P. "The policy iteration algorithm for average reward Markov decision processes with general state space." Automatic Control, IEEE Transactions on 42, no. 12 (1997): 1663-1680.