International Journal of Innovative
Computing, Information and Control
Volume **12**, Number **2**, April **2016**

ICIC International ©2016 ISSN 1349-4198
pp. 569–580

ANOMALY NETWORK INTRUSION DETECTION USING HIDDEN MARKOV MODEL

Chia-Mei Chen[1], Dah-Jyh Guan[2], Yu-Zhi Huang[2] and Ya-Hui Ou[1]

[1]Department of Information Management
[2]Department of Computer Science and Engineering
National Sun Yat-sen University
No. 70, Lienhai Rd., Kaohsiung 80424, Taiwan
cchen@mail.nsysu.edu.tw

Received August 2015; revised December 2015

Abstract. *Cyberattacks become more sophisticated than before, as they involve intelligent planning with respect to the target machine. The current defense products might not be able to correlate diverse sensor input. For example, a client with low security awareness is in the distributed network environment where the target resides might be compromised and unnoticed, which in turn is used as a stepping stone to intrude the target. The conventional signature-based intrusion detection systems might not be able to identify such planned attacks. A state-based classification model is suitable for detecting the attacks composed of a sequence of attack stages. This study defines a sequence of attack states corresponding to the attack stages and the proposed detection system adopts a stated-based classification model, Hidden Markov Model, for detecting such advanced planned attacks. The experimental results show that the proposed detection system can identify the attacks efficiently.*
Keywords: Distributed computing, Hidden Markov Model, Intrusion detection

1. **Introduction.** Nowadays computing environments involve distributed computing services. The interconnecting relationship with multiple machines and multiple platforms in the distributed computing environments complicates the security control and brings up potential security threats. An administrator is overwhelmed by the vast amounts of logs from different sensors and might not be able to connect the event of password guessing attack to the event of database query failure, if these are not presented together. Current security systems can collect event logs but contain many false positives and are lack of an efficient correlation algorithm to make links among the events of different sensors.

Advanced cyberattacks often involve a sequence of suspicious activities, where the activities are reported as separate events and distributed across various data sources. Under such advanced attacks described above, the administrator of a distributed computing network might not be able to identify the anomalies due to the massive overload of data. Furthermore, conventional signature-based IDS fail to correlate event data observed from various systems.

Hidden Markov Models (HMM) have been applied to anomaly detection since 1996 [1]. The previous researches [2,3] applying HMM were limited to small data sets or sensitive to the data errors. The contributions of this paper are: (1) proposing a correlation algorithm which efficiently processes big amount of event logs and identifies the temporal relation of anomalous events; (2) proposing a state-based detection model for identifying multi-stage advanced attacks; and (3) the experimental results show that the proposed detection system performs efficiently on a large amount of network event logs.

The rest of the paper is organized as follows. Section 2 briefly reviews the related research on network intrusion detection followed by an introduction of the Hidden Markov Model and its related researches on anomaly detection. Section 3 describes the proposed detection based on HMM. The experimental results are presented in Section 4 and the concluding remarks and future studies are stated in the last section.

2. Related Work.

Intrusion detection is vital in a distributed computing environment, as the large amounts of computing and storage resources attract attackers. Intrusion detection is a defense mechanism for detecting suspicious activities and protecting the perimeter from attacks. Intrusion detection and alert correlation will be studied in the beginning of this section. The theory of the Hidden Markov Model (HMM) will be introduced, followed by the detection researches based on HMM.

Lo et al. [4] proposed a framework for detecting distributed denial-of-service attacks by exchanging alert information with other intrusion detection systems. Zargar et al. [5] proposed an intrusion detection framework for a distributed computing environment where the service providers collaborate together to cope with attacks. A comprehensive trust management scheme is required to support the trust relationship among the service providers. Kumar et al. [6] proposed a clustering approach based on the Hidden Markov Model, since the distributed computing environment generates a large volume of security related data.

Liu et al. [7] proposed an alert correlation model for plotting attack scenarios. The work relies on the given attack graphs and signature rules to correlate security events, and the correlation method applies inductive and abductive reasoning. As IDS produce a large amount of alerts with many false positives, Raftopoulos and Dimitropoulos [8] proposed a correlation method which reduces the alerts and resulted in 15% false positives. The alerts used in this literature are pre-classified by the Snort classification rules and the work applies entropy-based information theoretic criterion to finding the recurring alerts. The evaluation demonstrated that the detection method performed better than the extant botnet detection. Siraj et al. [9] proposed a framework of intrusion alert prediction, which includes the following components: alert normalization, reduction, prioritization and attack scenario construction and prediction. The ensemble detection proposed by Amini et al. [10] combines different classifiers to obtain better detection results, including neural network, fuzzy clustering, and stacking combination method. The experimental results showed that the proposed ensemble approach performed better than the single classifiers.

2.1. Hidden Markov Model.

A Hidden Markov Model (HMM) is a doubly stochastic process with an underlying stochastic process which is not observable and can be examined through another set of stochastic processes [11]. A state of Markov model is directly visible, while that of HMM has a probability distribution over a set of outputs (observations). Therefore, a sequence of observations generated by HMM does not directly indicate the sequence of states.

A Hidden Markov Model is denoted as $\lambda = [A, B, \pi]$, where A is the state transition matrix, B is the observation probability matrix, and π is the HMM initial state probabilities. Three basic problems need to be solved:

(1) Given a set of observations $O = \{O_1, O_2, \cdots, O_T\}$ and the HMM $\lambda = [A, B, \pi]$, the probability of the given observation sequence $Pr(O|\lambda)$ is computed.

(2) Given a set of observations $O = \{O_1, O_2, \cdots, O_T\}$ and the HMM $\lambda = [A, B, \pi]$, an optimal state sequence $I = \{i_1, i_2, \cdots, i_T\}$ is computed.

(3) Given a set of observations $O = \{O_1, O_2, \cdots, O_T\}$, the parameters of the HMM model $\lambda = [A, B, \pi]$ are adjusted such that $Pr(O|\lambda)$ is maximized.

Problem (1) can be solved by either the forward method or the backward method [12]; the Viterbi algorithm [12,13] finds the answer of problem (2); the Baum-Welch algorithm (BW) solves the last one [13]. The approaches to apply the above algorithms are explained below.

In order to estimate the parameters of HMM, problem (1) should be solved first by calculating the probability value of an observation sequence. After the parameters are estimated, the forward or backward method can be applied to training the model. The forward variable $\alpha_t(i) = P(O, q_t = s_i)$ denotes the probability of the partial observation sequence that q_t is the current state produced when state s_i are at time t, given the model λ. Once the forward variable $\alpha_t(i) = \pi_t b_t(o_t)$ is initialized, an optimal model is obtained by applying the induction formula, $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) \times a_{ij}\right] \times b_j(o_{t+1})$, until $Pr(O|\lambda) = \sum_{i=1}^{N} \alpha_t(i)$ converges and is maximized, where $a_{ij} = P(q_{t+1} = s_j | q_t = s_i)$, $i, j = 1, 2, \cdots, N$ denotes the probability of state s_i at time t moving to s_j at time $t+1$ and $b_j(o_{t+1})$ is the probability of the observable state at time $t+1$ given the hidden state at j.

The next problem is to find the most likely sequence of the hidden states, given the HMM model and the observation sequence $O = \{O_1, O_2, \cdots, O_T\}$. The Viterbi algorithm is a dynamic programming algorithm for finding the most likely sequence of the states, i.e., the Viterbi path results from the given sequence of the observations. Initially, $\delta_1(i) = \pi_i b_i(o_1)$ and the induction formula is $\delta_t(j) = \max_{1 \le i \le N} \delta_{t-1}(i) a_{ij} b_j(o_t)$ for $2 \le t \le T$ and $1 \le j \le N$, where $\delta_t(j)$ is the probability of the most likelihood state sequence of the first t observations and j as its final state. By applying dynamic programming algorithm, the Viterbi algorithm finds the most likely hidden state at time T which is $q_t^* = \arg \max_{1 \le i \le N} \delta_T(i)$ for a given observation sequence, when $P_t^* = \max_{1 \le i \le N} \delta_T(i)$ is maximized.

The Baum-Welch algorithm optimizes the HMM model. It re-estimates the parameters of HMM $\lambda' = [A', B', \pi']$ as $\pi_t' = \gamma_1(i)$, $a_{ij}' = \frac{\sum_{t=1}^{T} \xi_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)}$ and $b_j'(k) = \frac{\sum_{t=1 \ s.t. o_t = k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$ recursively until the model converges when $Pr(O|\lambda)$ is maximized.

HMM is known for its application in temporal pattern recognition such as speech recognition [14] and bioinformatics. Forrest et al. [1] proposed the first research that applied HMM to identifying abnormal system call sequences. Many studies were conducted based on HMM which found anomalies in system call sequences. Hoang and Hu [15] presented an efficient training scheme for intrusion detection based on system calls. The HMM training scheme divides the observations into subsets and integrates the sub HMM incrementally into the final one. The results showed that the training time improved significantly.

2.2. **Detection based on Hidden Markov Model.** Ourston et al. [2] applied HMM to detecting coordinated network attacks and the results showed that the HMM approach has better performance than the decision tree and neural network approaches. This study was evaluated by a small set of one-day event logs. The advanced attacks nowadays may last for a longer duration and the past research might not be able to identify such attacks. Ye et al. [3] discovered that HMM has a high detection rate in low error data but is sensitive to noise. To reduce the effect of the sensitivity of noisy data, this study correlates the related logs by extracting the relevant attack events and an attack plan is characterized by a sequence of attacks in a time series. According to the previous studies and our analysis, HMM is suitable for identifying such state-based attacks.

3. **Proposed Approach.** Based on our preliminary study and the security reports [16], the modern attacks often involve a sequence of attack stages with an attack plan composed of the following three stages: (1) Reconnaissance (attack state R): Machine X is under low frequency scans from various sources; (2) Attack (attack state A): Once machine X's vulnerability is discovered, the machine is exploited; (3) Stepping stone (attack state S): The compromised machine X becomes a stepping stone and starts attacking others.

The proposed detection system applies a state-aware classification approach for identifying the attack stages of the attacks. As the attack stages (states) are hidden by the event logs, Hidden Markov Model (HMM) is adopted, in which the sequence of attack state transitions is a hidden process and is observed through a sequence of emitted observations. As illustrated in Figure 1, the activities observed from event logs are emitted observations and the sequence of the hidden states becomes a sequence of attack steps shown at the upper layer, where the observations are shown in the lower layer.

The proposed system architecture is plotted in Figure 2. As the logs from different sources have different formats, the preprocess module collects and normalizes them into a uniform format. Module Feature Extraction extracts features from the collected logs. Module Event Correlation aggregates and correlates the related events using the extracted features.

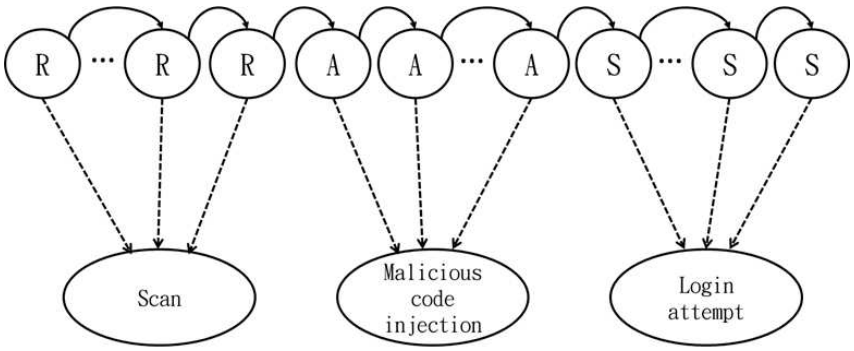

FIGURE 1. Illustration of the attack

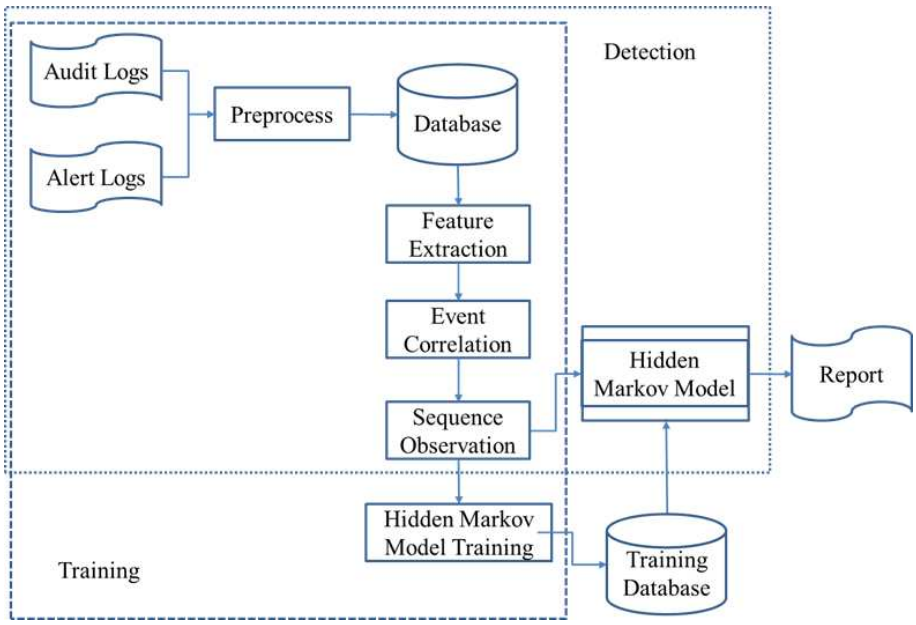

FIGURE 2. System architecture

The survey [16] indicated that the complexity and training time of HMM may depend on the data size and the number of the states, and Ye et al. [3] pointed out that HMM is sensitive to error data. Therefore, to reduce the number of the states, the Event Correlation module aggregates the same attacks within a given time frame into one event with weight and hit count. To reduce error data, it correlates the attack events targeted at the same host in a time sequence. Once a temporal sequence of observations is aggregated and correlated, it will be examined by the HMM-based classification model. It should be noted that training the HMM involves a similar process by feeding the training data.

As the duration of a stealthy attack might be diversified, the Event Correlation module applies the adaptive sliding window approach to performing temporal correlation. It (1) aggregates the events of the same attack strategy from one log, (2) correlates with the events of different strategies from multiple logs, and (3) identifies possible observable actions targeting at the specific machine in a temporal sequence. An adaptive dynamic length sliding window is used to accumulate events related to the current stage in which a target machine is identified in the log. The Event Correlation algorithm is shown in Table 1.

The links between the attack stages may rely on the events of a target machine from different data sources. For example, an internal machine with weak password is compromised and becomes a stepping stone attacking a target. Therefore, the destination IP address of a password guessing attack (a victim of password guessing attack) might appear at the source IP address of an event of the target (the attacking side). Based on the attack states shown in Figure 1, the event time of one state occurs before that of the next state. Therefore, the time dependency could further reduce the possible correlation events and improve the performance. The proposed correlation module links anomalous events from different data sources by the relationship of the IP addresses and the temporal dependency as illustrated above.

The proposed HMM based classification model illustrated in Figure 3 consists of three layers: the hidden states at the first layer, the observable events emitted from the hidden

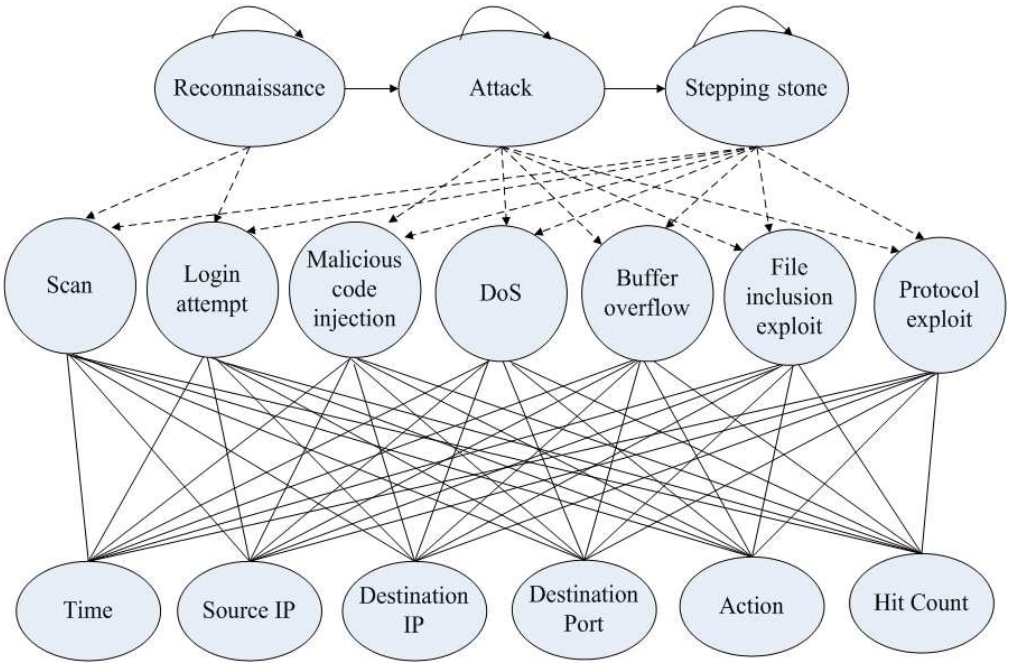

FIGURE 3. The proposed Hidden Markov Model

Table 1. Event Correlation algorithm

```
EventCorrelation( )
{
    Input: a set of logs from devices, {L_1, L_2, ··· , L_h};
    Output: a sequence of aggregated and correlated attack events;

    // Variable definition:
    // G_i(p) is a set of aggregated attacks on IP_p at time frame t_i.
    // t_i < t_{i+1} < t_{i+2}.
    // Aggregation is performed within a time frame.
    // Correlation is done in a sequence of time frames, t_i, t_{i+1}, and t_{i+2}.

    Initialize G_i(p) for all i and IP_p;
    For each log L_i
    {
        For each IP_p in log L_i
        {
            Aggregate the same attack events for Destination IP_p at time frame t_i to G_i(p);
            If the event type of G_i(p) ∈ the category of Reconnaissance
            {
                // Aggregate all the same types of attack events into G_i(p).
                Continue to group and unite log records of reconnaissance attacks
                for Destination IP_p into G_i(p);
                Update HitCount value accordingly;

                // Set the next time frame t_{i+1} to the timestamp of the last event log
                    from G_i(p), indicating the start of the next stage.
                Let start time of time frame t_{i+1} be the latest timestamp of group G_i(p);

                // Aggregate all attack events of the category of Attack.
                Collect the attack events intended for p and belonged to the category
                of Attack into group G_{i+1}(p);
                Update HitCount value accordingly;

                // Set the next time frame t_{i+2} to the last timestamp of G_{i+1}(p),
                    indicating the start of the next stage.
                Let time t_{i+2} be the latest timestamp of group G_{i+1}(p);

                // Aggregate all attack events of the category of Stepping Stone.
                Collect the attack events initiated from p and belonged to the category
                of Stepping Stone into group G_{i+2}(p);
                Update HitCount value accordingly;

            } // end if the first reconnaissance attack event found for IP_p
        } // end for each IP in log L_i
    } // end for each log
    // Complete correlation of all logs

    // output
    For all IPs found on the logs
    {
        Output the aggregated and correlated attack events in the following order:
        G_i(p), G_{i+1}(p), and G_{i+2}(p);
    }
}
```

states at the second layer, and the feature set used for correlation at the bottom layer. Different attack events might refer to different features.

4. **Performance Evaluation.** The performance evaluation consisted of two phases: (I) validating the proposed detection model by a controlled environment and comparing with an existing detection system, and (II) evaluating the detection performance using the event logs of a real network. Experiment I was conducted in a controlled and surveillant environment. For Experiment II, the five-week event logs were extracted from a real network with the average of four million web requests and over ten thousand IDS alerts per day.

In Experiment I, the controlled network environment consisted of three parts: attack sites, a controlled network accessible through some access mechanisms and a basic defense mechanism using netflow and syslog. The attacks were injected to the network under surveillance. The total of 48 thousand log records was collected including network flow logs and system logs.

The detection measurements, TP, TN, FP, and FN, are defined as follows, where TP is the number of true positive decisions, and TN, FP, and FN refer to the number of true negative, false positive and false negative decisions, respectively. The measurements are summarized in Table 2.

Table 2. Detection measurements

Actual \ Detected	Benign	Attacks
Benign	True Negatives (TN)	False Positives (FP)
Attacks	False Negatives (FN)	False Positives (FP)

4.1. **Experiment I: controlled network.** To validate the proposed model, attacks were injected to the network. As an attacker might attempt to intrude a target host stealthily to evade detection, Experiment I injected stealth attacks over a long period of time. The injected attacks imitated the real attack patterns found in real networks.

For illustration, a sequence of injected attacks is explained below. The logs related to the multi-stage attack are shown in Table 3. First, scanning attacks from multiple source IP addresses to the simulated network were injected covertly on an hourly basis. Once a vulnerability of the victim (*.*.241.171) was discovered, the low frequency login attempts BFA (Brute Force Attacks) from different source IP addresses were sent to the victim. Finally, one successful login attempt, Brute Force Attack Success (BFAS) was performed on the victim.

In the controlled environment, stepping stone attacks were terminated to prevent real damage. The detection report shown in Figure 4 illustrates the state sequence given the observation sequence from Table 3, where state 1 represents the Reconnaissance state, state 2 denotes the Attack state, and state 3 represents the Stepping stone state. The detection performance in Experiment I shown in Table 4 indicates that the proposed detection can identify true stealth attack sequences efficiently with a precision rate of 93.2%.

The detection results of the existing system are shown in Table 5. The existing defense is lack of efficient correlation. The existing defense mechanism failed to alert stealthy login attempts, while the proposed system could correlate suspicious events and identify the multi-stage attacks.

TABLE 3. A sequence of injected stealth attacks

Obs. #	Time	Source IP	Destination IP	Action
1	4/14 03:00:00	*.*.21.186	*.*.0.0	Scan
2	4/14 04:00:00	*.*.21.186	*.*.0.0	Scan
3	4/14 04:00:00	*.*.162.69	*.*.0.0	Scan
4	4/14 04:00:00	*.*.162.69	*.*.0.0	Scan
5	4/14 05:00:00	*.*.162.69	*.*.0.0	Scan
6	4/14 05:00:00	*.*.190.102	*.*.0.0	Scan
7	4/14 05:00:00	*.*.211.98	*.*.0.0	Scan
8	4/14 06:00:00	*.*.190.102	*.*.0.0	Scan
9	4/14 06:00:00	*.*.126.69	*.*.0.0	Scan
10	4/14 06:00:00	*.*.211.98	*.*.0.0	Scan
11	4/14 06:00:00	*.*.180.155	*.*.0.0	Scan
12	4/14 06:00:00	*.*.249.242	*.*.0.0	Scan
13	4/16 01:00:00	*.*.252.158	*.*.241.171	Login attempt
14	4/16 02:00:00	*.*.252.158	*.*.241.171	Login attempt
15	4/16 10:00:00	*.*.7.111	*.*.241.171	Login attempt
16	4/16 22:00:00	*.*.35.148	*.*.241.171	Login attempt
17	4/17 02:00:00	*.*.242.180	*.*.241.171	Successful login attempt

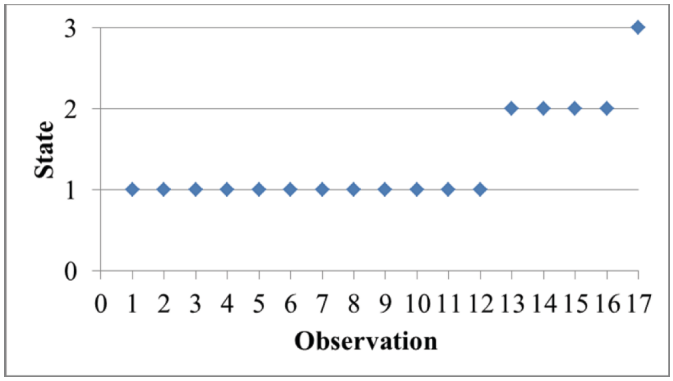

FIGURE 4. An attack sequence detected in the proposed HMM

TABLE 4. Detection performance in Experiment I

Actual \ Detected	Benign	Attacks
Benign	43	5
Attacks	13	69
Accuracy = 0.862; Precision = 0.932; Recall = 0.841		

TABLE 5. Comparison with the existing detection system

Incidents	Results from existing IDS
Scan	21
Login attempt attack	N/A

4.2. **Experiment II: real network.** In order to demonstrate the applicability to real networks, logs from a real network, audit log (from web traffic) and alert log (from IDS), were applied in Experiment II. The log records of five consecutive weeks were collected with an average of four million web requests and ten thousand IDS alerts per day. To evaluate the detection performance in a real environment, the attacks reported by the proposed detection system were analyzed and verified by the administrators. Two detected cases are explained below to demonstrate that the proposed detection system could identify the multi-stage attacks.

4.2.1. *Case A.* For the attack case A, a machine was under stealthy scanning attacks, followed by worm download once it was explored; the machine became a stepping stone attacking others by password guessing attacks. To illustrate the state transitions of the proposed HMM model, the logs and the detection reports were divided into two parts: state 1 to state 2 and state 2 to state 3. Table 6 shows the log records of the attack from state 1 to 2, wherein the machine *.*.4.72 was scanned covertly from different attack sites. Given the observable events in Table 6, the detection report shown in Figure 5 indicates that the victim was under scan attacks and was finally exploited. By correlating the related log records shown in Table 7, the system was able to identify the complete attack sequence from state 1 to 3 shown in Figure 6. The proposed detection system discovered the stealthy attack successfully.

Table 6. Logs demonstrating attack case A from state 1 to 2

Obs. #	Time	Source IP	Destination IP	Action	count
1	02/05 14:22:19	*.*.201.167	*.*.4.72	Scan	1
2	02/05 20:20:30	*.*.102.136	*.*.4.72	Scan	1
3	02/05 21:35:19	*.*.102.136	*.*.4.72	Scan	1
4	02/05 21:37:01	*.*.102.136	*.*.4.72	Scan	2
5	02/05 22:59:02	*.*.11.178	*.*.4.72	Scan	1
6	02/05 23:01:01	*.*.11.178	*.*.4.72	Scan	1
7	02/05 23:17:58	*.*.123.7	*.*.4.72	Scan	1
8	02/05 23:19:01	*.*.123.7	*.*.4.72	Scan	2
9	02/06 08:18:45	*.*.113.23	*.*.4.72	Scan	1
10	02/06 08:20:01	*.*.113.23	*.*.4.72	Scan	2
11	02/06 20:24:49	*.*.159.24	*.*.4.72	Scan	1
12	02/06 21:26:01	*.*.159.24	*.*.4.72	Scan	2
13	02/06 21:35:42	*.*.182.87	*.*.4.72	Scan	1
14	02/06 21:37:01	*.*.182.87	*.*.4.72	Scan	2
15	02/10 10:41:50	*.*.2.182	*.*.4.72	Scan	1
16	02/15 22:50:01	*.*.236.13	*.*.4.72	Worm	1

Table 7. Log results of the state transition to state 3 of attack case A

Obs. #	Time	Source IP	Destination IP	Action
17	02/24 11:23:57	*.*.4.72	*.*.115.155	Login Attempt
18	02/29 08:43:57	*.*.4.72	*.*.115.155	Login Attempt
19	02/29 14:46:36	*.*.4.72	*.*.115.155	Login Attempt

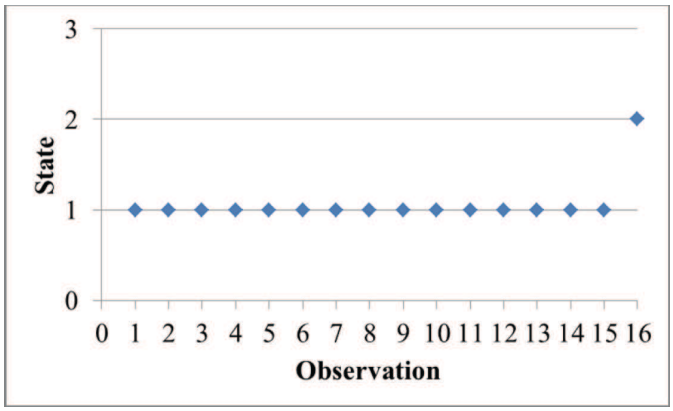

FIGURE 5. Attack case A detected from 1 to 2

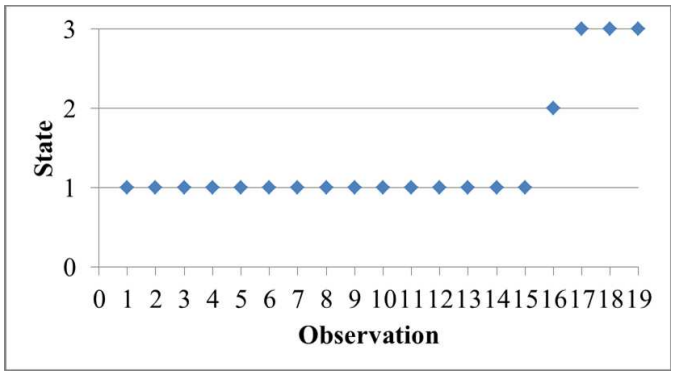

FIGURE 6. Attack case A identified

TABLE 8. Logs of attack case B from state 1 to 2

Obs. #	Time	Source IP	Destination IP	Action
1	02/29 10:18:36	*.*.148.94	*.*.11.138	Login attempt
2	03/01 10:32:55	*.*.198.25	*.*.11.138	Login attempt
3	03/07 19:34:39	*.*.198.25	*.*.11.138	Login attempt
4	03/13 14:07:55	*.*.148.94	*.*.11.138	Login attempt
5	03/20 15:28:29	*.*.148.94	*.*.11.138	Login attempt
6	03/21 10:13:10	*.*.148.94	*.*.11.138	Login attempt
7	03/21 11:35:15	*.*.148.94	*.*.11.138	Login attempt

4.2.2. *Case B.* Attack case B used different attack tactics. First, a machine (*.*.11.138) was attacked by login attempts. Once it was compromised, it became a stepping stone attacking others. The log records shown in Table 8 illustrate the sequence of low frequency attacks identified by the proposed system, but it did not trigger any IDS rule in the real network. The detection report in Figure 7 plots the observations and the corresponding attack states. Table 9 and Figure 8 show the corresponding logs of the transition to state 3 and the detection report. Based on the evaluation, it can be observed that the proposed detection system can identify the multi-stage attacks and is also useful for forecasting on-going attacks to prevent further damage.

5. **Conclusions.** This paper proposed a state-based Hidden Markov Model classification method for detecting the advanced attacks with a sequence of attack stages. The proposed system has been evaluated on a controlled network environment and a real network.

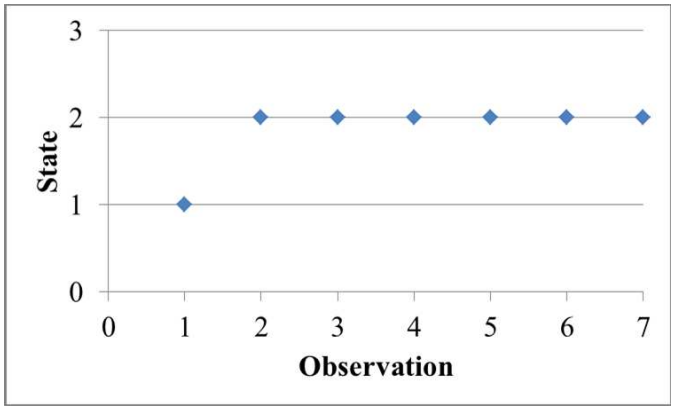

FIGURE 7. Attack case B detected from state 1 to 2

TABLE 9. Logs of attack case B on state 3

Obs. #	Time	Source IP	Destination IP	Action
8	03/21 11:35:15	*.*.11.138	*.*.148.94	Login attempt
9	03/21 11:33:21	*.*.11.138	*.*.148.94	Login attempt

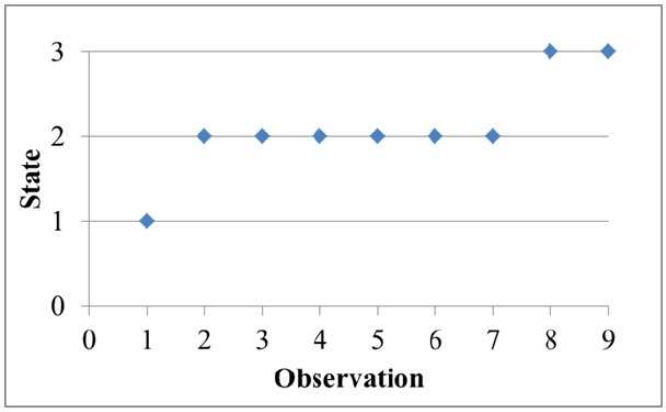

FIGURE 8. Attack case B of state 1 to 3

The mass amount of log records from a distributed network is not feasible for administrators to analyze manually. The proposed system could efficiently correlate logs, reduce the false positives, and improve the efficiency of the security administration work. Based on the results from Experiment II, the proposed detection system reports the attack stage of an attack and is suitable for predicting an on-going multi-stage attack and preventing further damage to the network. The proposed detection system yields a good detection performance with a precision rate of 93.2%, while the existing detection system might produce many false positive alerts and fail to report the multi-stage attacks.

As intrusion detection systems, firewalls, and servers generate different types of logs, further investigation can be done on classifying attacks to reduce the space of suspicious events. In addition, different attack strategies can be applied by attackers in different stages. It is important to categorize the possible attack strategies used in each stage to adopt different state-based classification model to evaluate the detection performance.

REFERENCES

[1] S. Forrest, S. A. Hofmeyr, A. Somayaji and T. A. Longstaff, A sense of self for unix processes, *IEEE Symposium on Security and Privacy*, pp.120-128, 1996.

[2] D. Ourston, S. Matzner, W. Stump and B. Hopkins, Applications of hidden Markov models to detecting multi-stage network attacks, *The 36th Hawaii International Conference on System Sciences*, 2003.
[3] N. Ye, Y. Zhang and C. M. Borror, Robustness of the Markov-chain model for cyber-attack detection, *IEEE Trans. Reliability*, pp.116-123, 2004.
[4] C. C. Lo, C. C. Huang and J. Ku, A cooperative intrusion detection system framework for cloud computing networks, *The 39th International Conference on Parallel Processing Workshops*, pp.280-284, 2010.
[5] S. Zargar, H. Takabi and J. Joshi, DCDIDP: A distributed, collaborative, and data-driven intrusion detection and prevention framework for cloud computing environments, *International Conference on Collaborative Computing: Networking, Applications and Worksharing*, pp.332-341, 2011.
[6] P. Kumar, Nitin, V. Secgal, K. Shah, S. S. P. Shukla and D. S. Chauhan, A novel approach for security in cloud computing using hidden Markov model and clustering, *Information and Communication Technologies*, pp.810-815, 2011.
[7] C. Liu, A. Singhal and D. Wijesekera, A model towards using evidence from security events for network attack analysis, *International Workshop on Security in Information System*, pp.83-95, 2014.
[8] E. Raftopoulos and X. Dimitropoulos, IDS alert correlation in the wild with EDGs, *IEEE Journal on Seleted Areas in Communications*, vol.32, no.10, 2014.
[9] M. M. Siraj, H. H. T. Albasheer and M. M. Din, Towards predictive real-time multi-sensors intrusion alert correlation framework, *Indian Journal of Science and Technology*, vol.8, no.12, 2015.
[10] M. Amini, J. Rezaeenoor and E. Hadavandi, Effective intrusion detection with a neural network ensemble using fuzzy clustering and stacking combination method, *Journal of Computing Security*, vol.1, no.4, 2014.
[11] L. R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, *Proc. of the IEEE*, vol.77, no.2, 1989.
[12] B. Bauer and K. Kraiss, Towards an automatic sign language recognition system using subunits, *Gesture and Sign Language in Human-Computer Interaction, Lecture Notes in Computer Science*, vol.2298, pp.64-75, 2002.
[13] L. Rabiner and B. Juang, An introduction to hidden Markov models, *IEEE Acoustic, Speech, and Signal Processing Magazine*, vol.3, no.1, pp.4-16, 1986.
[14] X. Zan, F. Gao, J. Han and Y. Sun, A hidden Markov model based framework for tracking and predicting of attack intention, *The International Conference on Multimedia Information Networking and Security*, vol.2, pp.498-501, 2009.
[15] X. D. Hoang and J. Hu, An efficient hidden Markov model training scheme for anomaly intrusion detection of server applications based on system calls, *The 12th IEEE International Conference on Networks*, pp.470-474, 2004.
[16] P. Wang, L. Shi, B. Wang, Y. Wu and Y. Liu, Survey on HMM based anomaly intrusion detection using system calls, *The International Conference on Computer Science and Education*, 2010.