Viterbi Algorithm for Intrusion Type Identification in Anomaly Detection System

january 14th 2019

Context

Intrusion Type

- Buffer overflow
  - xlock vulnerability
  - lpset vulnerability
  - kcms_sparc vulnerability
- S/W security vulnerability
- Setup vulnerability
- Denial of service

Markov Chain

A markov Chain is defined by :

. S, A finite set of N states

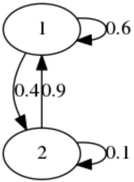
. π, A vector of initial probabilities over S :

$$\pi_i = P(S_1 = i), 1 \leq i \leq N$$

. A, A matrix of probabilities of transitions over SxS :

$$a_{ij} = P(S_t = j | S_{t-1} = i), 1 \leq i \leq N$$

. Markov assumption : $P(S_t | S_{t-1}, S_{t-2}, \ldots, S_1) = P(S_t | S_{t-1})$



$$A = \begin{pmatrix} 0.6 & 0.4 \\ 0.9 & 0.1 \end{pmatrix}$$

Figure: Simple example of Markov Chain

HMM - Hidden Markov Model

– Hidden Markov Model is a statistical model in which the modeled system is supposed to be a Markovian process of unknown parameters.

HMM - Hidden Markov Model

- Hidden Markov Model is a statistical model in which the modeled system is supposed to be a Markovian process of unknown parameters.
- Hidden Markov Model can be viewed as a Bayesian Network

HMM - Hidden Markov Model

- Hidden Markov Model is a statistical model in which the modeled system is supposed to be a Markovian process of unknown parameters.
- Hidden Markov Model can be viewed as a Bayesian Network
- We define a HMM including :

  V, A finite set of M observations

  B, A a matrix of probabilities of observations over state :

$$b_i(k) = P(0_t = V_k | S_t = i)$$

HMM - Forward Algorithm

input : λ The model, O Observed sequence
output : $P(0|\lambda)$
Step 1, Initialization : $\forall i, \alpha_1(i) = \pi_i b_i(0_1)$
Step 2, Induction :
for $t \leftarrow 2 : T$ **do**

$$\forall i, \alpha_t(i) = \left[\sum_{j=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(O_t)$$

end

Step 3, Termination : $P(0|\lambda) = \sum_{i=1}^{N} \alpha_t(i)$

[1]

[1] L. R. Rabiner (1989). "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2, pp. 257–286

HMM - Viterbi Algorithm

input : O Observed sequence
output : $\arg\max_{\lambda\in\Lambda} P(0|\lambda)$
Step 1, Initialization :
for $i \leftarrow 1 : N$ **do**
  $\delta_1(i) = \pi_i b_i(0_1)$
  $\psi_1(i) = 0$
end
Step 2, Recursion :
for $t \leftarrow 2 : T$ **do**
  for $j \leftarrow 1 : N$ **do**
    $\delta_t(j) = \max_i [\delta_{t-1}(i) a_{ij}] b_j(0_t)$
    $\psi_t(j) = \arg\max_i [\delta_{t-1}(i) a_{ij}] b_j(0_t)$
  end
end
Step 3, Termination :
$P^* = \max_{s\in S} [\delta_T(s)]$
$S^*_T = \arg\max_{s\in S} [\delta_T(s)]$
Step 4, Backtracking :
for $t \leftarrow T-1 : 1$ **do**
  $S^*_t = \psi_{t+1}(s^*_{t+1})$
end
return S^*

[2]

[2] A. Viterbi (1967). "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm". In: *IEEE Transactions on Information Theory* 13.2, pp. 260–269

Normal Behaviour Modeling

Normal Behaviour is modelised by a left-to-right HMM λ.

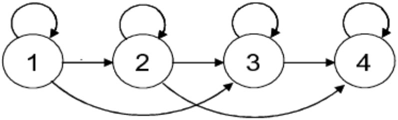


Figure: Left-to-Right Model with jumps

The forward allgorithm is used to decide whether normal or not with a threshold.

Intrusion Detection

Data

$$S = \{1, 2, 3, 4\}$$
$$M = \{1, 2, 3, 4\}$$
$$\pi = \{1.0, 0, 0\}$$
$$O = \{2, 1, 2, 4, 2, 3, 4, 3, 4, 3\}$$

$$A = \begin{pmatrix} 0.28 & 0.34 & 0.28 & 0 \\ 0.0 & 0.32 & 0.21 & 0.47 \\ 0.0 & 0.0 & 0.32 & 0.68 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{pmatrix} \qquad B = \begin{pmatrix} 0.8 & 0.04 & 0.1 & 0.06 \\ 0.0 & 0.13 & 0.45 & 0.42 \\ 0.0 & 0.9 & 0.1 & 0.0 \\ 0.64 & 0.12 & 0.06 & 0.18 \end{pmatrix}$$

Intrusion Detection

Initialization

$$\forall i, \alpha_1(i) = \pi_i b_i(0_1)$$

Intrusion Detection

Initialization

$$\forall i, \alpha_1(i) = \pi_i b_i(0_1)$$

$$\begin{aligned} O_1 = &\quad 2 \\ b_i(0_1) = &\quad (0.04, 0.13, 0.9, 0.12) \end{aligned}$$

Intrusion Detection

Initialization

$\forall i, \alpha_1(i) = \pi_i b_i(0_1)$

$$\begin{aligned} O_1 = & \quad 2 \\ b_i(0_1) = & \quad (0.04, 0.13, 0.9, 0.12) \\ \alpha_1(1) = & \quad \pi_1 * b_1(0_1) = 1 * 0.04 = 0.04 \end{aligned}$$

Intrusion Detection

Initialization

$\forall i, \alpha_1(i) = \pi_i b_i(0_1)$

$$
\begin{aligned}
O_1 = &\quad 2 \\
b_i(0_1) = &\quad (0.04, 0.13, 0.9, 0.12) \\
\alpha_1(1) = &\quad \pi_1 * b_1(0_1) = 1 * 0.04 = 0.04 \\
\alpha_1(2) = &\quad \pi_2 * b_2(0_1) = 0 * 0.13 = 0
\end{aligned}
$$

Intrusion Detection

Initialization

$$\forall i, \alpha_1(i) = \pi_i b_i(0_1)$$

$$\begin{aligned}
O_1 = &\quad 2 \\
b_i(0_1) = &\quad (0.04, 0.13, 0.9, 0.12) \\
\alpha_1(1) = &\quad \pi_1 * b_1(0_1) = 1 * 0.04 = 0.04 \\
\alpha_1(2) = &\quad \pi_2 * b_2(0_1) = 0 * 0.13 = 0 \\
\ldots & \\
\alpha_1 = &\quad \begin{pmatrix} 0.04 & 0 & 0 & 0 \end{pmatrix}
\end{aligned}$$

Intrusion Detection

Induction

for $t \leftarrow 2 : T$ **do**

$$\forall i, \alpha_t(i) = \left[\sum_{j=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(O_t)$$

end

Intrusion Detection

Induction

for $t \leftarrow 2 : T$ **do**

$$\forall i, \alpha_t(i) = \left[\sum_{j=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(O_t)$$

end

$$\begin{aligned} t = & \quad 2 \\ O_2 = & \quad 1 \\ b(0_t) = & \quad \begin{pmatrix} 0.8 & 0 & 0 & 0.64 \end{pmatrix} \\ \alpha_1 = & \quad \begin{pmatrix} 0.04 & 0 & 0 & 0 \end{pmatrix} \end{aligned}$$

Intrusion Detection

Induction

for $t \leftarrow 2 : T$ **do**

$$\forall i, \alpha_t(i) = \left[\sum_{j=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(O_t)$$

end

$$\begin{aligned}
t &= && 2 \\
O_2 &= && 1 \\
b(0_t) &= && \begin{pmatrix} 0.8 & 0 & 0 & 0.64 \end{pmatrix} \\
\alpha_1 &= && \begin{pmatrix} 0.04 & 0 & 0 & 0 \end{pmatrix} \\
\alpha_2(1) &= && \left[\sum_{j=1}^{N} \alpha_{t-1}(1) a_{1j}\right] b_j(O_t) = 0.00896
\end{aligned}$$

Intrusion Detection

Induction

for $t \leftarrow 2 : T$ **do**

$$\forall i, \alpha_t(i) = \left[\sum_{j=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(O_t)$$

end

$$\begin{aligned}
t = &\quad 2 \\
O_2 = &\quad 1 \\
b(0_t) = &\quad \begin{pmatrix} 0.8 & 0 & 0 & 0.64 \end{pmatrix} \\
\alpha_1 = &\quad \begin{pmatrix} 0.04 & 0 & 0 & 0 \end{pmatrix} \\
\alpha_2(1) = &\quad \left[\sum_{j=1}^{N} \alpha_{t-1}(1) a_{1j}\right] b_j(O_t) = 0.00896 \\
&\quad \dots \\
\alpha_2 = &\quad \begin{pmatrix} 0.00896 & 0 & 0 & 0 \end{pmatrix}
\end{aligned}$$

Intrusion Detection

Induction

for $t \leftarrow 2 : T$ **do**

$$\forall i, \alpha_t(i) = \left[\sum_{j=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(O_t)$$

end

$$\alpha = \begin{pmatrix} 0.04 & 0 & 0 & 0 \\ 0.00896 & 0 & 0 & 0 \\ 0.00010035 & 0.00039603 & 0.0022579 & 0 \\ 1.8882e^{-08} & 2.8849e^{-06} & 1.3193e^{-05} & 4.0995e^{-05} \\ 5.287e^{-10} & 4.1831e^{-07} & 4.8329e^{-07} & 3.0793e^{-06} \\ 8.8822e^{-12} & 5.6297e^{-08} & 0 & 6.4882e^{-07} \\ 2.487e^{-13} & 8.1081e^{-09} & 1.1825e^{-09} & 4.0517e^{-08} \\ 4.1782e^{-15} & 1.0898e^{-09} & 0 & 8.1237e^{-09} \\ 1.1699e^{-16} & 1.5693e^{-10} & 2.2885e^{-11} & 5.1816e^{-10} \end{pmatrix}$$

Intrusion Detection

Termination

$$P(0|\lambda)) = \sum_{i=1}^{N} \alpha_t(i)$$
$$= 1.1699e^{-16} + 1.5693e^{-10} + 2.2885e^{-11} + 5.1816e^{-10}$$
$$= 6.9797e^{-10}$$

Intrusion Detection

Decision

if $log(P(0|\lambda)) >$ *threshold* **then**
  return *Normal Behaviour*
else
  return *Intrusion*
end

$$log(P(0|\lambda) = -21.083 < threshold(-20.83) \implies Intrusion$$

Intrusion Detection

Results

Table: The performance of HMM-based IDS. Best results are in bold

Length	Thresold	Detection Rate	F-P Error
10	-9.43	100%	2.626
15	-9.43	100%	3.614
10	-14.42	100%	1.366
15	-14.42	100%	2.718
10	-16.94	100%	0.789
15	-16.94	100%	2.618
10	-18.35	100%	0.553
15	-18.35	100%	2.535
10	-19.63	100%	0.476
15	-19.63	100%	2.508
10	**-20.83**	**100%**	**0.372**
15	-20.83	100%	2.473

Intrusion Type Identification

Process in two steps :

- Viterbi algorithm used to find the optimal state sequence
- Euclidean distance to identify the intrusion type with the optimal state sequence

Intrusion Detection

Data

$$S = \{1, 2, 3, 4\}$$
$$M = \{1, 2, 3, 4\}$$
$$\pi = \{1.0, 0, 0\}$$
$$O = \{2, 1, 2, 4, 2, 3, 4, 3, 4, 3\}$$

$$A = \begin{pmatrix} 0.28 & 0.34 & 0.28 & 0 \\ 0.0 & 0.32 & 0.21 & 0.47 \\ 0.0 & 0.0 & 0.32 & 0.68 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{pmatrix} \qquad B = \begin{pmatrix} 0.8 & 0.04 & 0.1 & 0.06 \\ 0.0 & 0.13 & 0.45 & 0.42 \\ 0.0 & 0.9 & 0.1 & 0.0 \\ 0.64 & 0.12 & 0.06 & 0.18 \end{pmatrix}$$

Intrusion Type Identification

Initialization

for $i \leftarrow 1 : N$ **do**

$\quad \delta_1(i) = \pi_i b_i(0_1)$

$\quad \psi_1(i) = 0$

end

$$
\begin{aligned}
O_1 &= 2 \\
b_i(0_1) &= (0.04, 0.13, 0.9, 0.12)
\end{aligned}
$$

Intrusion Type Identification

Initialization

for $i \leftarrow 1 : N$ **do**
  $\delta_1(i) = \pi_i b_i(0_1)$
  $\psi_1(i) = 0$
end

$$
\begin{aligned}
O_1 = &\quad 2 \\
b_i(0_1) = &\quad (0.04, 0.13, 0.9, 0.12) \\
\delta_1(1) = &\quad \pi_1 * b_1(0_1) = 1 * 0.04 = 0.04
\end{aligned}
$$

Intrusion Type Identification

Initialization

for $i \leftarrow 1 : N$ **do**
$\delta_1(i) = \pi_i b_i(0_1)$
$\psi_1(i) = 0$
end

$$
\begin{aligned}
O_1 &= 2 \\
b_i(0_1) &= (0.04, 0.13, 0.9, 0.12) \\
\delta_1(1) &= \pi_1 * b_1(0_1) = 1 * 0.04 = 0.04 \\
\delta_1(2) &= \pi_2 * b_2(0_1) = 0 * 0.13 = 0
\end{aligned}
$$

Intrusion Type Identification

Initialization

for $i \leftarrow 1 : N$ **do**

$\quad \delta_1(i) = \pi_i b_i(0_1)$

$\quad \psi_1(i) = 0$

end

$$
\begin{aligned}
O_1 = \quad & 2 \\
b_i(0_1) = \quad & (0.04, 0.13, 0.9, 0.12) \\
\delta_1(1) = \quad & \pi_1 * b_1(0_1) = 1 * 0.04 = 0.04 \\
\delta_1(2) = \quad & \pi_2 * b_2(0_1) = 0 * 0.13 = 0 \\
\ldots & \\
\delta_1 = \begin{pmatrix} 0.04 & 0 & 0 & 0 \end{pmatrix} &
\end{aligned}
$$

Intrusion Type Identification

Initialization

for $i \leftarrow 1 : N$ **do**
  $\delta_1(i) = \pi_i b_i(0_1)$
  $\psi_1(i) = 0$
end

$$O_1 = 2$$
$$b_i(0_1) = (0.04, 0.13, 0.9, 0.12)$$
$$\delta_1(1) = \pi_1 * b_1(0_1) = 1 * 0.04 = 0.04$$
$$\delta_1(2) = \pi_2 * b_2(0_1) = 0 * 0.13 = 0$$
$$\dots$$
$$\delta_1 = \begin{pmatrix} 0.04 & 0 & 0 & 0 \end{pmatrix}$$

$$\psi_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}$$

Intrusion Type Identification

Recursion

for $t \leftarrow 2 : T$ **do**
  for $j \leftarrow 1 : N$ **do**
    $\delta_t(j) = \max_i [\delta_{t-1}(i) a_{ij}] b_j(0_t)$
    $\psi_t(j) = \arg\max_i [\delta_{t-1}(i) a_{ij}] b_j(0_t)$
  end
end

$$t = \quad 2$$
$$O_2 = \quad 1$$
$$\delta_1 = \quad \begin{pmatrix} 0.04 & 0 & 0 & 0 \end{pmatrix}$$

Intrusion Type Identification

Recursion

for $t \leftarrow 2 : T$ **do**
  for $j \leftarrow 1 : N$ **do**
    $\delta_t(j) = \max_i [\delta_{t-1}(i) a_{ij}] b_j(0_t)$
    $\psi_t(j) = \arg\max_i [\delta_{t-1}(i) a_{ij}] b_j(0_t)$
  end
end

$$
\begin{aligned}
t = & \quad 2 \\
O_2 = & \quad 1 \\
\delta_1 = & \quad \begin{pmatrix} 0.04 & 0 & 0 & 0 \end{pmatrix} \\
\delta_2(1) = & \quad \max_i [\delta_{t-1}(i) a_{i1}] b_1(0_2) \\
= & \quad 0.00896 \\
\delta_2 = & \quad \begin{pmatrix} 0.00896 & 0 & 0 & 0 \end{pmatrix}
\end{aligned}
$$

Intrusion Type Identification

Recursion

for $t \leftarrow 2 : T$ **do**
  for $j \leftarrow 1 : N$ **do**
    $\delta_t(j) = \max_i [\delta_{t-1}(i) a_{ij}] b_j(0_t)$
    $\psi_t(j) = \arg\max_i [\delta_{t-1}(i) a_{ij}] b_j(0_t)$
  end
end

$$
\begin{aligned}
t = &\quad 2 \\
O_2 = &\quad 1 \\
\delta_1 = &\quad \begin{pmatrix} 0.04 & 0 & 0 & 0 \end{pmatrix} \\
\delta_2(1) = &\quad \max_i [\delta_{t-1}(i) a_{i1}] b_1(0_2) \\
= &\quad 0.00896 \\
\delta_2 = &\quad \begin{pmatrix} 0.00896 & 0 & 0 & 0 \end{pmatrix} \\
\psi_2(1) = &\quad \arg\max_i [\delta_{t-1}(i) a_{i1}] b_1(0_2) \\
= &\quad 0
\end{aligned}
$$

Intrusion Type Identification

Recursion

for $t \leftarrow 2 : T$ **do**
  for $j \leftarrow 1 : N$ **do**
    $\delta_t(j) = \max_i[\delta_{t-1}(i)a_{ij}]b_j(0_t)$
    $\psi_t(j) = \arg\max_i[\delta_{t-1}(i)a_{ij}]b_j(0_t)$
  end
end

$$
\begin{aligned}
t = &\quad 2 \\
O_2 = &\quad 1 \\
\delta_1 = &\quad \begin{pmatrix} 0.04 & 0 & 0 & 0 \end{pmatrix} \\
\delta_2(1) = &\quad \max_i[\delta_{t-1}(i)a_{i1}]b_1(0_2) \\
= &\quad 0.00896 \\
\delta_2 = &\quad \begin{pmatrix} 0.00896 & 0 & 0 & 0 \end{pmatrix} \\
\psi_2(1) = &\quad \arg\max_i[\delta_{t-1}(i)a_{i1}]b_1(0_2) \\
= &\quad 0 \\
\psi_2 = &\quad \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}
\end{aligned}
$$

Intrusion Type Identification

Recursion

for $t \leftarrow 2 : T$ **do**
  for $j \leftarrow 1 : N$ **do**
    $\delta_t(j) = \max_i [\delta_{t-1}(i) a_{ij}] b_j(0_t)$
    $\psi_t(j) = \arg\max_i [\delta_{t-1}(i) a_{ij}] b_j(0_t)$
  end
end

$$\delta = \begin{pmatrix} 0.04 & 0 & 0 & 0 \\ 0.00896 & 0 & 0 & 0 \\ 0.00010035 & 0.00039603 & 0.0022579 & 0 \\ 1.6859e^{-06} & 5.3227e^{-05} & 0 & 0.00027637 \\ 1.8882e^{-08} & 2.2142e^{-06} & 1.006e^{-05} & 3.3164e^{-05} \\ 5.287e^{-10} & 3.1885e^{-07} & 3.2192e^{-07} & 1.9899e^{-06} \\ 8.8822e^{-12} & 4.2853e^{-08} & 0 & 3.5817e^{-07} \\ 2.487e^{-13} & 6.1709e^{-09} & 8.9992e^{-10} & 2.149e^{-08} \\ 4.1782e^{-15} & 8.2937e^{-10} & 0 & 3.8683e^{-09} \\ 1.1699e^{-16} & 1.1943e^{-10} & 1.7417e^{-11} & 2.321e^{-10} \end{pmatrix} \quad \psi = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 \\ 0 & 1 & 1 & 3 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 0 & 3 \\ 0 & 1 & 1 & 3 \\ 0 & 1 & 0 & 3 \\ 0 & 1 & 1 & 3 \end{pmatrix}$$

Intrusion Type Identification

Termination

$$P^* = \max_{s \in S}[\delta_T(s)] = 2.321e^{-10}$$

Intrusion Type Identification

Backtracking

for $t \leftarrow T - 1 : 1$ **do**
  $S_t^* = \psi_{t+1}(s_{t+1}^*)$
end

Optimal Sequence $S^* = \{1, 1, 3, 4, 4, 4, 4, 4, 4, 4\}$

Intrusion Type Identification

Decision

Table: Sequences for each type of intrusion

Type	Sequence	Distance
xlock	$\{2, 2, 3, 3, 3, 4, 4, 4, 4, 4\}$	3.7417
ipset	$\{2, 3, 3, 3, 4, 4, 4, 4, 4, 4\}$	4.4721
kcms_sparc	$\{1, 1, 2, 2, 2, 2, 4, 4, 4, 4\}$	3

Intrusion Type Identification

Results

Table: The performance of Viterbi-based Intrusion Type Identification. (A:xlock, B: lpset, C: kcms_sparc, D: processe creation, E: fill the disk, F: exhausting the memory)

	A	B	C	D	E	F	Rate
A	8	1	_	_	_	_	88%
B	_	6	1	_	_	_	86%
C	_	_	4	_	_	_	100%
D	_	_	_	3	_	6	33%
E	_	_	_	4	_	3	0%
F	_	_	_	2	1	6	66%

Intrusion Type Identification

Results

Table: The performance of Viterbi-based Intrusion Type Identification

Attack	Trial	Correct	Incorrect	Rate
Buffer Overflow	20	18	2	90%
Denial of Service	25	9	16	36%
Buffer Overflow	45	27	18	60%

Limitations & Remarks

- Try other distance metrics for Intrusion Type Identification : Ja-Min Koo and Sung-Bae Cho (2005). "Effective Intrusion Type Identification with Edit Distance for HMM-Based Anomaly Detection System". In: *Pattern Recognition and Machine Intelligence*. Ed. by Sankar K. Pal, Sanghamitra Bandyopadhyay, and Sambhunath Biswas. Springer Berlin Heidelberg

Limitations & Remarks

- Try other distance metrics for Intrusion Type Identification : Ja-Min Koo and Sung-Bae Cho (2005). "Effective Intrusion Type Identification with Edit Distance for HMM-Based Anomaly Detection System". In: *Pattern Recognition and Machine Intelligence*. Ed. by Sankar K. Pal, Sanghamitra Bandyopadhyay, and Sambhunath Biswas. Springer Berlin Heidelberg
- Hypothesis that there is only one sequence of state per each intrusion, and that it never changes.

Limitations & Remarks

- Try other distance metrics for Intrusion Type Identification : Ja-Min Koo and Sung-Bae Cho (2005). "Effective Intrusion Type Identification with Edit Distance for HMM-Based Anomaly Detection System". In: *Pattern Recognition and Machine Intelligence*. Ed. by Sankar K. Pal, Sanghamitra Bandyopadhyay, and Sambhunath Biswas. Springer Berlin Heidelberg
- Hypothesis that there is only one sequence of state per each intrusion, and that it never changes.
- This model is anomaly-based, but use the fact that we are supposed to know the sequence of state of the intrusion. they loose the main advantage of anomaly-based IDS to detect new types of intrusion.

Limitations & Remarks

- Low detection efficiency, especially due to the high false positive rate usually obtained Stefan Axelsson (1998). *Research in intrusion-detection systems: A survey*. Tech. rep. Technical report 98–17. Department of Computer Engineering, Chalmers ...

Limitations & Remarks

- Low detection efficiency, especially due to the high false positive rate usually obtained Stefan Axelsson (1998). *Research in intrusion-detection systems: A survey*. Tech. rep. Technical report 98–17. Department of Computer Engineering, Chalmers . . .
- Absence of appropriate metrics and assessment methodologies, as well as a general framework for evaluating and comparing alternative IDS techniques Salvatore J Stolfo et al. (2000). *Cost-based modeling for fraud and intrusion detection: Results from the JAM project*. Tech. rep. COLUMBIA UNIV NEW YORK DEPT OF COMPUTER SCIENCE

Methods using HMM

Intrusion Alert Prediction Using a Hidden Markov Mode

Alert prediction method based on prediction of the next alert cluster

Clusters contains :

- source IP address
- destination IP range
- alert type
- alert category.

Prediction of next alert cluster provides more information about future strategies of the attacker and does not depend on specific domain knowledge

3

[3] Udaya Sampath K Thanthrige, Jagath Samarabandu, and Xianbin Wang (2016). "Intrusion alert prediction using a hidden Markov model". In: *arXiv preprint arXiv:1610.07276*

Methods using HMM

Anomalybased HMMs

- Used for intrusion detection, with five states and six observation symbols per state
- States in the model are interconnected in such a way that any state can be reached from any other state
- Baum-Welch method is used

4

[4] Shrijit S Joshi and Vir V Phoha (2005). "Investigating hidden Markov models capabilities in anomaly detection". In: *Proceedings of the 43rd annual Southeast regional conference-Volume 1*. ACM, pp. 98–103

Other Methods

Technique: *basics*	■ Pros / ■ Cons	Subtypes
A) Statistical-based: *stochastic behaviour*	■ Prior knowledge about normal activity not required. Accurate notification of malicious activities. ■ Susceptible to be trained by attackers. Difficult setting for parameters and metrics. Unrealistic quasi-stationary process assumption.	A.1) Univariate models *(independent Gaussian random variables)* A.2) Multivariate models *(correlations among several metrics)* A.3) Time series *(interval timers, counters and some other time-related metrics)*
B) Knowledge-based: *availability of prior knowledge/data*	■ Robustness. Flexibility and scalability. ■ Difficult and time-consuming availability for high-quality knowledge/data.	B.1) Finite state machines *(states and transitions)* B.2) Description languages *(N-grams, UML, ...)* B.3) Expert systems *(rules-based classification)*
C) Machine learning-based: *categorization of patterns*	■ Flexibility and adaptability. Capture of interdependencies. ■ High dependency on the assumption about the behaviour accepted for the system. High resource consuming.	C.1) Bayesian networks *(probabilistic relationships among variables)* C.2) Markov models *(stochastic Markov theory)* C.3) Neural networks *(human brain foundations)* C.4) Fuzzy logic *(approximation and uncertainty)* C.5) Genetic algorithms *(evolutionary biology inspired)* C.6) Clustering and outlier detection *(data grouping)*

[5]

[5] Pedro Garcia-Teodoro et al. (2009). "Anomaly-based network intrusion detection: Techniques, systems and challenges". In: *computers & security* 28.1-2, pp. 18–28

Other Methods

TABLE VII
COMPLEXITY OF ML AND DM ALGORITHMS DURING TRAINING

Algorithm	Typical Time Complexity	Streaming Capable	Comments
ANN	O(emnk)	low	Jain et al. [107] e: number of epochs k: number of neurons
Association Rules	>> $O(n^3)$	low	Agrawal et al. [108]
Bayesian Network	>> O(mn)	high	Jensen [41]
Clustering, k-means	O(kmni)	high	Jain and Dubes [46] i: number of iterations until threshold is reached k: number of clusters
Clustering, hierarchical	$O(n^3)$	low	Jain and Dubes [46]
Clustering, DBSCAN	O(n log n)	high	Ester et al. [109]
Decision Trees	$O(mn^2)$	medium	Quinlan [54]
GA	O(gkmn)	medium	Oliveto et al. [110] g: number of generations k: population size
Naïve Bayes	O(mn)	high	Witten and Frank [89]
Nearest Neighbor k-NN	O(n log k)	high	Witten and Frank [89] k: number of neighbors
HMM	$O(nc^2)$	medium	Forney [111] c: number of states (categories)
Random Forest	O(Mmn log n)	medium	Witten and Frank [89] M: number of trees
Sequence Mining	>> $O(n^3)$	low	Agrawal and Srikant [92]
SVMs	$O(n^2)$	medium	Burges [112]

[6]

[6] Anna L Buczak and Erhan Guven (2016). "A survey of data mining and machine learning methods for cyber security intrusion detection". In: *IEEE Communications Surveys & Tutorials* 18.2, pp. 1153–1176