IoTMonitor: A Hidden Markov Model-based Security System to Identify Crucial Attack Nodes in Trigger-action IoT Platforms

Md Morshed Alam, Md Sajidul Islam Sajid, Weichao Wang, Jinpeng Wei

UNIVERSITY OF NORTH CAROLINA CHARLOTTE

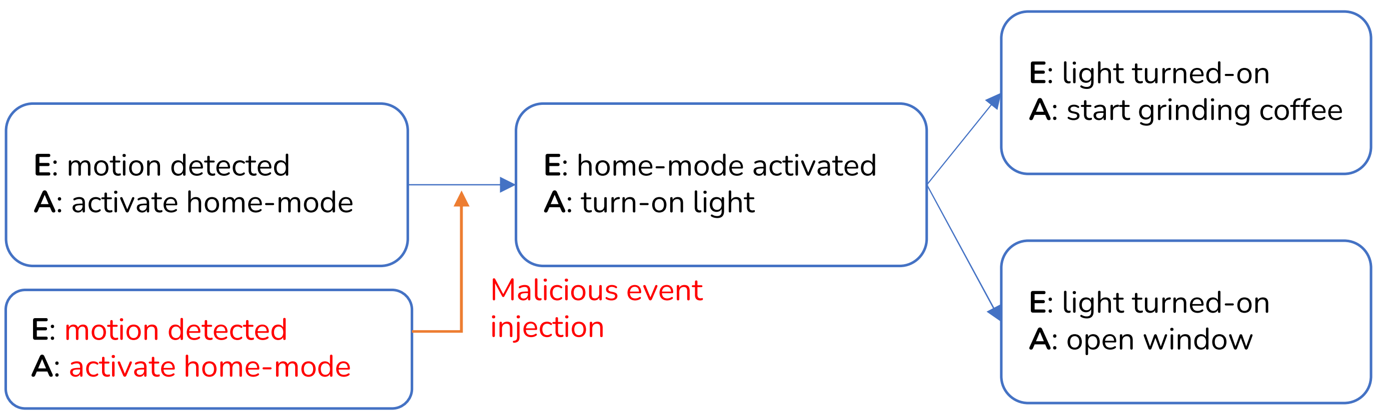
IEEE WCNC 2022

Motivation

- Trigger-action IoT platforms (e.g., IFTTT) are getting popular
- Chain of **interactions** creates security vulnerabilities
- Attackers inject malicious events remotely

IFTTT
ifthisthenthat

Problem Statement



- How can we determine the optimal attack path an attacker may adopt to implement a trigger-action based attack?

Existing Approaches

Approach-1:

- Performing static analysis on application source code
- Instrumenting customized codes
- Generating system models at runtime
- Identifying and blocking unsafe and insecure state transitions

Example: IoTGuard [1]

[1] Z. B. Celik, G. Tan, and P. Mcdaniel, "IOTGUARD : Dynamic Enforcement of Security and Safety Policy in Commodity IoT," no. February, 2019.

Existing Approaches (Contd.)

Approach-2:

- Analyzing network traffics to extract wireless fingerprints
- Using supervised learning methods to identify malicious activities

Example: HoMonit [1]

1] W. Zhang, "HoMonit : Monitoring Smart Home Apps from Encrypted Traffic," Comput. Commun. Secur., pp. 1074–1088, 2018.

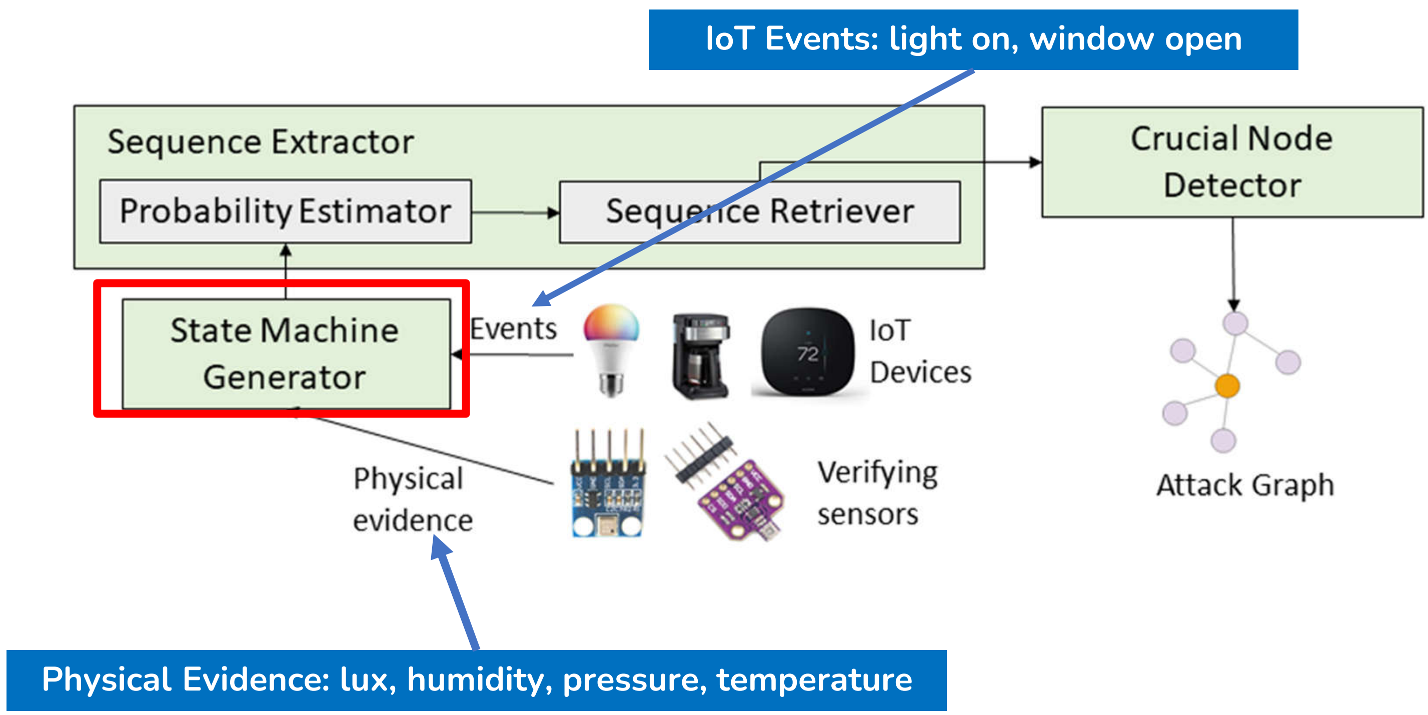
IoTMonitor at a Glance

- A **Hidden Markov Model** based security system
- Discovers probabilistic relationships between IoT event occurrences and physical evidence

Goals:

- Determining optimal attack sequence from a set of physical evidence
- Identifying the most frequently triggered IoT events

System Architecture

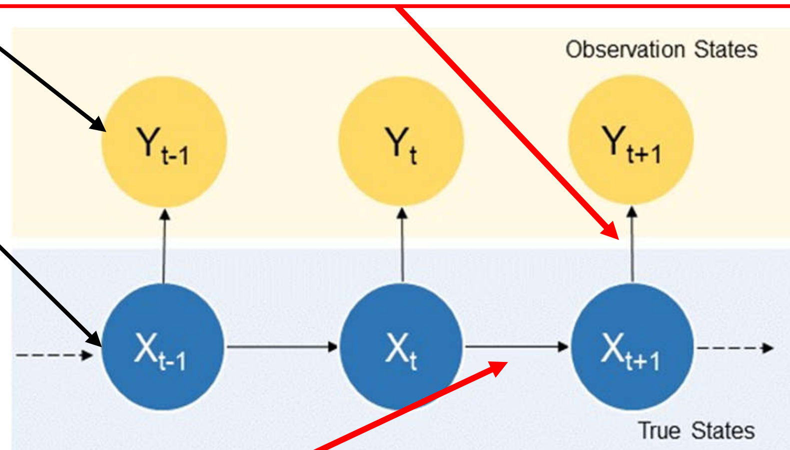


State Machine Generator

Emission Probability, $\mu_j(y_k) = Pr(Y_{t+1} = y_k \mid X_{t+1} = x_j)$

Observation states, $Y_j \subset \{ph_1, ., ph_L\}$

True states, $X_i \in \{d_1, d_2, \ldots.., d_N\}$



$\sigma = \{\sigma_i\}$, $\sigma_i = Pr(X_1 = x_i)$

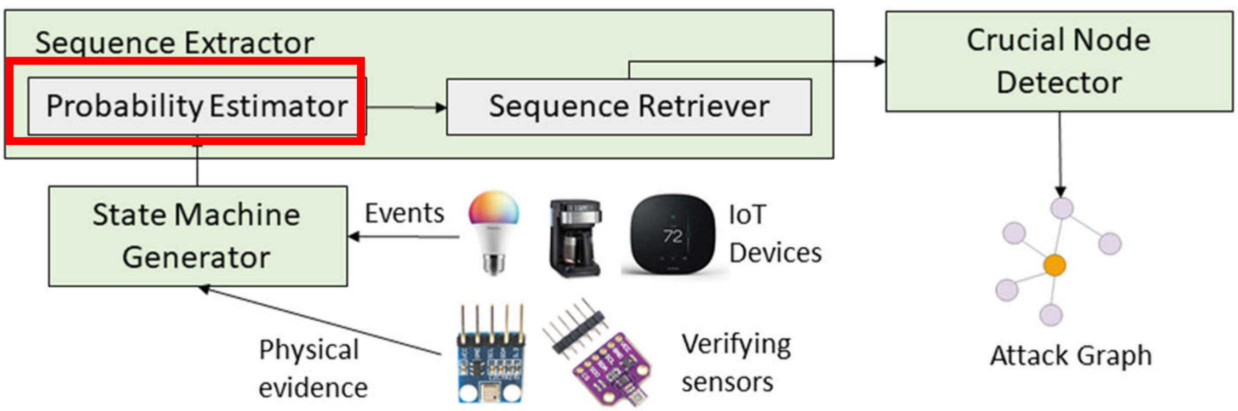
$Q = \{q_{ij}\}$,

$E = \{\mu_j(y_k)\}$

State Transition Probability, $q_{ij} = Pr(X_{t+1} = x_j \mid X_t = x_i)$

Current Model: $\theta = (\sigma, Q, E)$

Probability Estimator



Goal: Given the observation sequence $Y = \{Y_1, Y_2, \dots, Y_T\}$, determine

$$\theta^* = \underset{\theta}{argmax}\ Pr(Y_1, Y_2, ..., Y_T|\theta)$$

UNIVERSITY OF NORTH CAROLINA CHARLOTTE

IEEE WCNC 2022

Probability Estimator (Contd.)

- IoTMonitor uses **forward-backward procedure** to calculate θ^*.

the probability of being in the state x_i at time t given a history of observations <Y_1, Y_2, …, Y_t>

$$\alpha_t(i) = Pr(Y_1, Y_2, ..., Y_t, X_t = x_i|\theta)$$

$$\beta_t(i) = Pr(Y_{t+1}, Y_{t+2}, ..., Y_T|X_t = x_i, \theta)$$

the probability of being in the state x_i at time t given a set of observations <Y_{t+1}, Y_{t+2}, …., Y_T>

Initialization

$$\alpha_1(i) = \sigma_i \mu_i(Y_1), \quad 1 \le i \le N$$

$$\beta_T(i) = 1, \quad 1 \le i \le N$$

Induction

$$\alpha_{t+1}(j) = \mu_j(Y_{t+1}) \sum_{i=1}^{N} \alpha_t(i) q_{ij}, \quad 1 \le t \le T-1, \quad 1 \le j \le N$$

$$\beta_t(i) = \sum_{j=1}^{N} q_{ij} \mu_j(Y_{t+1}) \beta_{t+1}(j), \ t = T-1, ..., 2, 1, \quad 1 \le i \le N$$

UNIVERSITY OF NORTH CAROLINA CHARLOTTE

IEEE WCNC 2022

Probability Estimator (Contd.)

$$\delta_t(i) = Pr(X_t = x_i | Y_1, Y_2, ..., Y_T, \theta) = \frac{Pr(X_t = x_i, Y_1, Y_2, ..., Y_T | \theta)}{Pr(Y_1, Y_2, ..., Y_T | \theta)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

$$\xi_t(i,j) = Pr(X_t = x_i, X_{t+1} = x_j | Y_1, Y_2, ..., Y_T, \theta) = \frac{Pr(X_t = x_i, X_{t+1} = x_j, Y_1, Y_2, ..., Y_T | \theta)}{Pr(Y_1, Y_2, ..., Y_T | \theta)} = \frac{\alpha_t(i) q_{ij} \beta_{t+1}(j) \mu_j(Y_{t+1})}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) q_{ij} \beta_{t+1}(j) \mu_j(Y_{t+1})}$$

$\xi_t(i, j)$ = the probability of the system being in the true states x_i and x_j at time instances t and t+1

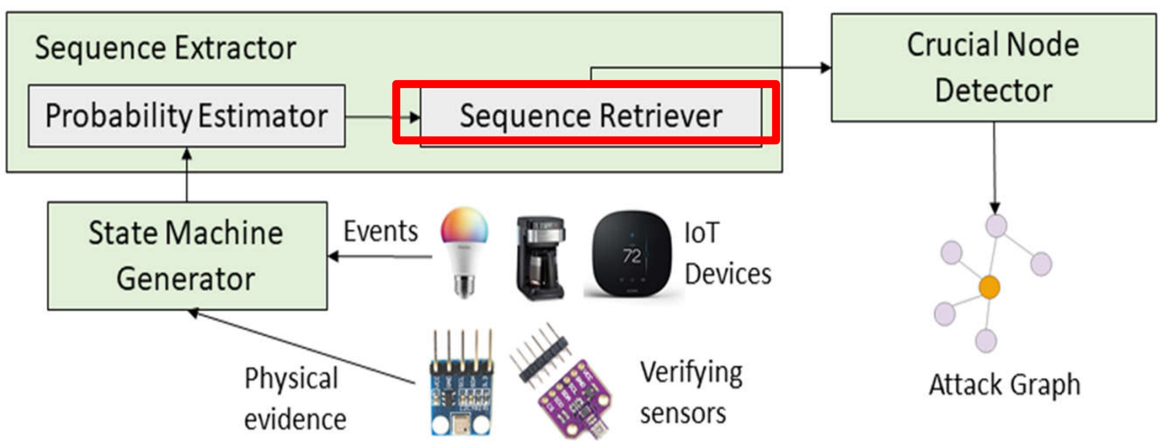
$\delta_t(i)$ = the probability of the system being in the true state x_i at time instance t

$$\bar{\sigma}_i = \delta_1(i)$$

$$\bar{q}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \delta_t(i)}$$

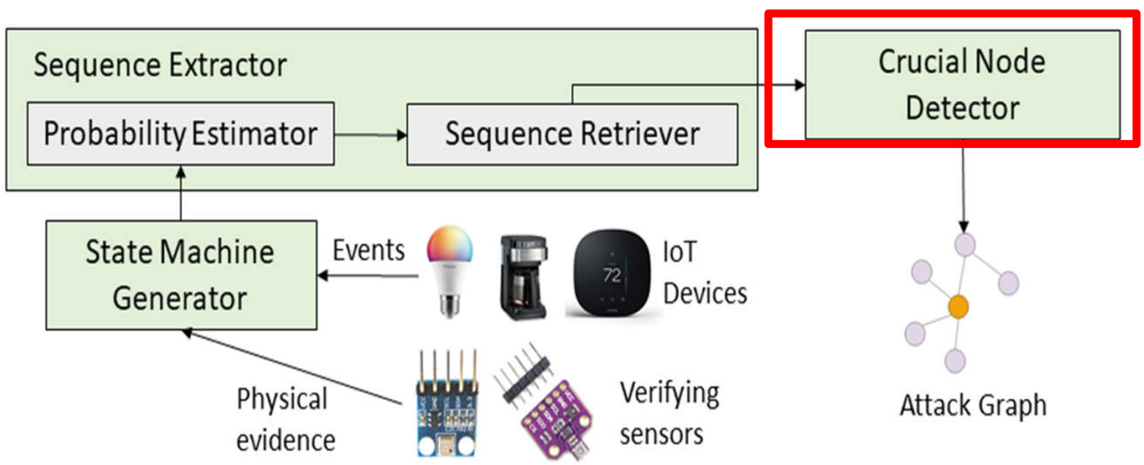
$$\bar{\mu}_j(y_k) = \frac{\sum_{t=1}^{T} 1_{(Y_{t+1}=y_k)} \delta_t(j)}{\sum_{t=1}^{T} \delta_t(j)}$$

Sequence Retriever



Goal: $\omega_t(i) = \max_{x_1,...,x_{i-1}} \left\{ Pr(\boxed{X_1 = x_1, ..., X_t = x_i,} Y_1, ..., Y_t = y_k|\theta) \right\}$

Crucial Node Detector



Goal: To identify the most frequently triggered events

UNIVERSITY OF NORTH CAROLINA CHARLOTTE

IEEE WCNC 2022

Crucial Node Detector (Contd.)

Algorithm 1 Crucial node detection algorithm

Input: $X, \Upsilon_1, \Upsilon_2, ..., \Upsilon_p$

Output: Pairs of true states responding to the most frequently triggered events

$i \leftarrow 1$
while $i \leq p$ **do**
  $S_i \leftarrow$ Longest Common Subsequence between X and Υ_i
  for $j \leftarrow 1$ to $(|S_i| - 1)$ **do**
    $E[i,j] \leftarrow \{S_i[j], S_i[j+1]\}$
    if $E[i,j]$ not in $SCORE.Keys()$ **then**
      $SCORE[E[i,j]] \leftarrow 1$
    else
      $SCORE[E[i,j]] \leftarrow SCORE[E[i,j]] + 1$
    end if
  end for
end while
return $\underset{E[i,j]}{argmax}\ (SCORE[E[i,j]])$

Number of times a particular pair is present in the sequence

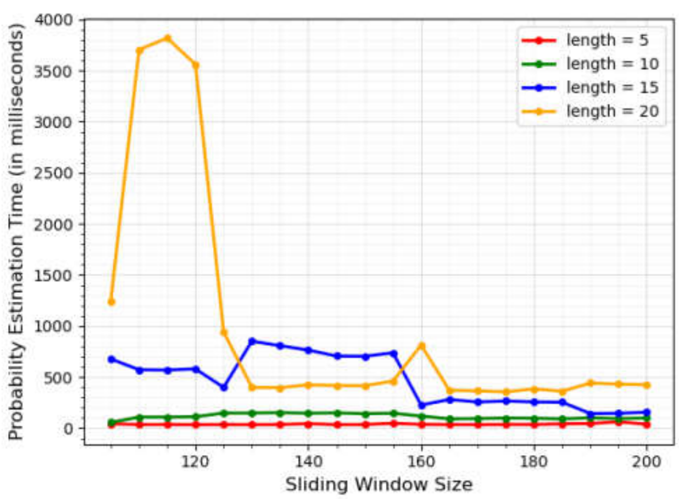
Experimental Settings

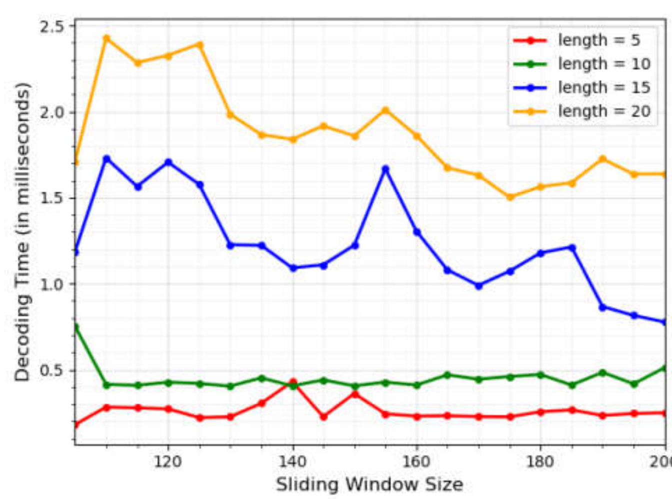
- Utilized the PEEVES [1] dataset
- Data collected from 12 distinct IoT devices and 48 sensors
- Conceptualized a sliding window w_i
- When an event is occurred at time t_i, we consider all sensor measurements collected within the time period (t_i+w_i) for the purpose of event verification

[1] S. Birnbach, S. Eberz, and I. Martinovic, "Peeves: Physical Event Verification in Smart Homes," *Proc. ACM Conf. Comput. Commun. Secur.*, pp. 1455–1467, 2019.

Probability Estimation Time vs Decoding Time

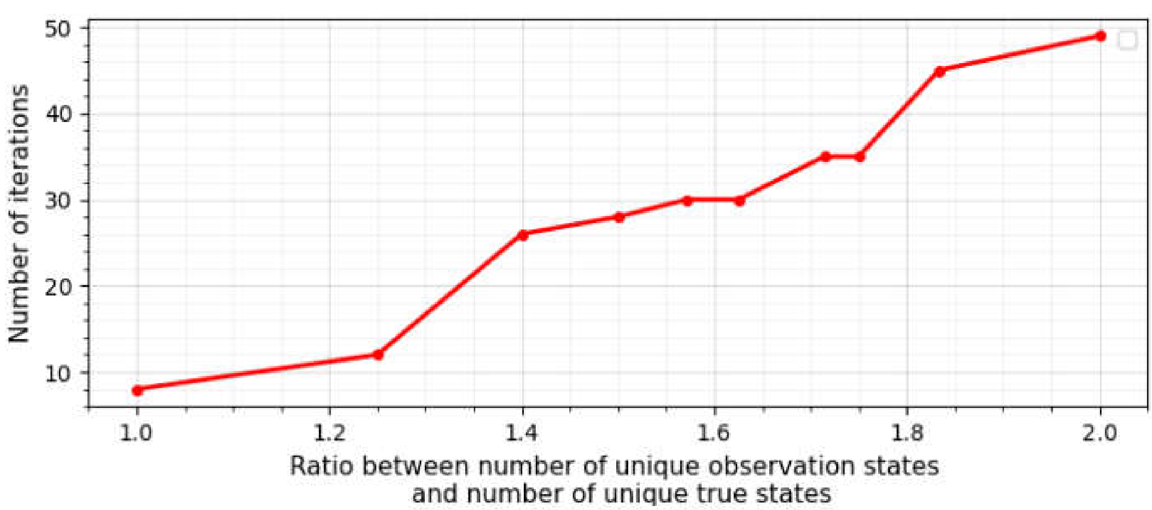
- **Estimation time**: the time required to estimate the converged θ^*
- **Decoding time**: the time required to extract the hidden sequence





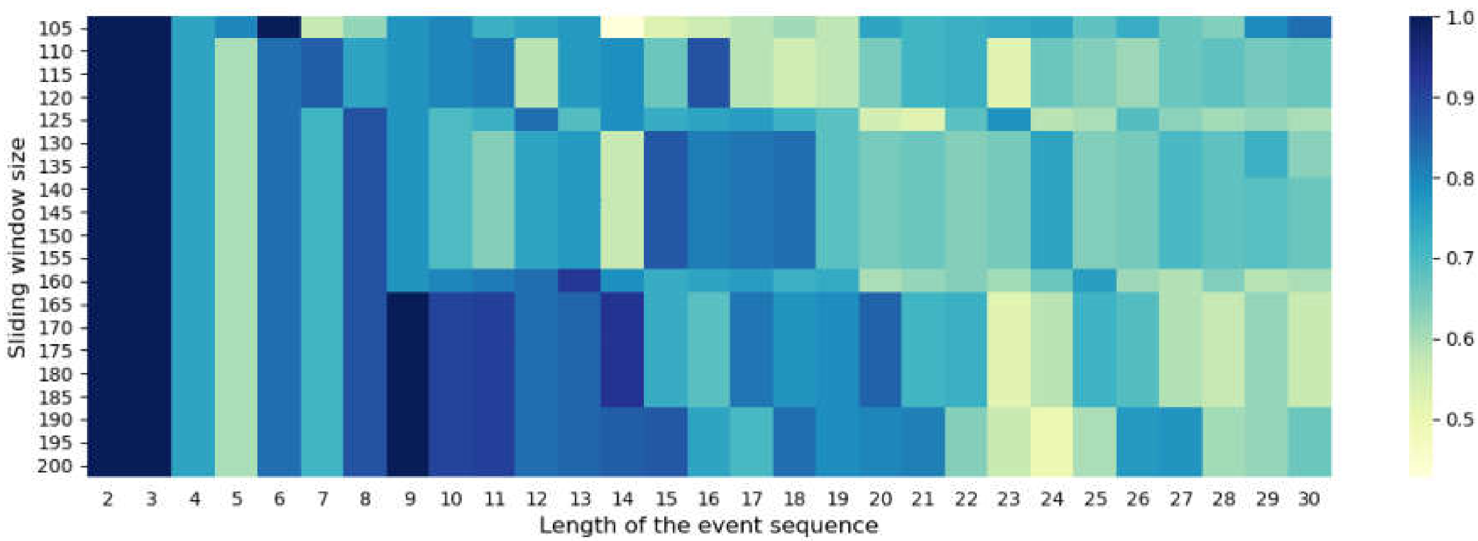
Computational Overhead

- We compute computational overhead for **forward-backward procedure** since IoTMonitor spends most of the computations for estimating probabilities



Accuracy Score

- Determines how accurately the extracted hidden sequence of events represent the actual IoT events triggered during the attack



UNIVERSITY OF NORTH CAROLINA CHARLOTTE

IEEE WCNC 2022

Future Work

- Modeling the joint contribution of multiple events into leading a single trigger operation
- Investigating noisy sensor’s impact on the observation space and the detection accuracy of attack path

Conclusion

- IoTMonitor uses a Hidden Markov Model based approach to determine the optimal attack sequence
- IoTMonitor leverages the probabilistic relation between physical evidence captured by sensors and actual IoT events triggered
- IoTMonitor discerns the underlying event sequence with >=90% accuracy mostly

Questions?