Latent states: model-based machine learning perspectives on cyber resilience

Marin François*, Pierre-Emmanuel Arduin† Myriam Merad‡
*LAMSADE, UMR CNRS 7243, Université Paris-Dauphine PSL, Paris, France
Email: marin.francois@dauhine.psl.eu
†Université Paris-Dauphine PSL, DRM, UMR CNRS 7088, Paris, France
Email: pierre-emmanuel.arduin@dauhine.psl.eu
‡Université Paris-Dauphine PSL, LAMSADE, UMR CNRS 7243, Paris, France
Email: myriam.merad@dauhine.psl.eu

***Abstract*—This paper presents a novel approach for modeling networks cyber resilience when undergoing cyberattack. We offer to use Model-Based Machine Learning (MBML) to address the limitations of scenario specific disruption modeling, the latter being inadequate for the evolving and highly uncertain nature of cyber threats. As a proof of concept, we collected and analyzed 300.272 signals for 9 security incidents over 30 days. We used a multinomial Hidden Markov Model for disruption response function parameters inference and latent state filtering. Along with selected signals, these are used as features to train a Random Forest in a multistep prediction task, where we predict occurrence of cyberattacks on next time-step, reaching an F1-Score of .98 on next-hour horizon. Our hybrid model uses large-scale cyberattack telemetry as ground-truth, with the ability to produce both resilience metrics and cyberattack early warning. All algorithmic design choices were motivated by explainability and robustness trade-off.**

***Index Terms*—Network Security, Cyber-Resilience, Signal Processing, Machine Learning**

I. Introduction

This section presents the background for this research, key takeaways from the literature, our hypothesis and our proposed contributions.

A. Background

Cyber resilience is the capacity to deliver services despite cyberattacks [1], [2]. In a context of growing cyber threats [3], cyber resilience now parallels concerns for social responsibility and legal compliance. Primarily associated with power-grid cyber-physical systems (CPS) and defense systems, cyber resilience has emerged as a critical concern within many industry sectors and organizations [4]. Disruption modeling plays a key role in understanding and enhancing cyber resilience, as it helps in understanding system changes, service delivery, and safety conditions [1]. Model-based approaches can help in understanding the need for cyber resilience and the effects of different strategies [4], supporting both investment and technical design decisions.

B. Resilience modeling

A wide range of factors impact cyber resilience, including technological [1], [5], regulatory [6], organizational [7], [8], human [4], [9], and environmental factors [10]. Resilience modeling methods can be qualitative (conceptual frameworks and semi-quantitative methods) or quantitative (generic resilience metrics and structural modeling). Structural quantitative modeling methods are primarily used to model the resilience of telecommunications infrastructure networks [11], where they typically involve spatial representations of CPS systems [5].

In 2011, MITRE published a study discussing the foundations of quantitative cyber resilience modeling [12], later leading to a consensus [2], [13]–[15] on resilience being the "capacity to limit divergence from expected Quality of Service after a disruptive event" [2]. This definition is illustrated in Fig. 1 below, adapted from [2], [16].

In a simple yet comprehensive model, the system and its value function QoS show a drop at time t_i due to event $S_{(t_i)}$, and the system recovers to normal regime R_s at time t_j, reaching the desired state level $QoS = Z_1$. Two scenarios are shown: one where the QoS function drops below Z_3 (P_2 curve), down to the critical point where the "mission is compromised" [16], and one where it does not (P_1 curve).

Quantitative resilience modeling then boils down to expressing the curves P_1 and P_2, their integration delta noted U and Z_1 (or any other state) and P_1, P_2, noted V, as a discrete probabilistic function of S_{t_i} to minimise this area and the time of recovery t_{j-i}. In the context of this paper, QoS is considered impacted when an "incident" is declared, subsequent Z_i is unknown (see III-A for details).

C. Applied modeling and limitations

On the one hand, this concept has analogies in many well-known control-engineering fields, such as Feedback-Control Analysis (FCA) [17]–[21] or Fault-Tolerance Analysis (FTA) [2], [22]–[26]. On the other hand, as highlighted in [3], cyber resilience models cannot rely on explicit disruption scenario specification to make-up for the rapidly evolving threat landscape, when FCA/FTA methods focus on specific faults of and explicit disruption scenarios of (cyber) physical systems. Also, while structural information can be used to parameterize interactions between physical components, lack of information or nonlinearity can make this information difficult to process in the context of large networks.

979-8-3503-5477-5/24/$31.00 ©2024 IEEE

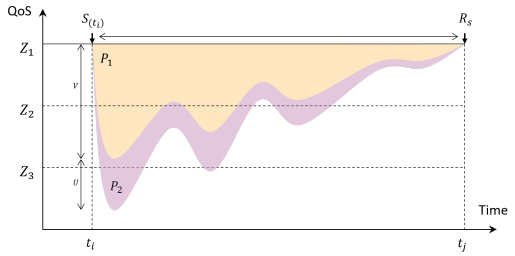

Fig. 1: The concept of resilience to disruption, inspired from [2]

D. State Space Models

State-Space Models (SSM) can make for FCA/FTA limitations, as they can avoid explicitly defining all disruption scenarii, based on known interaction logic [27]–[30]. An SSM is based on the assumption that the system's state can be described through a set of variables. At any time t_i, these variables can be expressed as a function of their value at t_0. Depending on the assumptions made, the SSM can assume multiple forms (continuous or discrete, linear or nonlinear dynamic system and observations), which makes it very flexible (see II for details). SSM are primarily used in the fields of signal processing and control engineering. Continuous time SSM such as the Kalman Filter (KF) [31] and its nonlinear equivalents [32]–[34] or their discrete linear counterpart the Hidden Markov Model, have shown great efficiency in processing joint distributions of unknown variables for filtering, prediction or smoothing problems, including for CPS.

E. Related work

In [27], an hybrid SSM is used to represent CPS resilience to cyberattacks. In [30], the authors modeled cyber resilience Supervisory Control and Data Acquisition (SCADA) systems using a multi-level hybrid SSM. In [29], a two level composite Markov model is used to analyze power-CPS resilience. More recently in [28], an implementation of Harmonic State Space (HSS) model is showcased for cyber-attack detection in photovoltaic farms. Those examples highlight successful applications of SSM modeling cyber resilience of CPS with known physical component interactions. However modeling cyberattack disruption response function for nonphysical cyber systems can still be difficult using these approaches, as the underlying parameters of the disruption function are unknown (see I-C).

F. Model Based Machine Learning

Such modelling constraints are studied in the Model Based Machine Learning (MBDL) [35] literature, or more recently Model based Deep-Learning (MoDL) [36], [37], where Dynamic Bayesian Networks [38] (including SSM) are combined with machine-learning algorithms. In MBML, the Model Based (MB) part of the algorithm is used for constraining the model to domain knowledge, and the Machine Learning (ML) is used to learn specific model parameters based on data observations. There are three main design patterns for these hybrid models [36]. *ML-Aided Optimizer*, where ML algorithms will be used to replace a non-linear operation within the original model, *Learned Optimizer*, where the MB is used to generate features for the ML model, and *Deep-Unfolding*, where each layer of a Deep Neural Network will be tasked with representing an operation of the initial model.

G. Research question, hypotheses and proposed contributions

While the academic literature provides hints on the applicability of SSM (and MBML) for addressing quantitative resilience modeling, none of the related work (see I-E) proposes a proof-of-concept MBML implementation of SSM for inferring network disruption response function in the context of cyberattacks on computer networks. We believe this approach should be investigated, as it would allow for embedding domain expert knowledge (see I-C) and leveraging big data collected from passed incidents to further refine QoS model (see I-E).

Hence the following research question: **Given a computer network undergoing a cyberattack, can we learn the network's response function to disruptions ?** Our hypotheses are that we can *(a.)* use an SSM to infer the latent state Z_T of the network by finding the response function structural parameters θ, and *(b.)* learn the nonstructural latent state representation of signals S_T to produce early warnings W_T of cyberattack based on ground-truth incidents I_T. Signals, response function parameters, latent states and early warnings can then be used to model resilience and optimise incident response.

In section II of this paper, we formalize the methods used to develop our model. In section III, we will provide a proof-of-concept on real-world data showcasing the efficiency of our model for inferring response function parameters, learning latent state representations and forecasting upcoming incidents. Finally, in section IV we will conclude with key takeaways and limits of this research, and present our future works.

II. Methods

This section formalizes the process which we developed: data collection (II-A) and feature selection (II-B), followed by an overview of SSM principles (II-C) and model selection (II-D). We will then present the algorithms used for retrieving parameters (II-E) and latent state sequence (II-F). Finally, we will explain subsequent resilience (II-G) and early warning (II-H) models.

A. Data collection and Analysis

We first collect data representing the network disruptions in discrete time. In our proof-of-concept (see section III) and for the sake of applicability to real-world scenarii, we chose to use Endpoint-Detection and Response (EDR) telemetry, as it contains the core disruption event information and is primarily used by analysts for incident response and forensics. The number of EDR alerts S_T for the horizon T is given by the sum of alerts s of all types s_x, time t_i and class c_k (1).

$$S_T = \sum_{t_i \subset T} s(s_x, t_i, c_k) \tag{1}$$

B. Feature selection

Alerts, also called signals, are then analysed for correlation (*e.g.* as multiple signals $s(s_x, t_i, c_k)$ tied to a single cyberattack could have a signal type s_x, time t_i and class c_k proximity). We also analyse serial-correlation, as multiple periodic weak signals of the same type s_x might self-correlate in a delay function over t_i. To select the most relevant signals for our model, we also use a regression between the signal time t_i for a given signal type s_x and the set of ground-truth incidents I_T. This regression will help assert if some signals can be used as direct estimators for the response function.

1) Correlation: For two signals $X = s(s_x, t_i, c_k)$ and $Y = s(s_y, t_i, c_n)$ with expected values μ_X, μ_Y and standard deviations σ_X, σ_Y (2), we measure the correlation as the Pearson coefficient. This correlation is also measured between X (or Y) and the incident occurrence, $I_{t_i} \in I_T$. We select the signals s_x that are the most correlated to the incidents within I_T.

$$\rho_{X,Y} = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y}, \text{ if } \sigma_X \sigma_Y > 0 \tag{2}$$

2) Serial-correlation: For the process $\{X_T\}$ representing the set of same-class signals $s_x \in S_T$, with a mean μ_t and variance σ_t^2, we measure the serial-correlation as the correlation of the signal at different time intervals t_i and t_{i+1}. See (3), where E is the expectation and $\overline{\text{X}}$ is the complex conjugation of X.

$$R_{X_T, X_T}(t_i, t_{i+1}) = E[X_{t_i} \overline{\text{X}}_{t_{i+1}}] \tag{3}$$

3) Logistic Regression: is measured between I_T and X_T, using the r^2 coefficient as accuracy measure, given a confidence interval of 95%. See (4) where I is the probability of seeing an incident at time t_i, $\sigma(X)$ the logistic regression of X, the probability of occurrence of s_x at time t_{i-1}, β_0 the intercept and β_1 the parameter coefficient.

$$P(I|X) = \sigma(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}} \tag{4}$$

Filtering the initial signal set collected S_T, we selected the best features as a set of signals F_{S_T}. This set contains signal classes F_{s_x} that present the best regression with I_T and best serial-correlation (*i.e.* lagging variables). Classes s_x that are the most correlated to classes F_{s_x} are filtered out from the set.

Given the steps of data-collection, analysis and feature selection described above, we now have a set of signal classes $F_S = \{F_{s_1}, \ldots, F_{s_x}\}$ that present the most potential for fitting a model to infer the system's response function to disruptions.

C. State-Space Model Principles

Next, disruption signals F_S and incidents I_T are fitted to the SSM (Fig. 2), which represents a system using a set of two differential equations: state (5) and output (6) [39]. These equations describe the system's evolution over time by defining its state, inputs, outputs, and the response function between them. The SSM can also be represented as a Factor Graph [40], as shown in Fig. 2.

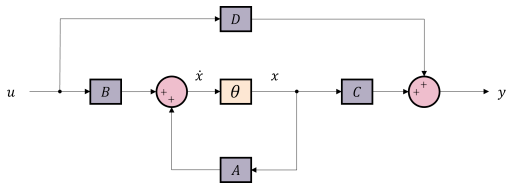

Fig. 2: Factor Graph representation of a Linear Time Invariant (LTI) State-Space Model (SSM)

1) SSM state equation: In (5), $x(t)$ is the state vector representing the internal state of the system at time t, $\dot{x}(t)$ is the derivative of the state vector, indicating how the state changes over time. A is the state matrix, which defines the relationship between the current state and its rate of change. $u(t)$ is the control vector, representing external influences on the system, B is the input matrix, which defines how the inputs affect the state.

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{5}$$

2) SSM output equation: In (6), $y(t)$ is the output vector, representing the observable outputs of the system. C is the output matrix, defining the relationship between the state and the outputs, D is the feed-forward matrix, describing the direct effect of the input on the output.

$$y(t) = Cx(t) + Du(t) \tag{6}$$

D. Hidden Markov Models

More specifically, we used a Hidden Markov Model (HMM) [38] to fit our signals, which is a discrete-linear (or non-linear depending on algorithmic implementation) SSM. This choice is motivated by [41], [42], showcasing applications of HMM in the context of cyber resilience. In [41], the overall exposition to cyberattacks of a computer network is inferred on externally observed events using an HMM (vulnerability releases in their case). While we share similarities with their approach, we only rely on network sensors, rather than external signals, as in [42], where the author uses HMM fitted network sensor data for detecting Advanced Persistent Threats (APT).

A Hidden Markov Model is a partially observed Markov model in which the hidden state, Z_i, evolves over time according to a Markov process [43], and each hidden state generates some observations s_i at each time step [44], as illustrated in Fig. 3. The key idea behind using an HMM is to infer the sequence of hidden states that most likely generated

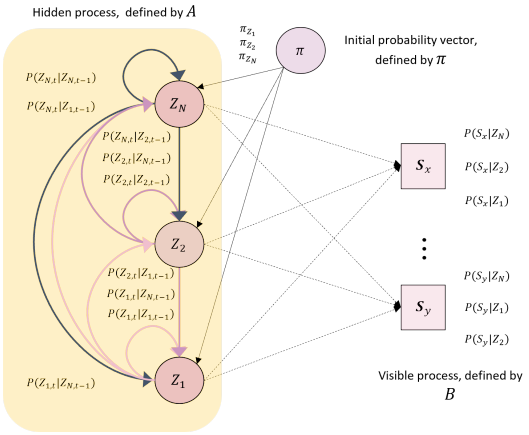

Fig. 3: General form of a Hidden Markov Model (HMM)

a given sequence of observable symbols (*i.e.* smoothing) using algorithms such as the Viterbi algorithm [45].

Formally, the HMM is characterised by a set of n possible states noted $\{Z_1, Z_2, \ldots, Z_n\}$, a sequence of states $Z_{i1}, Z_{i2}, \ldots, Z_{ik}$, a Markov-chain condition $P(Z_{ik}|Z_{i1}, Z_{i2}, \ldots, Z_{ik-1}) = P(Z_{ik}|P(Z_{ik-1})$ (*i.e* state probability only depends on the previous state) and a Hidden condition (*i.e.* states are not visible but emit signals of S observations $\{s_1, s_2, \ldots, s_M\}$). By convention, the HMM is formalized as $\theta = (A, B, \pi)$.

The transition probabilities are represented in the system's transition matrix, $A = (a_{ij})$ with $a_{ij} = P(Z_i|Z_j)$, the emission probabilities are represented in the observation matrix $B = (b_i(s_x))$ with $b_i(s_x) = P(s_x|Z_i)$. The initial probability state vector is noted $\pi = (\pi_i)$ with $\pi_i = P(Z_i)$. In our case, the set of states Z contains three states $\{Z_1, Z_2, Z_3\}$ representing the level of degradation of the network. The observable symbols are the classes within the selected signal classes F_S and initial state probabilities π are unknown.

E. Algorithm for parameters inference

To infer the model parameters $\theta = (A, B, \pi)$, we implemented the Baum-Welch algorithm [46] (which is a special case of the EM algorithm [47] for HMM), an iterative algorithm for obtaining (local) maximum likelihood or Maximum-A-Posteriori (MAP) estimates of the parameters. The iteration involves alternating between an Expectation (E) step, which creates a function for the expectation of the log-likelihood based on the current parameter estimate (7), and a maximisation (M) step, which computes parameters that maximise the expected log-likelihood obtained in the E step. These parameter estimates are then used to determine the distribution of the latent variables in the subsequent E step.

$$Q(\theta|\theta^{(t)}) = E_{Z|S,\theta^{(t)}}[\log L(\theta; S, Z)] \tag{7}$$

where $Q(\theta|\theta^{(t)})$ is the expected log-likelihood given the observed data S and the current parameter estimates $\theta^{(t)}$, $L(\theta; S, Z)$ is the log-likelihood function, which depends on both the observed data S and the latent state variables Z. $E_{Z|S,\theta^{(t)}}[\cdot]$ represents the expectation operator over the latent variables given the observed data S and the current parameter estimates $\theta^{(t)}$. In the M step, the algorithm maximises the expected log-likelihood obtained in the E step with respect to the parameter values, yielding the updated parameter estimates for the next iteration. See (8), where $\theta^{(t+1)}$ is the updated parameter estimates for the next iteration and $\arg\max_\theta$ represents the maximisation operation with respect to the parameter values θ.

$$\theta^{(t+1)} = \arg\max_{\theta} Q(\theta|\theta^{(t)}) \tag{8}$$

The algorithm continues to iterate between the E step and M step until convergence, where the parameter estimates no longer significantly changes between iterations. Therefore, it is guaranteed to converge if no target is specified.

F. Algorithm for state sequence inference

Having retrieved the model's parameters θ, we then used the Viterbi algorithm [45] to retrieve the most likely state-sequences that produces the observed sequence of symbols (this operation is also known as *smoothing*). The Viterbi algorithm computes the most likely sequence of hidden states $Z_T^* = \{Z_1^*, Z_2^*, \ldots, Z_N^*\}$ that generated the observable sequence given θ. The Viterbi algorithm has been extensively documented and analysed in the literature [45], [48], [49].

G. Resilience Modeling

In the context of resilience modeling, by inferring θ and Z_T^*, we retrieve very practical information regarding the network behaviour and its response function to disruptions. The θ parameters can be used as in [41], where the transition matrix A is used to compute the network's *Stability*, *Health*, *Anti-Fragility*, and its state *Dispersion*.

1) Stability: represents the capacity of the system to stay in its better state (Z_1). It represents the system's ability to remain in a positive state. Thus, a value is positive for the system and indicates a low transition potential (this metric is calculated from the system's transition matrix).

2) Health: which is the proportion of "healthy" (Z_1) states among all states, indicates whether the system is generally in a high resiliency state or not.

3) Anti-Fragility: which is the capacity of the system to get in a better state after being in a degraded health state, indicates the system's ability to escape from a sequence of degraded health states. An anti-fragility indicator close to 1 indicates strong anti-fragility.

4) Dispersion: which represents the dispersion of the health states among time. It indicates the tendency of the system to change state frequently, but without taking into account the direction of the change. A value close to 0 indicates that the system changes state very often and is therefore unstable on the considered period.

We supplemented these measures with three additional metrics based on the Z_T^* inference : Mean Time Between

Failures (MTBF), Mean Time To Failure (MTTF) and Mean Time To Recovery (MTTR). The metrics choice was motivated by [13]–[15], as those metrics are the most likely to be used by organisations for cyber resilience management, due to their presence in ISO-27001 and NIST-CSF standards [50].

5) Mean Time Between Failures (MTBF): is the average time between consecutive occurrences of Z_1 (start state) and Z_3 (end state) in the Z_T^* sequence. MTBF is calculated by finding all occurrences of Z_1 in the Z_T^* sequence and then calculating the time difference between each occurrence and the next occurrence of Z_3. The mean of these time differences represents the MTBF. See (9), where Z_{1i} and Z_{3j} are the occurrences and n is the total number of occurrences of Z_1. As MTBF represents the average time it takes for the system to fall into situation Z_3 from state Z_1, the higher this indicator, the better.

$$\text{MTBF} = \frac{1}{n}\sum_{i=1}^{n}(Z_{3_j} - Z_{1_i}) \tag{9}$$

6) Mean Time To Failure (MTTF): is the average time from the first occurrence of Z_1 (start state) to the first occurrence of Z_3 (end state) in the Z_T^* sequence. MTTF is computed by finding the first occurrence of Z_1 and then calculating the time difference between this occurrence of Z_1 and the next occurrence of Z3. See (10), where Z_{1i}, Z_{3j} are occurrences. As MTTF represents the average time it takes for the system to fail, we can find MTTF $= 0$ if the sequence has no Z_1 state over the sequence observed. The higher this indicator the better.

$$\text{MTTF} = Z_{3i} - Z_{1j},\ i, j = 0 \tag{10}$$

7) Mean Time To Recovery (MTTR): which is the average time from the first occurrence of Z_3 (end state) to the next occurrence of Z2 or Z_1 (start state) in the Z_T^* sequence. MTTR is calculated by finding all occurrences of Z_3 and then calculating the time difference between each occurrence of Z_3 and the next occurrence of a better state. See (11), where Z_{3i}, Z_{2j} (or Z_{1k}) are the indices of the occurrences in the Z_T^* sequence, and m is the total number of occurrences of Z_3. The MTTR indicates the average time it takes for the system to switch from a problematic Z_3 state to Z_1 state. The lower this metric, the faster the system recovers.

$$\text{MTTF} = \frac{1}{m}\sum_{i=1}^{m}(Z_{2j} \text{ or } Z_{1k} - Z_{3i}) \tag{11}$$

Given the metrics provided, we will now put forward a method for early warning of cyberattacks. To do so, we will learn latent representations Z_T^* of I_T, given F_S, in a supervised manner.

H. Early warnings

Having retrieved the θ parameters of the model, the Z_T^* most likely state sequence, given the signals F_S and set I_T of ground-truth incidents, we then *learned* latent state representations of upcoming incidents [51], [52] to produce early warnings. This problem has been addressed as a multi-step prediction problem, where we tried to predict accurately the existence of I_{t_j} given $Z_{t_i}^*$ and $F_{S_{x,t_i}}$, $t_i \leq t_j$. At first, $j = i$, *i.e.* we are predicting the current day class, then we iterated over $j = j + 1$. Intuitively, prediction on current time-step should be effective and forecasting precision should rapidly decrease as the Markov-Chain condition structurally limits the forecasting horizon as a function of the current state (see II-D).

We tested the following models: Random Forest (RF) [53], Long Short Term Memory (LSTM) [54], Gradient Boosting (XGB) [55], Support Vector Machine (SVM) [56], Multi-layer Perceptron (MLP) [57] and pure Logistic regression. Fine tuning the model with the best performance (Random-Forest) we have reached a 0,98 F1-Score [58] on predicting I, the probability of occurrence of I_{t_i} (*i.e.* incident starting next hour). The detailed results of the training and optimization of the models are presented in the next section.

III. Proof-of-concept

In this section, we provide a proof-of-concept for the methods described in the previous section. We first provide insight on the data collected (III-A), select the features (III-B), then fit the data to an HMM (III-C). Finally we computed the resilience metrics (III-D), and used supervised learning to produce early warnings (III-E).

A. Data Collection

We used data from a large computer network undergoing 9 different cybersecurity incidents within a 30 days period, with time-steps measured in hours. The collected EDR data (see II-A) represent 300.272 events, including 45 unique signal classes s_x, 3 level classes c_k, and 722 hour slots t_i. These various order of magnitude are expressed in Table I. We annotated the signals using a unique code representing their class s_x and c_k characteristics in the form of *Event_{level}{type}* (*e.g. Event_LowMalware*), and their unique timestamp[1].

TABLE I: Data collected

Descriptor	Value	Sample Value
Unique alerts (S_T)	300.272	Event_LowMalware_{timestamp}
Unique location (l_n)	63	France, Paris
Unique hosts (h_p)	13,762	Computer_{machineId}
Unique level (c_k)	3	{High, Medium, Low}
Unique type (s_x)	45	Malware
Unique hours (t_i)	722	14:00:00

B. Data Analysis and Feature Selection

Fig. 4 shows a sample plot used for exploratory data analysis. As discussed in section II-A, we analysed the Pearson-correlation, Serial-correlation and logistic regression of the signals, in order to select the features F_S.

[1]We have also collected location l_n and host h_p information for later use, see section IV.

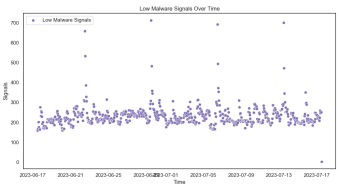

Fig. 4: Sample signal "Low Malware" over time

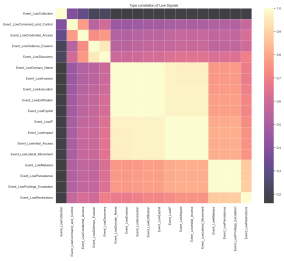

Fig. 5: Pearson correlation of same class c_k (low)

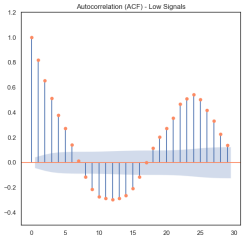

Fig. 6: Serial correlation of same class c_k (low)

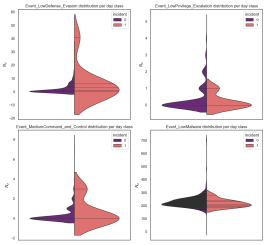

Fig. 7: Distribution of signals $s_{x,t}$ *vs.* incident I_t

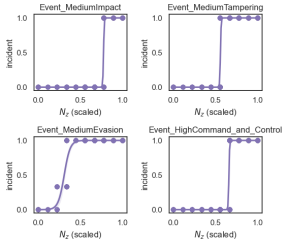

Fig. 8: Logistic Regression of signals N_s *vs.* Incident set I_T

First we analysed the signals at group level, *i.e.* correlation between signals of similar class c_k and intralevel Serial-correlation (*i.e.* lagging of same c_k signals, cumulative). Fig. 5 shows the intralevel correlation measure, Fig. 6 shows the intralevel serial-correlation. We also analysed signal wise distribution of the signals per hour class (*incident/no-incident*), as show in Fig. 7.

Then we studied the signal correlation, as intra/interlevel analysis only shows high level information about the class c_k. Fig. 9 shows the full results of signal correlation measure (including correlation with the incident set I_T. Serial-correlation analysis at class level showed sine-shaped, slowly decaying coefficients for *low* and *medium* classes, indicating these data presents a lagging of high order (several days in our case). At individual level, we found that "Restriction Bypass", "Malware", and "Domain Name" related signals presented the highest serial correlation. Signal-wise incident correlation using a logistic regression showed multiple signals are very likely to be good predictors for incident forecasting. Fig. 8, showing high r^2 scores for "Persistence", "Exfiltration" and "Malware" related signals. We selected the top ten features

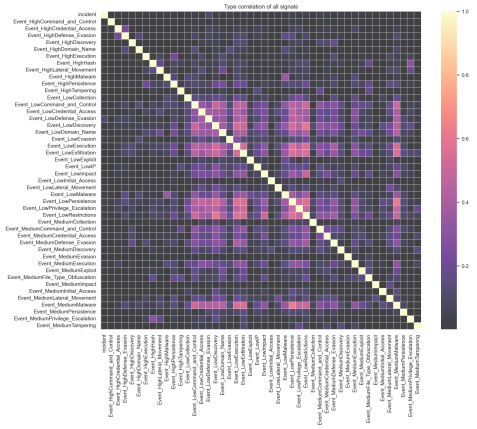

Fig. 9: Pearson correlation of all signals and incidents (absolute values)

following the methods detailed in section II-A. Those compose the F_{S_t} set and present the best lagging, regression and correlation combination.

C. Model Fitting

We fitted the signals to a multinomial Hidden Markov Model and applied the Baum-Welch algorithm to infer the model parameters θ, retrieving the transition matrix A, emission matrix B and prior π. These parameters are then used to infer the most-likely state sequences Z_T^* as described in II-F. The most-likely state sequences can then be plotted as shown in Fig. 10. This is the closest we can get to the initial problem statement model described in I-B. On this plot, we can see the state sequences representing the response function (described as QoS in [12]), and the associated timeline.

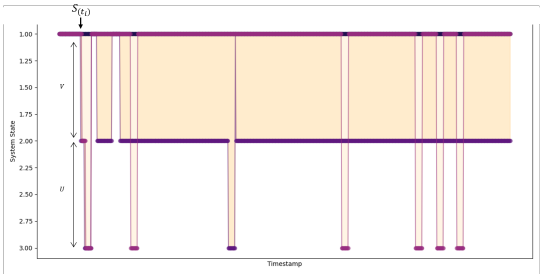

Fig. 10: Hidden states inferred, visualisation from [2]

D. Resilience Metrics

Based on the inferred states, we computed the resilience metrics as presented in section II-G. We computed the four metrics from [41] (Health, Stability, Anti-Fragility, Dispersion) and our three additional metrics (MTBF, MTTF, MTTR). The results are shown in Table II.

These indicators can also be measured for individual signals (*i.e.* with state sequences based on a single event sequence of observations). By doing so, resilience metrics show the system's response function to single disruption signals. One

one-hand, this information offers better precision for adjusting signal-specific incident response processes, on the other hand these data are less representative of the environment, since the inferred sequences Z_T^* and parameters θ are only based on this signal.

TABLE II: Network Resilience Metrics

Resilience Metric	Value (days)	Value (hours)
MTBF	3,26 (days)	78:15 (hh:mm)
MTTF	6,40 (days)	153:3 (hh:mm)
MTTR	3,32 (days)	79:20 (hh:mm)
Stability	0,5021	*n/a*
Health	0,4979	*n/a*
Anti-Fragility	0,3148	*n/a*
Dispersion	0,7216	*n/a*

E. Early warnings

Finally, having inferred model parameters state sequences Z_T^*, we trained multiple supervised learning algorithms to forecast incident occurrence I_j given filtered signals $F_{S_{x,i,\ldots,j-1}}$, passed states $Z^*_{i,\ldots,j-1}$ and current state Z_i^*. Without fine-tuning the models, we obtained good results on short horizon prediction, but saw the long horizon accuracy of all models drop significantly. This result was expected and is due to the Markov Condition (see section II-D).

We then fine-tuned the models for short term prediction (next hour). The best performance was achieved with a Random Forest, reaching an F1-Score of 0,9841. Random forests are an ensemble learning technique which builds a large number of decision trees. The class that the majority of the trees choose is the random forest's output for classification problems [53].The best performance was obtained using the following set of hyper-parameters: bootstrapping, no limit on tree depth, and node splitting based on square root of the total features. The minimum number of samples required to be at a leaf node is set to 2, and the minimum number of samples required to split an internal node is also set to 2. Our most accurate model uses 100 trees.

TABLE III: Classification performance for next-hour incident

Model	Loss	Accuracy	Precision	Recall	F1
LSTM	0.7941	68.75	0.9032	0.5895	0.7134
DNN	1.0092	73.61	0.9014	0.6737	0.7711
SVM	0.2666	90.28	0.988	0.8632	0.9213
XGB	0.1702	97.92	0.9893	0.9788	0.9840
Random Forest	**0.2791**	**97.92**	**0.9894**	**0.9789**	**0.9841**
Logistic Reg.	0.2688	88.89	0.9877	0.8421	0.9091

The performance of the other models for this time horizon can be seen in Table III. These results show that the Random Forest (and other models) presents significant capacity on predicting I_j.

IV. Conclusion and Perspectives

In this paper, after describing the current limitations of resilience modeling for large computer networks and having analysed the existing propositions in the control engineering literature, we have proposed a method for porting State-Space Modeling to cyber resilience modeling tasks using Model Based Machine Learning.

Our method involves: *(i.)* data collection and analysis for feature selection, using correlation, serial-correlation and regression; *(ii.)* Hidden Markov Model parameter inference; *(iii.)* latent state sequence inference; *(iv.)* computing seven resilience metrics, including four based on the works of [41] and three additional metrics based on [13]–[15]; and *(v.)* producing early warnings of upcoming incidents.

Our results suggest that the use of model-based machine learning is promising for modelling cyber attacks resilience. Testing several supervised learning algorithm, we obtained significant results (.95 F1) on predicting incident occurrence I_j using a Random Forest. However, the model's accuracy significantly drops for long-term predictions due to the Markov condition, limiting its utility for long-term forecasting.

However, many elements remain to be refined. For example, we used sequences derived from the model parameters to train the classifier. This means that, from a practical point of view, both algorithms need to be implemented. Ultimately, we would like to streamline training and inference in a "Learned Optimizer" end-to-end approach. Overall, the model's performance and scalability need further validation on larger and more diverse datasets over longer periods to ensure its robustness in different contexts.

V. Acknowledgments

The authors thank the anonymous reviewers for their valuable suggestions. The full data that support the findings of this study are available from the corresponding author upon reasonable request. The data are not publicly available due to their sensitive nature and them containing information that could compromise research participant security and privacy.

References

[1] C. Chittister and Y. Haimes, "The role of modeling in the resilience of cyberinfrastructure systems and preparedness for cyber intrusions," *Journal Of Homeland Security And Emergency Management*, vol. 8, 2011, article 000010220215477355157 7.

[2] J. Zalewski, S. Drager, W. McKeever, A. Kornecki, and B. Czejdo, "Modeling resiliency and its essential components for cyberphysical systems," *FedCSIS (Position Papers)*, pp. 107–114, 2015.

[3] M. François, P. E. Arduin, and M. Merad, "Classification of decision support systems for cybersecurity," in *15th Mediterranean Conference On Information Systems (MCIS) And The 6th Middle East & North Africa Conference On Digital Information Systems (MENACIS)*, 2023.

[4] J. Carias, L. Labaka, J. Sarriegi, and J. Hernantes, "An approach to the modeling of cyber resilience management," *2018 Global Internet Of Things Summit (GIoTS)*, pp. 1–6, 2018.

[5] C. Colman-Meixner, C. Develder, M. Tornatore, and B. Mukherjee, "A survey on resiliency techniques in cloud computing infrastructures and applications," *IEEE Communications Surveys & Tutorials*, vol. 18, pp. 2244–2281, 2016.

[6] V. Gisladottir, A. Ganin, J. Keisler, J. Kepner, and I. Linkov, "Resilience of cyber systems with over-and underregulation," *Risk Analysis*, vol. 37, pp. 1644–1651, 2017.

[7] R. Kleij and R. Leukfeldt, "Cyber resilient behavior: integrating human behavioral models and resilience engineering capabilities into cyber security," in *Advances In Human Factors In Cybersecurity: Proceedings Of The AHFE 2019 International Conference On Human Factors In Cybersecurity*, 2020, pp. 16–27.

[8] K. Hausken, "Cyber resilience in firms, organizations and societies," *Internet Of Things*, vol. 11, p. 100204, 2020.

[9] Z. Minchev, "Human factor role for cyber threats resilience," *Handbook Of Research On Civil Society And National Security In The Era Of Cyber Warfare*, pp. 377–402, 2016.

[10] G. Ahmadi-Assalemi, H. Al-Khateeb, G. Epiphaniou, and C. Maple, "Cyber resilience and incident response in smart cities: A systematic literature review," *Smart Cities*, vol. 3, pp. 894–927, 2020.

[11] S. Hosseini, K. Barker, and J. Ramirez-Marquez, "A review of definitions and measures of system resilience," *Reliability Engineering & System Safety*, vol. 145, pp. 47–61, 2016.

[12] R. Pietravalle and D. Lanz, *Resiliency Research Snapshot*. Bedford, Mass: The MITRE Corporation, 2011.

[13] D. Bodeau, R. Graubart, W. Heinbockel, and E. Laderman, "Cyber resiliency engineering aid–the updated cyber resiliency engineering framework and guidance on applying cyber resiliency techniques," *MTR140499R1, PR*, pp. 15–1334, 2015.

[14] D. Bodeau, R. Graubart, R. McQuaid, and J. Woodill, *Cyber resiliency metrics catalog*. Bedford, MA: The MITRE Corporation, 2018.

[15] D. Bodeau, R. Graubart, R. McQuaid, J. Woodill, and M. Ma, *Cyber Resiliency Metrics and Scoring in Practice*. MITRE CORP BEDFORD MA, 2018.

[16] M. Bishop, M. Carvalho, R. Ford, and L. Mayron, "Resilience is more than availability," in *Proceedings Of The 2011 New Security Paradigms Workshop*, 2011, pp. 95–104.

[17] J. Vempati, M. Thompson, and R. Dantu, "Feedback control for resiliency in face of an attack," in *Proceedings Of The 12th Annual Conference On Cyber And Information Security Research*, 2017, pp. 1–7.

[18] X. Zhou, Z. Gu, and F. Yang, "Resilient event-triggered output feedback control for load frequency control systems subject to cyber attacks," *IEEE Access*, vol. 7, pp. 58 951–58 958, 2019.

[19] L. Zha, R. Liao, J. Liu, X. Xie, E. Tian, and J. Cao, "Dynamic event-triggered output feedback control for networked systems subject to multiple cyber attacks," *IEEE Transactions On Cybernetics*, vol. 52, pp. 13 800–13 808, 2021.

[20] Z. Gu, J. Park, D. Yue, Z. Wu, and X. Xie, "Event-triggered security output feedback control for networked interconnected systems subject to cyberattacks," *IEEE Transactions On Systems, Man, And Cybernetics: Systems*, vol. 51, pp. 6197–6206, 2020.

[21] C. Zhai and W. Wu, "Designing continuous delay feedback control for lattice hydrodynamic model under cyberattacks and connected vehicle environment," *Communications In Nonlinear Science And Numerical Simulation*, vol. 95, p. 105667, 2021.

[22] P. Popov, "Models of reliability of fault-tolerant software under cyberattacks," in *2017 IEEE 28th International Symposium On Software Reliability Engineering (ISSRE)*, 2017, pp. 228–239.

[23] G. Franzè, W. Lucia, and F. Tedesco, "Resilient model predictive control for constrained cyber-physical systems subject to severe attacks on the communication channels," *IEEE Transactions On Automatic Control*, vol. 67, pp. 1822–1836, 2021.

[24] X. Sun, Y. Liu, and L. Deng, "Reliability assessment of cyber-physical distribution network based on the fault tree," *Renewable Energy*, vol. 155, pp. 1411–1424, 2020.

[25] V. Yadav, R. Youngblood, K. Le Blanc, J. Perschon, and R. Pitcher, "Fault-tree based prevention analysis of cyber-attack scenarios for pra applications," *2019 Annual Reliability And Maintainability Symposium (RAMS)*, pp. 1–7, 2019.

[26] V. Nagaraju, L. Fiondella, and T. Wandji, "A survey of fault and attack tree modeling and analysis for cyber risk management," in *2017 Ieee International Symposium On Technologies For Homeland Security (hst)*, 2017, pp. 1–6.

[27] J. Bradley and E. Atkins, "Toward continuous state–space regulation of coupled cyber–physical systems," *Proceedings Of The IEEE*, vol. 100, pp. 60–74, 2011.

[28] J. Zhang, L. Guo, and J. Ye, "Cyber-attack detection for photovoltaic farms based on power-electronics-enabled harmonic state space modeling," *IEEE Transactions On Smart Grid*, vol. 13, pp. 3929–3942, 2021.

[29] R. He, H. Xie, J. Deng, T. Feng, L. Lai, and M. Shahidehpour, "Reliability modeling and assessment of cyber space in cyber-physical power systems," *IEEE Transactions On Smart Grid*, vol. 11, pp. 3763–3773, 2020.

[30] L. Yang, X. Cao, and J. Li, "A new cyber security risk evaluation method for oil and gas scada based on factor state space," *Chaos, Solitons & Fractals*, vol. 89, pp. 203–209, 2016.

[31] D. Simon, "Kalman filtering," *Embedded Systems Programming*, vol. 14, pp. 72–79, 2001.

[32] K. Fujii, "Extended kalman filter," *Refernce Manual*, vol. 14, p. 41, 2013.

[33] E. Wan and R. Van Der Merwe, "The unscented kalman filter," *Kalman Filtering And Neural Networks*, pp. 221–280, 2001.

[34] F. Gustafsson, "Particle filter theory and practice with positioning applications," *IEEE Aerospace And Electronic Systems Magazine*, vol. 25, pp. 53–82, 2010.

[35] C. Bishop, "Model-based machine learning," *Philosophical Transactions Of The Royal Society A: Mathematical, Physical And Engineering Sciences*, vol. 371, 2013, article 20120222.

[36] N. Shlezinger, J. Whang, Y. Eldar, and A. Dimakis, "Model-based deep learning," *Proceedings Of The IEEE*, vol. 111, pp. 465–499, 2023.

[37] N. Shlezinger, Y. Eldar, and S. Boyd, "Model-based deep learning: On the intersection of deep learning and optimization," *IEEE Access*, vol. 10, pp. 115 384–115 398, 2022.

[38] Z. Ghahramani, "An introduction to hidden Markov models and Bayesian networks," *International Journal Of Pattern Recognition And Artificial Intelligence*, vol. 15, pp. 9–42, 2001.

[39] G. Szederkényi, R. Lakner, and M. Gerzson, *Intelligent control systems: an introduction with examples*. Springer Science & Business Media, 2006.

[40] H. Loeliger, "An introduction to factor graphs," *IEEE Signal Processing Magazine*, vol. 21, pp. 28–41, 2004.

[41] S. Chatterjee and S. Thekdi, "An iterative learning and inference approach to managing dynamic cyber vulnerabilities of complex systems," *Reliability Engineering & System Safety*, vol. 193, p. 106664, 2020.

[42] G. Brogi, *Real-time detection of Advanced Persistent Threats using Information Flow Tracking and Hidden Markov Models*. Conservatoire national des arts et metiers-CNAM, 2018.

[43] E. Dynkin and E. Dynkin, *Markov processes*. Springer, 1965.

[44] S. Eddy, "What is a hidden Markov model?" *Nature Biotechnology*, vol. 22, pp. 1315–1316, 2004.

[45] G. Forney, "The viterbi algorithm," *Proceedings Of The IEEE*, vol. 61, pp. 268–278, 1973.

[46] P. Baggenstoss, "A modified baum-welch algorithm for hidden Markov models with multiple observation spaces," *IEEE Transactions On Speech And Audio Processing*, vol. 9, pp. 411–416, 2001.

[47] S. Ng, T. Krishnan, and G. T. E. M. a. McLachlan, "Handbook of computational statistics: Concepts and methods," *pp.*, pp. 139–172, 2012.

[48] H. Lou, "Implementing the viterbi algorithm," *IEEE Signal Processing Magazine*, vol. 12, pp. 42–52, 1995.

[49] M. Ryan and G. Nudd, *The viterbi algorithm*. University of Warwick. Department of Computer Science, 1993.

[50] G. Culot, G. Nassimbeni, M. Podrecca, and M. Sartor, "The iso/iec 27001 information security management standard: literature review and theory-based research agenda," *The TQM Journal*, vol. 33, no. 7, pp. 76–105, 2021.

[51] J. Zhang, P. Porras, and J. Ullrich, "Gaussian process learning for cyber-attack early warning," *Statistical Analysis And Data Mining: The ASA Data Science Journal*, vol. 3, pp. 56–68, 2010.

[52] S. Kim and S. Hong, "Study on the development of early warning model for cyber attack," in *2011 International Conference On Information Science And Applications*, 2011, pp. 1–8.

[53] M. Segal, "Machine learning benchmarks and random forest regression," 2004.

[54] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: Lstm cells and network architectures," *Neural Computation*, vol. 31, pp. 1235–1270, 2019.

[55] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers In Neurorobotics*, vol. 7, p. 21, 2013.

[56] W. Noble, "What is a support vector machine?" *Nature Biotechnology*, vol. 24, pp. 1565–1567, 2006.

[57] M. Popescu, V. Balas, L. Perescu-Popescu, and N. Mastorakis, "Multilayer perceptron and neural networks," *WSEAS Transactions On Circuits And Systems*, vol. 8, pp. 579–588, 2009.

[58] Y. Sasaki, "The truth of the f-measure," *Teach Tutor Mater*, vol. 1, pp. 1–5, 2007.