2020 16th International Conference on Mobility, Sensing and Networking (MSN) | 978-1-7281-9916-0/20/$31.00 ©2020 IEEE | DOI: 10.1109/MSN50589.2020.00091

Machine Learning Enabled Secure Collection of Phasor Data in Smart Power Grid Networks

Wassila Lalouani and Mohamed Younis
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County,
Baltimore, Maryland, USA,
{lwassil1, younis}@umbc.edu

***Abstract*—In a smart power grid, phasor measurement devices provide critical status updates in order to enable stabilization of the grid against fluctuations in power demands and component failures. Particularly the trend is to employ a large number of phasor measurement units (PMUs) that are inter-networked through wireless links. We tackle the vulnerability of such a wireless PMU network to message replay and false data injection (FDI) attacks. We propose a novel approach for avoiding explicit data transmission through PMU measurements prediction. Our methodology is based on applying advanced machine learning techniques to forecast what values will be reported and associate a level of confidence in such prediction. Instead of sending the actual measurements, the PMU sends the difference between actual and predicted values along with the confidence level. By applying the same technique at the grid control or data aggregation unit, our approach implicitly makes such a unit aware of the actual measurements and enables authentication of the source of the transmission. Our approach is data-driven and varies over time; thus it increases the PMU network resilience against message replay and FDI attempts since the adversary's messages will violate the data prediction protocol. The effectiveness of approach is validated using datasets for the IEEE 14 and IEEE 39 bus systems and through security analysis.**

Keywords: Phasor networks, Replay attack, False data injection, Predictive sampling, Recurrent neural networks (RNN).

I. INTRODUCTION

Cyber-physical systems realize an integrated design approach of automatic control applications where distributed sensing, computation and actuation resources are interconnected through communication links to operate in a coordinated manner. The smart grid is a prominent example of these systems. The mission criticality of the power grid can make it a target of attacks [1]. To sustain stable operation of the grid, the load throughout the distribution network should be continuously monitored. A phasor measurement unit (PMU) monitors the AC power signals to enable detection of overload conditions. PMUs are equipped with GPS receivers to synchronize their readings so that a consistent snapshot of the state of the power grid could be obtained. Based on the system state autonomous (or operator triggered) reconfiguration of the grid could be applied to redistribute the energy supply and sustain load stability. For the grid to dynamically respond to load variation, the trend is to employ phasor units at a larger scale to enable comprehensive status monitoring and fine-grained control [2]-[4].

Given its important role, a PMU can be a favorable target for adversary's attacks. Most prominent examples of these attacks include impersonation, message replay, or false data injection. When erroneous, old, or tardy phasor data is provided, the grid could simply miss a serious overload condition or make wrong reconfiguration decisions that degrade the performance and even cause outages. With radio links being the most cost-effective choice for connecting PMUs with the control station, an eavesdropper can intercept transmissions and launch the aforementioned attacks. Although the network usually employs data integrity and authentication measures, e.g., by employing cryptographic primitives and key management protocols, the recent technological advances have boosted the perspective for cryptanalysis to succeed in breaking the system. In addition, asymmetric cryptosystems are complex and impose high computational overhead. Moreover, frequent updating of the encryption keys imposes communication overhead, in addition to the concern about the possible vulnerability of the rekeying process. Therefore, there is a need for an effective solution that stays robust overtime and does not involve major computation and communication overhead.

This paper opts to fill the technical gap and proposes an effective Machine Learning based Grid Protection (MLGP) system against replay and FDI attacks. The main idea is to not transmit the data but rather send indicators that could be interpreted by the controller to infer the data. This is not just a function mapping or special encoding of the PMU data, but rather a derived value based on prior PMU readings. In essence, we define a time series in a dynamic manner so that both the PMU and controller will have a consistent view of what value to expect next. Specifically, we employ a Long Short-Term Memory (LSTM) model that factors in prior PMU measurements and forecasts the next value. The PMU will then send the difference between the predicted and actual measurement along with the LSTM confidence in the prediction. By running the same LSTM model, the controller will generate a predicted value and assess confidence. By matching the prediction confidence to what the PMU sent, the controller further authenticates the source of the data, i.e., the PMU, and then uses the predicted value and adjustment sent by the PMU to retrieve the original data. Unlike existing schemes that detect FDI through data fusion, our solution prevents such an attack at the protocol level. To further deprive the adversary from knowing the actual confidence to launch replay and FDI attacks, our MLGP approach manipulates adaptively the certainty of the LSTM model. In fact, a replay in our context will simply be a retransmission of adjustment and confidence

978-1-7281-9916-0/20/$31.00 ©2020 IEEE
DOI 10.1109/MSN50589.2020.00091

level, where the latter will not match the controller's LSTM output and hence the replayed message will be rejected. In summary the paper makes the following contributions:

- Proposing a solution to guard the smart power grid against message replay and FDI attacks that target the PMU-controller interaction.
- Developing a protocol to collect PMU measurements without explicitly communicating the actual data. The protocol prevents an adversary from getting access to the data even if being able to successfully intercept and decode data packets.
- Employing advanced machine learning techniques to enable accurate measurements calculation without explicit transmission of the actual data.
- Validating the effectiveness of proposed techniques using datasets for the IEEE 14 and IEEE 39 bus systems and through security analysis.

The remainder of the paper is organized as follows. The next section discusses the related work. Section III goes over our system and attack models. Section IV presents an overview of MLGP, highlighting the employed defense strategy and outlining the communication protocol. The detailed design of MLGP is provided in Section V. Section VI analyzes the resilience of MLGP to security attacks and discusses the validation environment and performance results. Finally, the paper is concluded in Section VII.

II. Related Work

Work can be categorized based on the attack model into authentication, replay, and data forgery; and based on the specific part of the network into transmission/distribution network, and home area network, including advanced metering infrastructure (AMI) [5][6][7]. Also some work investigates attacks targeting time synchrony within the grid [8]. Given the scope of the contribution, this section focuses only on related work on FDI and message replay attacks that could impact state estimation in the power grid.

False data injection: FDI attacks can be classified as observable or unobservable depending on whether it can be detected by a system-wide consistency check [9]. Basically, if the maliciously injected data causes a major variation in certain state variables or contradicts with other measurements, the attack will be detected and sometimes even tolerated. Unobservable FDI attacks are those that manipulate the data such that no anomaly could be detected. Several methods have been developed to mitigate FDI attacks using state estimation. *Kosut et al.* [10] have proposed a heuristic based on Bayesian formulation and prior information about the likely state estimation to find an undetectable FDI. Meanwhile, *Liu et al.* [11] correlate historical data and the sparsity of the attack attributes to detect FDI attempts. *Chaojun et al* [12] use Kullback–Leibler distance as a means to detect deviations from the probability distribution of the historical data and point out potential FDI attacks. Machine learning techniques have also been applied for bad data detection and malicious FDI identification in smart grids using a variety of classifiers such as SVM and deep-learning models [13]-[16]. *Hao et al* [17] exploit sparse principal component analysis and approximation to detect unobservable FDI using blind topology estimation [18]. Our proposed MLGP approach limits the adversary's ability in launching FDI attacks, and thus constitutes an attack prevention rather than a detecting mechanism.

Authentication and replay: Countering replay attacks conventionally have been through appending a timestamp and an authentication code to each data message. Nonetheless, to avoid the high computational overhead and/or mitigate the risk of cryptanalysis, application centric methodologies have been pursued in the realm of smart power grids. The underlying principle for these methodologies is to make sure that the message replay will become an observable FDI that can be detected by state estimation operators. Multiple published schemes achieve such a goal by adding random noise to the input in order to amplify the effect of replay attacks [19][20]. However, such noise often degrades the system performance. Other work perturbs the input slightly, in order to limit the negative impact on the control system [21]. Some of the approaches add the noise all the time [19][21], while others do so intermittently [20]. The latter methodology tries to minimize the impact on the system and realizes that an effective replay attack has to be through a sequence of messages and not only one message. Thus, by intermittent noise injection the adversary is being deprived from having such a sequence. Irita and T. Namerikawa [22] have tried to strike a better balance and modeled the problem as a cooperative game between attack protection and system performance. However, noise injection based approaches still have a low attack detection rate. MLGP pursues a more effective strategy by obscuring the data based on a recurrent network and thus will not degrade the system performance. To our knowledge such a defense mechanism does not exist in the literature.

III. System and Attack Models

A. System Architecture

An electric power grid interconnects energy sources, e.g., power plants, and renewables, e.g., wind farms, to consumers. To enable effective management under varying power generation rate and load, a wide area monitoring system (WAMS) is established where PMUs are deployed at various transmission buses to monitor the frequency, amplitude and phase of the voltage/current and reporting the measurements to a control station, e.g., a SCADA. By synchronizing the clocks of the PMUs, the control station can correlate the data of the individual PMUs to estimate the state of the grid and determine whether reconfiguration actions, e.g., cutting some load are warranted to sustain stability. To enable fine-grained control, micro-level PMU are employed at the edge of the grid. PMUs can be connected to the control station using wired or wireless links; the latter has become more viable recently due to its lower cost and rapid deployment. We also note that a hierarchical WAMS topology could be pursued to enable scalability, where concentrator nodes are deployed to collect measurements from a subset of PMU and then sends the aggregated data to the control station. In that case, MLGP will be deployed to protect the interaction between a PMU and the corresponding data

concentrator node. Finally, MLGP does not require or prevent the grid network from encrypting the payload of data packets as an additional level of protection; MLGP provides protection even when the adversary is able to break the cryptosystem.

As pointed out, the control station uses the PMU measurements, e.g. voltage amplitude and phase, to assess the current state of the grid. The following equation estimates the state vector *x*:

$$z = H(x) + e \quad (1)$$

where *z* and *x* are *m*-dimensional vectors, *H* is a set of nonlinear functions relating the states to measurements, and *e* is the measurement error due to inaccuracy. In an over-determined case where the number of observations *m* exceeds the number of variables *n*, the state vector *x* is determined using weighted least square (WLS) optimization based on a residual function:

$$\hat{x} = \arg\min\big(z - H(x)\big)^T W(z - H(x)) \quad (2)$$

W is defined as diag of $\sigma_1^{-2}, \ldots, \sigma_{N_m}^{-2}$, σ_i^{-2} is the variance of i^{th} measurement. Erroneous measurements, due to attacks or PMU faults, could be detected based on residues if:

$$\big(z - H(x)\big)^T W\big(z - H(x)\big) > \tau \quad (3)$$

where τ is a threshold. The error α in z is due to a bias γ to the state vector x, i.e., $z = H(x) + \alpha + e = H(x + \gamma) + e$. Often the error can be detected by consistency checks or just diminished by the WLS optimization. However, if no detector can distinguish x from $x + \gamma$, the error (attack) becomes *unobservable* and has the form $\alpha = H(\gamma)$.

B. Attack Model

The objective of the adversary is to mislead the control station by providing wrong PMU measurements so that unwarranted reconfiguration actions are taken. To achieve this objective, the adversary opts to alter the PMU data maliciously. We consider two threat scenarios. In the first, the adversary deploys malicious nodes to replicate the functionality of PMUs, and impersonate existing nodes in the grid network. The second scenario occurs when the adversary eavesdrops on the communication link between the PMU and control station, aiming at intercepting transmissions and repeating them at a later time. The following describes the specific attacks that MLGP tackles:

False Data Injection (FDI): This attack is launched by mimicking PMU transmissions but with malicious data payload. A rogue node will impersonate a PMU to provide measurements so that the estimated state x becomes $x + \gamma$. FDI can be classified as observable and unobservable. The former is detectable by the system through consistency checks or tolerable by the WLS optimization; meanwhile the latter is subtle. In order to launch unobservable FDI attacks, an adversary often needs to know the grid topology and measurements made by multiple PMUs [18].

Message Replay: this attack is launched by retransmission of a valid data message from a PMU at a later time. As a real-time control system, the power grid needs to base actions of fresh measurements. By retransmitting old PMU messages, the attacker opts to get the controller to consider an old state and make reconfiguration decisions that can harm the systems, e.g., unjustly shedding some load and causing inconvenience, causing short circuits on some lines, etc. It has been shown in [22] that multiple consecutive messages have to be replayed in order for the attack to be impactful.

IV. Attack Defense Strategy

The principal strategy applied by MLGP for defending against FDI and message replay attacks is to avoid the transmission of the raw measurements. MLGP does so by employing a novel PMU measurement prediction mechanism based on advanced machine learning models. The key idea is to determine a base value for the PMU measurement; such baseline value is mutually agreed upon by, and independently generated, by both the PMU and control station. Then, only the difference between the baseline and the actual data values is sent out to the control station. Such a baseline is not constant and varies overtime; in fact it is regenerated for every PMU transmission. Basically, MLGP applies a machine learning model where the most recent *n* data values, i.e., PMU measurements, are used to forecast what could be the next value. Specifically, MLGP employs time series based machine learning, namely LSTM, to assess the utility of the previous PMU measurements and predict the next measurement. The predicted value is used to calculate what the PMU sends to the control station. The machine learning model will also compute a confidence level, Ψ_{PMU}, for the prediction; the value of Ψ_{PMU} will also be shared with the control station and will be used to authenticate the source PMU and confirm the integrity of the data, as we explain below.

Both the PMU and the control station have the same learning function. In addition, the controller already has previous PMU measurements. In other words, the prediction mechanism will be applied at the control station using the same LSTM model and the same input, and thus the raw PMU measurement will be perfectly recovered. Fundamentally the PMU measurements should be time-stamped so that they can be correctly ordered and processed. GPS receivers are usually deployed at both the PMU and the control station to ensure time synchronization. A

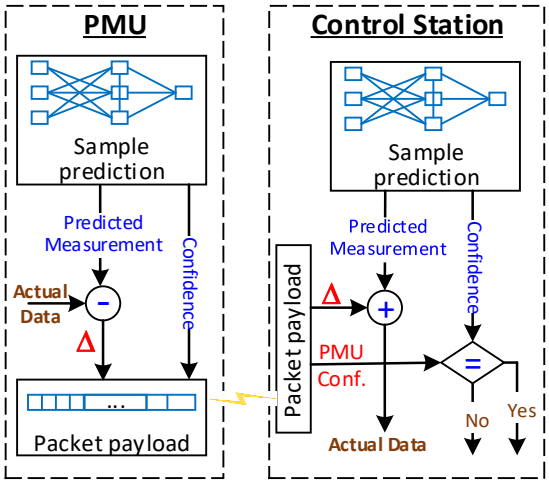

Fig. 2. A high level block diagram of the MLGP approach.

missed sample (measurement) is attributed to communication failure and appropriate reaction from the controller is required, e.g., request retransmission. The LSTM model will also generate $\Psi_{control}$ to reflect the confidence level in the prediction; by matching $\Psi_{control}$ to Ψ_{PMU} the control station ensures that the received data packet indeed was sent by the PMU and the integrity of the data was preserved. Fig. 1 provides an overview of the steps at the PMU and the control station.

MLGP safeguards the control station against adversarial attempts to provide wrong or outdated data that could lead to inappropriate actions and disturb the grid operation. First, eavesdropping on all PMU transmissions still will not allow the adversary to know the data since the actual measurements are not included in the message payload. Consequently, MLGP prevents the adversary from performing reverse engineering and uncovering the topology of the grid network [23]; implicitly, this obscures any possibility for generating unobservable FDI. Moreover, replaying an old PMU message or forming a packet with randomly picked data value and Ψ_{PMU}, will be detected by MLGP since Ψ_{PMU} and $\Psi_{control}$ will not match. In fact, such an attempt will most probably constitute an observable FDI even if it is undetected by the state estimator.

Finally, MLGP thwarts any adversarial attempt to model the data sharing protocol and then launch an FDI attack that mimics such a protocol. As a means for confirming the integrity of the data and authenticating the PMU, MLGP employs the confidence of the LSTM model, namely Ψ_{PMU}, on a per packet basis. To ensure that the statistical distribution of the Ψ_{PMU} values cannot be exploited by an adversary to generate undetectable FDI and message replay attacks, MLGP adjusts the LSTM model, specifically using dropout, in order to introduce indeterminism to the Ψ_{PMU} distribution. Dropout is a regularization method that is often pursued in machine learning to avoid overfitting the training data. MLGP conducts regularization by probabilistically updating recurrent activations of LSTM units to confuse the adversary.

V. Detailed MLGP Approach

MLGP mainly consists of two modules, namely, data prediction (baseline calculation) and confidence level estimation, as explained in the balance of this section.

A. Sample Prediction Technique

The objective of MLGP is to enable transmission of PMU measurements while obscuring them from the adversary. Without knowing the data, fundamentally an eavesdropper cannot generate the power system topology and cannot launch unobservable FDI and message replay attacks. As mentioned in the previous section, MLGP predicts the next PMU measurement and only sends the difference between the predicted and actual values. Basically MLGP leverages the prediction capabilities of recurrent neural networks (RNN). The idea is to see the transmitted data as a time series. In our case, we use the mean square error as a loss function in order to increase the accuracy approximation of the prediction compared to the real measurement. Note that in case of deterministic error in the model, the gradient captures all parameter changes to fit the loss function. Thus, the model error cannot be predictable and by knowing only the difference between real measurements and predicted ones, an adversary can never learn the error pattern of the model. Particularly, we employ an LSTM network, which is a special type of RNN that is trained using backpropagation through time and overcomes the vanishing gradient issue of ordinary recurrent networks.

Instead of neurons, LSTM networks employ memory blocks known as gates. Gates operate upon an input sequence and use a specific activation function to control whether a prior input is to be factored in (remembered) or be deemed outdated (forgotten). This in effect will control changes in the LSTM state and consequently the information flow. Those conditional gates can be either forget, input or output gates. The forget gates decide which information to throw away from the block. Input Gate decides which values from the input to update the memory state. Output Gate decides what to output based on input and the memory of the block. Each gate has weights that are learned during the training process. We note that the predicted sample in essence does not affect the control operation of the grid, since MLGP uses it as a baseline and sends only its deviation from the actual value so that the controller could retrieve the correct measurement. Unlike existing work on sample predicting in the realm of smart grids, the use of LSTM is deemed quite suitable for MLGP as the length of the sequence, i.e., the number of previous samples n, can be controlled, and can be exploited as a means for trade-off, where n can be small to limit the required buffering space when the data sample size is large. Fig. 2 shows the architecture of the LSTM that we employ to predict PMU measurements. The following operations are performed for each sample, i.e., at each time epoch t [24].

$$i_t = \sigma(W_i . [H_{t-1}, x_t] + b_i) \quad (1)$$

$$f_t = \sigma(W_f . [H_{t-1}, x_t] + b_f) \quad (2)$$

$$\check{C}_t = \tanh(W_{\check{C}} . [H_{t-1}, x_t] + b_{\check{C}}) \quad (3)$$

$$C_t = C_{t-1} . f_t + i_t . \check{C}_t \quad (4)$$

$$o_t = \sigma(W_0 [H_{t-1}, x_t] + b_0) \quad (5)$$

$$h_t = o_t . o(c_t) \quad (6)$$

where:

- x_t is the input sequence (vector) for the LSTM unit in at time epoch t, which constitutes the n^{th} previous samples.
- C_t is the new LSTM state vector at time epoch t, and depends on prior state and input.

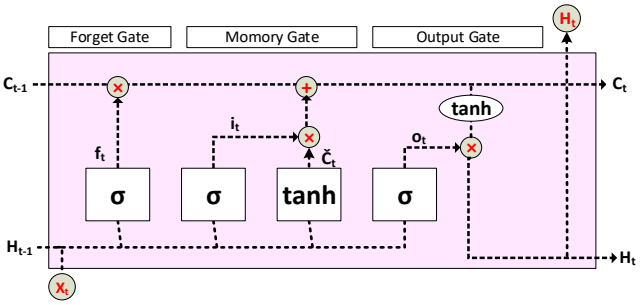

Fig. 2. The detailed design of the LSTM cell. A LSTM model consists of multiple cells; in MLGP the use of four cells provided the best results when considering PMU data.

- H_t is the output for cell at time epoch *t*, which is dependent on C_{t-1} and x_t.
- i_t , f_t , o_t are input, forget, and output gate sub-tensors for time epoch *t*.
- $\tilde{C}_t$ is a new cell candidate in an input sequence at *t*.
- b is bias for appropriate input sub-tensor.
- $W_i, W_o,$ and $W_{\tilde{C}}$ are the weight vectors for the four gates, respectively. They are determined through training.

First, the forget gate determines what information is to be discarded based on H_{t-1} and the input vector, i.e., samples, and generates the output f_t. Then, the memory gate decides on what to store in the cell state. The input gate subtensor calculates the new value of *i*, while the *tanh* layer creates a vector of new candidate values, $\tilde{C}_t$, to be included in the state. The new value of the state is the aggregation of the previous state multiplied by the forget gate output and add $i_t \times \tilde{C}_t$. We need to decide the output based on filtered cell state. To do so, we use a sigmoid layer to determine the parts of the cell state to output. Then, we pass the cell state through *tanh* and multiply it by the output of the sigmoid gate, so that we only output the relevant parts.

Both the PMU and the controller need to have the same LSTM architecture. The PMU collects measurements (samples) periodically. For each period *t*, each PMU forecasts the sample using its LSTM based on the previous *n* samples in the time series. The PMU reports only the difference between the actual measurement at time *t* and the predicted sample. By running the same LSTM, the controller will predict the same sample value and can thus calculate the actual PMU measurement. The LSTM is PMU-dependent and needs to be trained accordingly. At the time of PMU deployment, we collect samples over a period of time. The collected samples serve as a training dataset where each sample is labeled with the corresponding timestamp. During testing, the LSTM predicts data sample s_k^t at time *t* for device *k* using s_k^{t-n} …., s_k^{t-1}. We again note that the accuracy of the predicted sample is not an issue since the PMU will provide correction (deviation) based on the actual measurements; therefore, the value of *n* can be exploited to cope with any storage constraints.

B. Confidence Level Estimation

As mentioned above, MLGP obscures the data from the adversary by transmitting customized, rather than actual, data values such that the control station can still infer the actual PMU measurements. Although the control station would know the predicted sample through the LSTM model, it does not know the actual data, and thus it is conceivable that an adversary replays an old message or introduces a new one with wrong values. Therefore, to tackle the possibility of impersonating the PMU and launching message replay and FDI attacks, MLGP generates a confidence level, Ψ, that is specific to the employed machine learning model and to the particular time instance. By matching the value of Ψ sent by the PMU to what is generated by the control station, MLGP implicitly authenticates the PMU and validates the integrity of the data. The advantages of such an approach are: (1) it does not impose additional overhead since the machine learning model is already processing the data, (2) it is lightweight compared to complex asymmetric cryptographic systems, and (3) it is data driven and varies per transmission. Nonetheless, two issues are to be addressed.

The first issue is how to generate the confidence using the employed LSTM model. Basically, deep learning models, such as LSTM, estimate parameters and continually improve their prediction without any guarantees of the absence of randomized guesses in the output. Multiple techniques have been proposed in the literature for assessing information uncertainty in deep learning models [25]. Uncertainty in deep learning can be aleatoric and epistemic based on whether it is caused by noisy data and model parameters, respectively. Both aleatoric and epistemic uncertainty can be used in MLGP to associate confidence for the LSTM output, i.e., the predicted value of PMU measurements. Basically, MLGP introduces dropout to be applied to the recurrent state on the LSTM cell.

Dropout is a regularization function, i.e., the fraction of the units to drop for the linear transformation of the recurrent state. In the case of MLGP, it can be seen as masks applied to the hidden state in each LSTM cell to exclude it from activation. To illustrate, let us consider a cell that is activated by a d-dimensional vector *h*. Applying a dropout with probability *p* will prevent some activations in *h* from happening during training, i.e., $h_{train}=m \odot h$, where $\odot$ is the element-wise product, and *m* is a binary mask vector of size *d* [26]. Each element in *m* is drawn independently from a Bernoulli distribution, *Bernoulli*(*p*). During testing, all units are retained but their activations are weighted by *p*: $h_{test}=ph$.

Given different values for masks according to a probability distribution, some of the input will not have significant importance. Thus, the dropout allows the LSTM model to generate variable output. The range of the predicted outputs or the variance on the output constitutes the confidence while the mean reflects the prediction. Applying the dropout to the input of the cascading recurrent layers, rather than the model parameters enables controlling the confidence according to input, i.e., PMU measurements. In essence, the dropout can be varied to impose randomness to the predicted PMU measurements by MLGP so that the adversary faces a greater challenge in guessing the probability distribution matching the prediction and the confidence of the LSTM model. It is possible to vary the confidence by manipulating the historical joint distribution of the data and the confidence.

The second issue is how to prevent an eavesdropper from modeling the association between the transmitted value, Δ, i.e., difference between actual and estimated measurements, and the confidence level, Ψ, generated by the LSTM model. MLGP tackles this issue through careful setting the precision of Δ and adaptively varying Ψ based on the perceived adversary's gained knowledge overtime. The effect of precision is quite intuitive. Basically, if the adversary is to guess Δ up to 5th decimal digit is way less feasible than doing so for up to the 1st decimal digit. Thus, if the precision of the PMU measurements is high and the LSTM model handles prediction of fine-grained precision, the distribution of Δ becomes almost continuous and the probability that the adversary guesses the correct value

within an FDI attack massively diminishes, if not reaching zero. However, increasing the precision may also cause rejection of legitimate PMU messages due to possible rounding and truncation during computation. Thus, a trade-off is warranted. Both the PMU and controller need to use the same precision rounding. Our experiments suggest reasonable precision ranges.

On the other hand, MLGP adaptively varies Ψ by dynamically adjusting the dropout. As discussed above, the dropout can be seen as the scaled random vector from *Bernoulli*(*p*) with a dropout rate corresponding to *p*. The vector has the same dimensions as the input of the hidden layers and it constitutes the mask applied to the input of each recurrent cell. The random vector is scaled by *1/(1 - dropout rate)*. In each layer, the length of the random vector is equal to the number of hidden units in the preceding dropout. MLGP adapts the LSTM model to introduce noise based on the multivariate distribution of Δ and Ψ over time. To factor in the gained knowledge by the eavesdropper, we use a self-learning mechanism, where for each high likelihood occurrence of the pair (Δ, Ψ); the dropout is rescaled accordingly. Conventionally, the dropout is used to improve the confidence of machine learning models; yet in MLGP it is pursued as a means to confuse the adversary. Thus, we employ periodical manipulation of the dropout in order to prevent an adversary from picking the best action with some predetermined probability. A pseudo code summary of MLGP is provided in Fig. 3 for the PMU side; similar steps apply at the control station with different order.

VI. Validation Experiments

To validate the effectiveness of MLGP, we have used both synthetic and published datasets. We have first generated PMU measurements for IEEE 14-bus using Matpower [27] for a load profile from New York ISO [28]. In addition, we have the dataset of M. Naglic [29] which contains PMU measurements of all ten generators of IEEE 39-bus transmission system. The dataset was generated using the RTDS power system emulator and GTNETx2 based PMUs. The dataset contains 5197 PMU measurements; we used the voltage magnitude for our predictions mechanism. For both datasets, we have studied how MLGP performs under different attack scenarios. To launch an attack with valid (Δ, Ψ) pairs, we have considered the cases where the adversary applies: (i) a deterministic learning mechanism, and (ii) probabilistic modeling. For the former, we assume that the adversary is employing a recurrent time series learning to predict a (Δ, Ψ) pair that will not be rejected by the control station. When the adversary applies probabilistic modeling, we formulate the adversary prediction as a kernel distribution estimation problem. The kernel distribution estimation is a non-parametric distribution and does not make any assumption about the distribution of the data. In both cases, we assess the effect of the dropout on the knowledge of the adversary. The validation experiments and performance results are discussed in the balance of this section.

1. Prepare the dataset and normalize it
2. Train the model
3. Extract the weights of the network
4. **For each** sampling time:
5. Predict the sample using the LSTM.
6. Determine the variance, Ψ_{PMU}, of the predicted PMU measurements prediction using LSTM.
7. Δ = actual measurement – predicted data
8. Send Δ and Ψ_{PMU} to the control station
9. Estimate the probability distribution of the dataset
10. Rescale the dropout
11. **End for**

Fig. 3. Pseudo code summary of MLGP (on the PMU side).

A. Experiment Setup

In order to study the performance of MLGP, each dataset is divided into two subsets for supporting the training and test phases. Assuming the current time is *t*, we want to predict the value at the next time epoch (*t*+1) given the measurements for current and *n* previous time epochs. The LSTM is trained for 100 epochs with batch size of 1024, reflecting measurements of a single PMU. After training LSTM, we extract the parameters of the model, specifically the weights $W_i, W_o,$ and $W_{\tilde{c}}$. We have experimented with the number of LSTM cells and found the incorporation of 4 cells does yield the best results. We have also evaluated the effect of *n*, and observed no significant variation of the prediction accuracy when more than three samples are factored in. Therefore, the reported results in this section are based on *n*=3, i.e., the LSTM predicts s_k^{t+1} for a device *k* using ($s_k^{t-2}, s_k^{t-1}, s_k^t$). The simulation opts to capture the effect of:

- *Dropout rate*: This reflects the probability of introducing noise to the input data, and impacts the confidence of the LSTM model. It is used to adapt the confidence level according to the perceived adversary's gained knowledge.
- *Tolerable inaccuracy*: This indicates how different the predicted measurements at the PMU and control station, and

Table I: The adversary success rate for various data precisions.

	Tolerance	10^{-6}	10^{-5}	10^{-4}	10^{-2}	10^{-1}	0
IEEE 14-bus	Δ	0	0	0	0	0	0.48
	Ψ	0	0	0	0	0.005	0.67
	(Δ , Ψ)	0	0	0	0	0	0.32070
IEEE 39-bus	Δ	0	0	0	0	0	0.49
	Ψ	0	0	0	0	0	0.89
	(Δ , Ψ)	0	0	0	0	0	0.44

Table II: Effect of dropout threshold on the adversary success rate.

	Dropout	0.2		0.4		0.6		0.8	
	Precision	0	2	0	2	0	2	0	2
IEEE 14-bus	Δ	0.48	0	0.48	0	0.48	0	0.48	0
	Ψ	0.58	0	0.55	0	0.81	0	0.99	0.14
	(Δ , Ψ)	0.27	0	0.26	0	0.38	0	0.48	0
IEEE 39-bus	Δ	0.497	0	0.489	0	0.484	0	0.479	0
	Ψ	0.75	0	0.89	0	0.75	0	0.84	0
	(Δ , Ψ)	0.35	0	0.43	0	0.35	0	0.4	0

is in essence the precision of the data, i.e., Δ, in MLGP. It also affects the error in the adversary learning process.

We measure the effectiveness of MLGP in terms of the following metrics:

- *The accuracy of predicting* Δ: It reflects the adversary's ability in guessing a correct Δ when applying time series learning-based prediction.
- *The accuracy of predicting* Ψ: This assesses the probability of the adversary's success in estimating the next confidence level by applying a time series based learning model.
- *The accuracy of predicting the next (Δ , Ψ) pair*: This metric opts to capture the possibility of launching successful replay attacks that is undetected by state estimation.
- *Probabilistic multivariate distribution*: This metric captures the probabilistic auto correlation between the probability distribution of Δ and Ψ.

B. Performance Results

MLGL performance: For the case when an adversary applies deterministic learning over the transmitted data, we have used an LSTM network with the same configuration of that of MLGP. Table I reports the achieved adversarial success ratio while varying the precision between 6 and 0 decimal digits. The table reports the success rate of the adversary in guessing Δ, Ψ, and both. Overall, using a very low precision enables the adversary to experience some success in predicting Δ and Ψ. Obviously, a precision of zero and one decimal digits is meaningless in a smart grid application. The results confirm the effectiveness of MLGP since with just a precision of two digits, the prospect of successful attacks completely vanishes. The adversary also has very minute chances to accurately predict both Δ and Ψ. Thus, even if the adversary tries to apply an unobservable replay attack due to the correlation over time, the attack cannot succeed in modelling the power flow and thus it can be discovered in any static state estimation techniques.

Table II shows the performance while varying the dropout rate from 0.2 to 0.8 while fixing the precision to 0 and 2 decimals. With two-digit precision, the adversary becomes more successful when the dropout rate increases. However, by manipulating both the precision and dropout, it is strictly impossible for the adversary to infer the (Δ , Ψ) pair of a transmitted message. We see that the adversary's accuracy in predicting Δ is not affected by the dropout. This is not a surprise since principally the dropout affects the confidence, i.e., certainty in the estimated PMU measurements, rather than the output of the LSTM model. While Table II indicates that a high precision may suffice in ruining adversarial attempts, the dropout is essential since the confidence is used in authenticating the source of the data message, meaning that the message is indeed sent by the PMU.

To better capture the effect of dropout, Fig. 4 illustrates the probability distribution of the (Δ , Ψ) pair using dropout rates of 0.2 and 0.8. In both plots, the effect of Δ isnot considerable, where the horizontal variation, reflected by change in darkness, is similar; this is consistent with Table II and due to the fact that the dropout affects confidence rather than the predicted data.

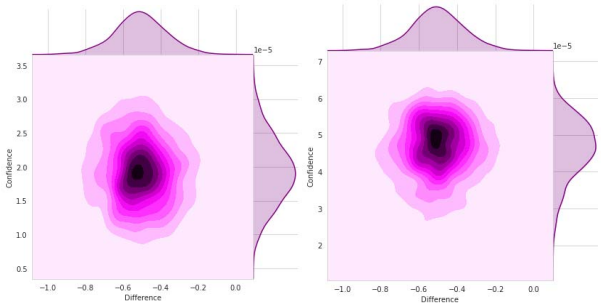

Fig. 4. When dropout has to vary up to: (a) 20%, and (b) 80%

Meanwhile, with a small dropout rate of 0.2, the range of the most frequent confidence levels broadens (approximately 0.7) in comparison to 0.5 when the dropout is 0.8. The shape of their respective histogram confirms that, where with a dropout rate of 0.8, the distribution of the most frequent confidence values is denser. Thus, applying a small dropout rate would make it very difficult to successfully guess a valid (Δ , Ψ) pair at a certain point of time; such dropout and along with a high precision setting for the data, the adversary has no chance to determine the MLGP measurement prediction model.

Security Analysis: In the following, we will prove the effectiveness to handle FDI and impersonation attack.

Lemma 1: MLGP mitigates (un)observable FDI attacks.

Proof: According to the description of FDI attacks in Section III, we distinguish the following cases:

Case #1: If the adversary does not know the topology matrix *H*, but could intercept and know Δ and Ψ, an FDI attack can be only based on random guesses by applying learning models based on the intercepted values of Δ and Ψ. Such an attack can be easily detected because the confidence will not match $\Psi_{Control}$. The validation results have also demonstrated the inability of learning and probabilistic models to yield the right confidence level. Thus any FDI will be detected as the source of the message will fail the authentication process. Specifically, if the adversary tries to apply:

(i) *Time series based learning models*: Such prediction mechanism minimizes the loss function and applies gradient descent to determine the variation of the parameter according to the loss. If the loss is deterministic (error is predictable), the optimization will detect the impact of the divergence on the model parameters. Thus, the prediction error is variable, which means that Δ cannot be guessed as it represents the correct measurement minus the predicted.

(ii) *Kernel distribution and Bayesian generative classification*: Varying the dropout rate makes Δ and Ψ uncorrelated.

Case #2: The attacker can strategically correlate some historical measurement to Δ in order to approximate the matrix *H*, e.g., using SVD [18]. Then, the adversary can create a set of critical measurements and inject an attack vector, which cannot be identified by traditional state estimation operators. The adversary can inject an attack vector of corrupted measurements $z' = z + \alpha$ and α=γH. However, the adversary needs to send

exclusively α as Δ. Let's assume that the value of $z + \alpha$ will be within the acceptable range and pass the state estimation checks. However, the adversary also needs to equate Ψ_{PMU} with $\Psi_{Control}$, which will fail as shown in Case #1.

Lemma 2: MLGP mitigates message replay attacks.

Proof: Assuming that a PMU collects measurements every τ time units, the adversary could create a record of the transmitted PMU data (i.e., Δ) over time. The adversary may then send one of the previously captured values, Δ_A, that was transmitted at time T. If successful, the attacker may replay a sequence of messages that were transmitted at time ($T+\tau$), ($T+2\tau$), etc. However, the attack will fail since ψ_A will be based on outdated LSTM model state at the control station. In essence, the hidden state of the LSTM will differ and we control the probability of successful guesses of the confidence using adaptive dropout. Moreover, a replay attack is usually mitigated by correlating the successive state estimation [22]. Thus, the adversary needs to replay a long sequence the measurements and not Δ's, which is what the PMU actually transmits.

VII. Conclusions and Future Work

This paper has presented MLGP, a novel approach for guarding the smart power grid against message replay and false data injection attacks. MLGP employs an advanced machine learning technique, namely, LSTM, to predict the next measurements made by a PMU based on current and previous ones; the PMU only reports the deviation from between actual and predicted data to the control station. A confidence level is also generated by the LSTM model and transmitted along with the data. MLGP duplicates the measurement prediction mechanism at the control station in order to achieve consensus on the confidence level and infer the actual measurements by regenerating the predicted values at the PMU side. Thus, the authenticity of the transmitter and the integrity of the data will be confirmed by matching the confidence levels at both ends. MLGP is validated through simulation using synthetic and publically available datasets. The validation results have demonstrated the robustness of MLGP and the inability of an attacker to regenerate a valid data and confidence pair. In the future, we plan to extend the scope to counter multi-PMU based replay and FDI attacks.

Acknowledgement: This work is supported by in part by Cisco under contract # 12430.

Reference

[1] N. Komninos, E. Philippou, and A. Pitsillides, "Survey in Smart Grid and Smart Home Security: Issues, Challenges and Countermeasures," *IEEE Comm. Surveys & Tutorials,* vol.16, no.4, pp.1933-1954, 2014.

[2] H. Gharavi and B. Hu, "Synchrophasor Sensor Networks for Grid Communication and Protection," in *Proceedings of the IEEE*, vol. 105, no. 7, pp. 1408-1428, Jul. 2017.

[3] M. Hojabri, U. Dersch, A. Papaemmanouil, and P. Bosshart, "A Comprehensive Survey on Phasor Measurement Unit Applications in Distribution Systems," *Energies*, no. 12, no. 23, pp. 4552-4574, 2019.

[4] A. von Meier, E. Stewart, A. McEachern, M. Andersen and L. Mehrmanesh, "Precision Micro-Synchrophasors for Distribution Systems: A Summary of Applications," *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 2926-2936, Nov. 2017.

[5] P. Kumar, Y. Lin, G. Bai, A. Paverd, J. S. Dong and A. Martin, "Smart Grid Metering Networks: A Survey on Security, Privacy and Open Research Issues," *IEEE Comm. Surv. & Tut.*, 21(3), pp. 2886-2927, 2019.

[6] H. He and J. Yan, "Cyber-physical attacks and defences in the smart grid: A survey," *IET Cyber Phys. Syst. Theory Appl.*, vol. 1, no. 1, pp. 13–27, Dec. 2016.

[7] S. Tan, D. De, W. Song, J. Yang and S. K. Das, "Survey of Security Advances in Smart Grid: A Data Driven Approach," *IEEE Comm. Surveys & Tutorials*, vol. 19, no. 1, pp. 397-422, First quarter 2017.

[8] T. Bi, J. Guo, K. Xu, L. Zhang and Q. Yang, "The Impact of Time Synchronization Deviation on the Performance of Synchrophasor Measurements and Wide Area Damping Control," *IEEE Transactions on Smart Grid*, vol. 8, no. 4, pp. 1545-1552, Jul. 2017.

[9] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," *ACM Transactions on Information and System Security (TISSEC)*, vol. 14, no. 1, p. 13, 2011.

[10] O. Kosut, L. Jia, R.J. Thomas, and L. Tong, "Limiting false data attacks on power system state estimation," *Proc. of the 44th Annual Conference on Information Sciences and Systems (CISS*), Princeton, NJ, March 2010.

[11] L. Liu, M. Esmalifalak, Q. Ding, V.A. Emesih, and Z. Han, "Detecting false data injection attacks on power grid by sparse optimization," *IEEE Trans. Smart Grid*, vol. 5, no., 2, pp. 612–621, 2014.

[12] G. Chaojun, P. Jirutitijaroen, M. Motani, "Detecting false data injection attacks in ac state estimation," *IEEE Trans. Smart Grid*, vol. 6, no. 5, pp. 2476–2483, 2015.

[13] M. Esmalifalak, L. Liu, N. Nguyen, R. Zheng, Z. Han, "Detecting stealthy false data injection using machine learning in smart grid," *IEEE Systems Journal*, vol. 11, no. 24, pp. 1644–1652. 2017

[14] Y. He, G.J. Mendis, and J. Wei, "Real-time detection of false data injection attacks in smart grid: A deep learning based intelligent mechanism," *IEEE Trans. Smart Grid*, vol. 8, pp. 2505–2516, 2017.

[15] Y. Wang, M.M. Amin, J. Fu, and H.B. Moussa, "A Novel Data Analytical Approach for False Data Injection Cyber-Physical Attack Mitigation in Smart Grids," *IEEE Access*, vol. 5, pp. 26022–26033, 2017.

[16] H. Wang, et al., "Deep Learning-Based Interval State Estimation of AC Smart Grids Against Sparse Cyber Attacks," *IEEE Trans. Ind. Inform.,* vol. 14, pp. 4766–4778, 2018.

[17] J. Hao, R.J. Piechocki, D. Kaleshi, W.H. Chin, and Z. Fan, "Sparse malicious false data injection attacks and defense mechanisms in smart grids," *IEEE Trans. Ind. Inform.*, vol. 11, pp. 1–12, 2015.

[18] A. Anwar and A. Mahmood, "Stealthy and blind false injection attacks on SCADA EMS in the presence of gross errors," *Proc. of IEEE Power and Energy Society General Meeting (PESGM)*, Boston MA, Jul. 2016.

[19] Y. Mo, R. Chabukswar, and B. Sinopoli, "Detecting Integrity Attacks on SCADA Systems," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 4, pp. 1396-1407, 2014.

[20] T. Tran, O. Shin and J. Lee, "Detection of replay attacks in smart grid systems," *Proc. of the Int'l Confe. on Computing, Management and Telecomm. (ComManTel 2013)*, Ho Chi Minh City, 2013, pp. 298-302.

[21] J. Zhao, J. Wang and L. Yin, "Detection and Control against Replay Attacks in Smart Grid," *Proc. of the 12th Int'l Conf. on Computational Intelligence and Security (CIS 2016)*, Wuxi, China, 2016, pp. 624-627.

[22] T. Irita and T. Namerikawa, "Detection of replay attack on smart grid with code signal and bargaining game," *Proc. of the American Control Conference (ACC 2017)*, Seattle, WA, 2017, pp. 2112-2117.

[23] A. Gomez-Exposito, and A. Abur, *Power System State Estimation: Theory and Implementation*; CRC Press: Boca Raton, FL, USA, 2004.

[24] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9 , no. 8, pp. 1735–1780, 1997.

[25] Y. Gal. "Uncertainty in Deep Learning," *PhD thesis,* University of Cambridge, 2016.

[26] V. Pham, C. Kermorvant, and J. Louradour, "Dropout improves recurrent neural networks for handwriting recognition," *arXiv preprint arXiv:1312.4569,* 2013.

[27] R. D. Zimmerman, C. Murillo-Sanchez, and R. J. Thomas, "Matpower's extensible optimal power flow architecture," *Proc. the IEEE Power and Energy Soc. General Meeting*, pp. 1–7, Calgery, Alberta, Jul. 2009.

[28] NYISO, "Load data profile," *http://www.nyiso.com*, May 2012.

[29]M. Naglic, "PMU measurements of IEEE 39-bus power system model," *IEEE Dataport*, 2019. [Online]. Available: http://dx.doi.org/10.21227/vkz3-2e96. Accessed: Aug. 21, 2020