Measurement Platform for Latency Characterization of Wide Area Monitoring, Protection and Control Systems

Paolo Castello, *Member, IEEE*, Giacomo Gallus, *Student Member, IEEE*, Paolo Attilio Pegoraro, *Senior Member, IEEE*, Sara Sulis, *Senior Member, IEEE*

***Abstract*—Wide Area Monitoring Protection and Control (WAMPAC) systems have emerged as a critical technology to improve the reliability, resilience, and stability of modern power grids. They are based on Phasor Measurement Unit (PMU) technology and synchronized monitoring on a wide area. Since these systems are required to make rapid decisions and control actions on the grid, they are characterized by stringent time constraints. For this reason, the latency of WAMPAC systems needs to be appropriately assessed. Following this necessity, this paper presents the design and implementation of a measurement platform that allows latency characterization of different types of WAMPAC systems in several operating conditions. The proposed WAMPAC Characterizer has been metrologically characterized through a WAMPAC Emulator and then used to measure the latency of a WAMPAC system based on an open-source platform frequently used by transmission system operators for the implementation of their PMU-based wide area systems.**

***Index Terms*—Wide Area Monitoring Protection and Control (WAMPAC), Wide Area Measurement System (WAMS), Phasor Measurement Units (PMUs), Power systems, Latency, Transmission System Operator**

I. Introduction

In recent decades, monitoring of power transmission systems has undergone a significant upgrade with the introduction of Wide Area Monitoring Systems (WAMSs) based on Phasor Measurement Unit (PMU) technology and their evolution towards Wide Area Monitoring Protection and Control (WAMPAC) systems. WAMPAC systems, based on advanced monitoring and communication technologies, offer an innovative approach for real-time synchronized monitoring, advanced protection and automated control on a wide area. The growing interest in WAMPAC systems is driven by the need to manage increasingly complex and interconnected electric grids, which are characterized by a high variability, due to the growing presence of distributed energy resources, the automation foreseen by smart grid paradigm and the continuous increase in energy demand. These factors have made the development of innovative solutions for monitoring and managing grids on a large scale essential.

P. Castello, G. Gallus, P. A. Pegoraro, and S. Sulis are with the Department of Electrical and Electronic Engineering of the University of Cagliari, via Marengo 2, 09123 Cagliari, Italy (email: [paolo.castello, giacomo.gallus, paolo.pegoraro, sara.sulis]@unica.it).

Dr. Pegoraro's work was partially funded by Fondazione di Sardegna for the research project "IQSS", Information Quality aware and Secure Sensor networks for smart cities".

WAMPAC systems involve, as the first element in their monitoring architecture, the PMU [1], which is capable of measuring the so-called voltage and current synchrophasors, instantaneous frequency and rate of change of frequency (ROCOF) [2], and the Phasor Data Concentrator (PDC), which enables the reception and time alignment of measurements from PMUs distributed throughout the power grid [3]. Measurements made by PMUs require a common and absolute time reference in order to be compared with each other and, for this reason, synchronization technologies that can provide high temporal accuracy (better than 1 µs [2]) are used. To this aim, PMUs are often equipped with a Global Positioning System (GPS) receiver or make use of packet-based synchronization, like that provided by Precision Time Protocol (PTP), to tag each measurement with an absolute timestamp in the Coordinated Universal Time (UTC) timescale. The measurements computed by PMUs are usually encapsulated in data packets compliant with the IEEE C37.118.2 standard [4].

PMUs play a crucial role in a wide range of WAMPAC applications, as the monitoring and control actions are based on synchrophasor, frequency and ROCOF measurements. In [5], WAMPAC applications based on ROCOF estimation with three established state-of-the-art algorithms are analyzed, and their impact on the successful grid restoration is shown. In [6], several WAMPAC applications are described, including recording of dynamic events, real-time system state determination and phase-angle monitoring (PAM). PAM, for instance, is an application of significant relevance to WAMPAC systems, as it provides information about power transfer [7]. In addition, it allows handling angular instability issues in the transmission network by activating real-time protection functions in order to prevent propagation phenomena and thus improve resilience [8]. As often happens with WAMPAC applications, there are still no standardized recommendations, e.g., regarding angular difference bounds for transmission grids, as pointed out in [8]. Indeed, the thresholds considered by WAMPAC depend on the specific context and transmission system operator (TSO) expertise.

The other key component in the measurement chain of the WAMPAC system is the PDC. This element, initially conceived as a module responsible for receiving data from the PMUs, is now assuming an increasingly active role in control systems, as it is the first element to supervise a network portion in the synchrophasor hierarchy. Indeed, the IEEE C37.247-2019 PDC standard [3] recommends the ability of

PDCs to implement control logic, such as event detection and classification, and examples of these techniques are described in [9]. In [10], a PDC prototype capable of handling delay and accurately evaluating PMU data flow latencies is presented. In [11], the architecture and validation of a low-latency PDC are studied using different network infrastructures, e.g., with fiber-optic cables or with 4G LTE wireless technology.

In this context, one of the critical aspects in the design and implementation of WAMPAC systems is the total latency, i.e., the time delay between the occurrence of an event in the electric system and the execution of associated control actions. Latency can greatly affect overall system performance, with possible implications for the safety, reliability and stability of the grid. Therefore, it is useful to evaluate the latency of each element within the monitoring and control chain. In this regard, it would be beneficial to propagate time properly across the entire network infrastructure [12].

Depending on the implemented algorithm, PMUs can introduce considerable delay into WAMPAC systems and, for this reason, the literature has focused on their latency characteristics and requirements. In [13], for instance, a low-latency (less than 10 ms) algorithm, suitable for control applications, is proposed. Characterization of PMU latency is analyzed in [14], [15].

To reduce the delay caused by communication, the network topology must also be carefully evaluated [16], [17], and different communication technologies, like supplementary wireless link, can be used to compensate temporary wired network failures [18]. In [19], the experience of the Croatian Transmission System Operator (HOPS) with a PMU-based angle stability protection application is presented. The communication latency due to the different geographical locations of the PMUs installed in their network is discussed.

Focusing on the control chain, a characterization of the operational delays of control systems using real-time digital simulator (RTDS) with hardware-in-the-loop (HIL) architecture is presented in [20]. A similar approach is followed in several scientific works including [21] (using OPAL-RT) and [22], [23] (using RTDS), but, in this literature, no measurement tool is provided specifically for the latency of the WAMPAC system and no metrological assessment is reported. In [24], EPRI reports the results of experiments involving two ad-hoc modified PMUs, an Event and Time monitor and a PDC. Using different transport protocols, the delays of the event detection are computed with resolutions equal to the reporting interval.

From another perspective, WAMPAC systems, which act automatically on the power network, should be also secure and resilient against possible cyber-attacks. In this context, cybersecurity in WAMPACs involves several levels, from design to system vulnerability testing. A detailed discussion can be found in [25], and a series of technical recommendations are reported in [26]. Focusing on specific examples, in [27], possible attacks on a WAMPAC infrastructure equipped with attack-resilient algorithms and a defense architecture are explored. In [28]–[30], the impact of time synchronization attacks is experimentally evaluated, which is important since PMU synchronization error impacts on the accuracy of phase-angle and derived measurements and also on the associated timestamps, thus affecting the PMU sending time and latency.

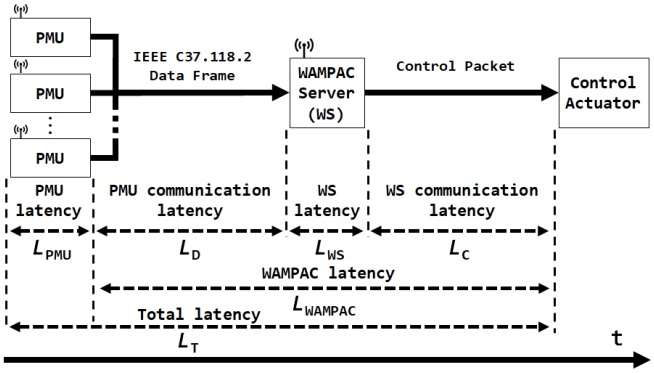

Fig. 1. Diagram of the considered architecture of a generic WAMPAC system, composed of PMUs, WAMPAC Server (WS) and Control Actuator, with the associated latencies.

Given the importance of characterizing the delays within WAMPAC systems, in [31] the latency of a WAMPAC prototype of the Italian TSO was characterized through an emulator of 2 PMU TCP data streams, with fixed reporting rate and packet size, and based on 4-20 mA actuator technology.

Stimulated by these preliminary studies, this paper presents the design and implementation of a WAMPAC latency characterizer, i.e., an instrument to accurately measure latency in purely digital WAMPAC systems, which allows testing several PMUs at once, with different protocols, packet sizes and reporting rates. A modular instrumentation platform by National Instruments (NI) equipped with real-time hardware is used to prototype the proposed design. In order to carry out a metrological verification of the functionality and performance of the proposed instrument, a second device has been designed and implemented to act as an emulator of a WAMPAC system, which is programmable and able to mimic real WAMPAC features. Then, the experimental characterization of a WAMPAC system based on software tools commonly used in real WAMS/WAMPAC applications is illustrated and confirms the validity of the proposed measurement platform.

II. Wide Area Monitoring and Protection Systems

The considered WAMPAC architecture, shown in Fig. 1, consists of PMUs scattered throughout the territory, a WAMPAC core (referred to as WAMPAC Server, WS from here on) and a module representing the control actuator. This is a common scheme and is used throughout this paper for presentation purpose. Specific adaptations of such architecture can be also addressed with slight variations of the proposed approach.

PMUs are distributed in the power network (e.g., in 380 kV and 220 kV substations) and can be configured according to two compliance classes, Class P and M, defined by the synchrophasor standard IEC/IEEE Std 60255-118-1 [2]. Class P is used for applications that require measurements with short response times and thus it is expected to play a role especially for electric system protection, whereas Class M has less stringent timings and is preferable where better disturbance rejection and larger off-nominal ranges are more important.

For this reason, the use of P-type algorithms is often considered preferable for WAMPAC systems, in order to reduce the so-called reporting latency introduced by PMUs, which mainly depends on the algorithm computation window. An interesting solution for WAMPAC systems would be to choose PMUs that meet the more stringent latency requirements of the two classes [32]. Also crucial is the choice of the reporting rate (RR) of the PMUs on which latency may depend. The RR commonly used by TSOs is 50 frames per second (fps), which means a packet containing measurements every $20\,\text{ms}$ [33].

These packets are typically received by an *ad hoc* PDC-enabled device (the WS in the considered architecture) that time-aligns the packets and then, in the WAMPAC architecture, analyzes the incoming data and sends a control packet according to several possible implementations. For example, it may send a command only when the PMU measurements satisfy a certain WAMPAC policy. The packet sent by the WS can be of various types, such as a Routable-Generic Object Oriented Substation Event (R-GOOSE) message conforming to IEC 61850-90-5 [34], like in the WAMPAC system in [31], or an IEEE C37.118.2 packet as those sent by the PMUs.

The IEEE C37.118.2 standard specifies the various packet types or frames for real-time communication between devices in a distributed measurement architecture. The standard outlines frames of the header, data, command, and configuration types. The focus here will be on the data frames, which contain the measurements collected from the PMUs, as well as command frames. Command frames are primarily used to initiate communication between PDCs and PMUs or to request information from the devices. In a pure monitoring context like WAMS, the role of the command frame is considered limited to the basic functionality defined in the IEEE C37.118.2 protocol. However, the standard defines seven standard commands (turn on transmission, turn off, send configuration frames, etc.), with some bits reserved for user designation. Wide-area applications that rely on synchronized measurements may thus use updated versions of the command frames for specific control applications.

An updated IEEE C37.118.2 command frame with a specific command for communication between WS and control actuator is considered in this paper as an example of possible implementation. Such command frame allows a PDC to evolve into an active device for power system control. This message can be used in many applications as the activation signal for the actuator of the control (see Fig. 1). Otherwise, the content of the control packet can be transformed by adapters/gateways into various analogue control signals if a WAMPAC system requires it (e.g., current signals in the 4-20 mA range as in [31]). The control actuator, based on the PMU measurements and the policies implemented by the WS, allows physical intervention in the system according to the required strategy.

The total latency L_T of the system, schematized in Fig. 1, is given by:

$$L_\text{T} = L_\text{PMU} + L_\text{WAMPAC} \tag{1}$$

with:

$$L_\text{WAMPAC} = L_\text{D} + L_\text{WS} + L_\text{C} \tag{2}$$

where:

- PMU reporting latency L_PMU in (1) is the time interval between the measurement time as indicated by the data timestamp, and the instant when the data becomes available at the PMU output [2];
- WAMPAC latency L_WAMPAC is defined in (2) as the sum of the following terms:
  - PMU communication latency L_D is the time needed for the network to transmit the IEEE C37.118.2 data frame from the PMU to the WS;
  - WS latency L_WS is the time required for the WS to process data from the PMUs and send a control/command message;
  - WS communication latency L_C is the delay of the control/command message sent by the WS over the network.

The latency of the PMU is not taken into account in L_WAMPAC, as it depends mainly on the instrument, and the objective of latency characterization is twofold: find the total latency and find the latency introduced by WAMPAC system regardless the chosen commercial PMUs or their configuration. It is worth recalling that L_PMU depends on the performance class and RR considered, and its maximum value is recommended in [2]. For this reason, and since L_PMU is usually measured in advance by manufacturer or laboratory characterization, here the focus is on L_WAMPAC. As it will be described in what follows, the designed characterization platform uses the IEEE C37.118.2 PMU packet output time (referred to as PMU sending time in the following) for the calculation of L_WAMPAC.

It is worth noticing that in a distributed PMU framework, i.e., in the presence of merging units or instrument transformers with digital output, the contributions given by the latency of the digitizer and the communication latency of the packets carrying the sampled values sent to the PMU [35] affect L_PMU. Therefore, these delays have to be characterized to understand their weight.

L_D and L_C include also the latency caused by devices, like routers, in the communication network. Moreover, L_D may be different between PMUs because they are located at different nodes in the network.

L_WS can in turn be divided into two contributions, according to needed WS functionality operation [3]:

- the "WS wait time" starts when the first PMU message with a given timestamp arrives and ends when the last message is received or the configured maximum wait time expires, after which the partial dataset is handled. The difference between the end of such wait time and the arrival time of the last on-time PMU packet contributes to L_WS;
- the "WS processing time", usually quite smaller than the first, represents the time for the WS to process and complete the output stream. In WAMPAC applications, this interval includes WAMPAC policy evaluation.

As a general consideration, if the architecture is more complex than that in Fig. 1, e.g., if several PDCs are present in the hierarchy, the latency of each device and the communication latency of their sent packets is also considered in L_{WAMPAC}. Furthermore, if the message sent by the WS needs to be transformed into an analogue signal, the latency given by the transducer device, which is usually negligible with respect to other contributions, should also be considered.

III. Proposed Latency Characterization System

In order to evaluate the latency of a generic WAMPAC system, a "WAMPAC Characterizer" (also briefly "Characterizer" from here on) has been designed and implemented to emulate the behavior of different PMUs. Indeed, the Characterizer allows sending several packets compliant with the IEEE C37.118.2 standard as if they were originated from a set of PMUs. In each PMU stream, packets are sent every T_{RR} instants, i.e., according to the configured PMU RR. On the other hand, the Characterizer allows receiving control packets from the WAMPAC system, thus closing the control chain and allowing latency evaluation. As mentioned above, control packets are here assumed, for the sake of presentation clarity, to be formatted according to IEEE C37.118.2, which is a realistic choice in many applications. This choice goes in the direction of industrial best practice [26] and the typical scenario is that of TSO private or virtual private communication networks. The Characterizer matches the PMU communication capabilities and, in this regard, if additional security reinforcement approaches (e.g., security gateways) are used for PMU data, they can be applied straightforwardly also to the Characterizer output. It is foreseeable that, in future scenarios, when security protocols may be applied directly to PMU streaming [36], dedicated solutions can be used to extend the Characterizer capabilities in this perspective.

The WAMPAC Characterizer, which will be described in detail in Section III-A, is an instrument devoted to WAMPAC latency measurement and thus needs in turn to be characterized to assess its performance before its application on the field. For this reason, a device called "WAMPAC Emulator" (or "Emulator") was also designed and implemented to physically emulate the behavior of a generic WAMPAC architecture with minimum additional latency.

Both the Characterizer and the Emulator allow configuring the number of PMUs, their RR and the PMU data frame size so that different scenarios of WAMPAC systems can be tested.

Figure 2 shows the setup used for the laboratory characterization tests, in which there is a direct connection between the Characterizer and the Emulator to measure the latency required by a WAMPAC chain that mimics a real architecture without the impact of communication latency due to network devices and external network traffic. Through characterization, which will be discussed in Section IV-B, an evaluation of the internal latency of the Characterizer can be established in order to provide an indication of the measurement error introduced by the instrument. Then, the Characterizer can be applied to a real WAMPAC scenario (see Section IV-C).

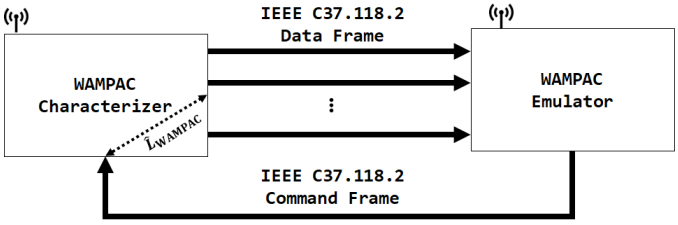

Fig. 2. Schematic diagram of the laboratory characterization architecture.

A. WAMPAC Characterizer

The WAMPAC Characterizer allows the emulation of a variable number of PMUs (during the test, this number was raised up to 10, but more PMUs can be emulated depending on the computational capability of the considered hardware platform) capable of generating IEEE C37.118.2 packets with different packet sizes. User Datagram Protocol (UDP) or Transmission Control Protocol (TCP) can be chosen for the transport layer protocol as common to commercial PMUs. The RR of PMUs is also user-modifiable, and a RR up to 200 fps can be configured. Once again, higher RRs can be also used depending on the computational power available. These features make the WAMPAC Characterizer more flexible for testing WAMPAC systems with different requirements compared to the solution in [31], which allowed emulation of only 2 PMUs with RR at 50 fps and TCP messages of fixed packet size.

To minimize the intrinsic latency of the Characterizer, that inevitably affects the measured WAMPAC latency, the emulated PMU packets are generated at UTC time instants corresponding to the PMU reporting instants (multiples of T_{RR} and written as the packet timestamps). In this way, packet flows emulate the real behaviour of PMUs without introducing PMU measurement latency ($L_{\text{PMU}} \approx 0$). Moreover, to increase the level of emulation of the PMUs, the Characterizer can implement a specific delay for the sent packets of each PMU. In this way, real cases in which the PMUs do not have the same communication or reporting latency can be simulated.

Whenever packets have to be sent, the sending time of each packet ($t_{\text{DS},i}$, where i represents the index of the emulated PMU) is obtained and written into the sent packet as an analogue field. It allows having also the sending time available at the side of the receiver and not only the PMU timestamp (reporting time). Clearly, this option can be deactivated when the WAMPAC does not support the interpretation of this additional information, or the exact packet format of the real system needs to be used. After that, $t_{\text{DS},i}$ of each PMU is stored in the system.

As mentioned above, the Characterizer is programmed to receive IEEE C37.118.2 packets from the WAMPAC system in response to PMU packets. Whenever a control packet is received, the current time (t_{CR}) is stored, and the difference between this arrival time and the largest timestamp of the corresponding PMU packets is calculated, thus computing the following latency:

$$\hat{L}_{\text{WAMPAC}} = t_{\text{CR}} - \max(t_{\text{DS},1}, t_{\text{DS},2}, \ldots, t_{\text{DS},i}, \ldots, t_{\text{DS},N}) \quad (3)$$

where $\widehat{}$ denotes hereinafter measured quantities and N is the total number of emulated PMUs.

In addition, the Characterizer also computes the received control packet latency (WS communication latency)

$$\widehat{L}_{\mathrm{C}} = t_{\mathrm{CR}} - t_{\mathrm{CS}} \tag{4}$$

defined as the difference between the WS packet arrival time t_{CR} and the timestamp of the command frame, t_{CS}. In this way, the Characterizer calculates the latency of the packets coming from the WS, estimating the communication latency and allowing to highlight its contribution to the WAMPAC latency.

Figure 3 shows the main functionalities of the Characterizer using a block diagram. The proposed instrument is designed to rely on real-time (RT) hardware and software and on a platform equipped with Field Programmable Gate Array (FPGA) to guarantee determinism in performance. These features are important to address timing and synchronization requirements and achieve high measurement accuracy as will discussed in what follows. Configuration operations are performed on the parameters that can be set, such as the number of PMUs N, RR, packet payload size, protocol type and input data file. The actual synchrophasor streams of the various emulated PMUs can be chosen by the user based on an auxiliary file of measured values (e.g., 3-phase synchrophasor, frequency and ROCOF values for a certain time interval, collected from the field or simulated). The measured values can be used cyclically to test WAMPAC response repeatedly during long tests. The Characterizer can also provide simple measurement streams generated internally.

The main functionalities are divided into two parallel RT cycles, disciplined by the internal clock with a frequency equal to 10 MHz. The top RT cycle allows the set of PMUs to be emulated in terms of data transmission and timestamping. Indeed, the first block ensures alignment with reporting time instants by calculating the wait time required for alignment according to UTC. Whenever a time reference is needed, as seen in Fig. 3, the program makes a software call to the FPGA, which provides UTC time. Indeed, the FPGA is synchronized using an external time source; in this paper, a GPS receiver is considered.

Within a number N of parallel instances, the UTC time is obtained and, consequently, an IEEE C37.118.2 data frame packet is built and sent with the corresponding reporting timestamp and, as mentioned above, with the measured sending time $t_{\mathrm{DS},i}$. The parallel instances are distributed efficiently according to the number of processors available in the platform adopted for implementation. At the output of the parallelized loop, a stop condition is evaluated: for instance, if the count of sent measurements is less than a set number (which depends on the RR, the length of the test, and the number of stream iterations), the RT loop is re-executed otherwise the loop terminates.

The bottom RT loop allows the command or data frame packets of the WS to be received. The possibility of receiving data frame packets was also included to allow testing even WAMPAC systems based only on a PDC and thus on data frames. The first block is responsible for receiving and checking these packets.

When a packet is received, the current time is calculated and saved as t_{CR} and immediately afterwards the packet is decoded, extracting useful data such as

t_{CS}. After decoding, $\widehat{L}_{\mathrm{WAMPAC}}$ and $\widehat{L}_{\mathrm{C}}$ are computed and stored according to (3) and (4), respectively. Lastly, a stop condition is checked.

The proposed Characterizer has additional features: it can be indeed configured to be used also as an instrument to measure L_T, thus including PMU latency by checking the arrival time of the control packets with respect to the PMU timestamp. In this perspective, it allows testing mixed scenarios where the emulated PMUs and commercial PMUs coexist. L_{PMU} can be set for each emulated PMU in order to test different monitoring configurations.

The Characterizer was implemented in a NI CompactRIO-9068 (NI cRIO-9068), a modular system equipped with the NI Linux real-time operating system and provided with ARM Cortex-A9 dual-core processor with a maximum frequency of 667 MHz and 1 GB RAM. The NI cRIO-9068 was equipped with the NI 9467 module for GPS synchronization with a pulse per second (PPS) accuracy of ±100 ns. In this paper, to reflect the behavior of digital WAMPAC systems, an IEEE C37.118.2 packet was chosen as the activation signal. Nonetheless, the modular and configurable hardware is ready to adapt the system to the needs of the specific WAMPAC system under test by changing the nature of the activation signal, for instance using a current signal in the 4-20 mA range, or by changing the communication protocol, such as the IEC 61850-90-5 [34].

A specific version of the Characterizer software adapted to generic personal computers (PCs) was also implemented in order to provide another tool for preliminary latency assessment to TSOs and to evaluate the performance differences when using general purpose (GP) instead of RT hardware. This version differs from that used in the RT platform in terms of timestamping, since, in the absence of the FPGA, a software call to the computer time is used. The PC clock is synchronized via network time protocol (NTP). For instance, in the tests of Section IV below, the synchronization is provided by a Meinberg Lantime M1000 GPS receiver within the same LAN of the PC. Due to the non-RT behavior, it is not possible to have a highly accurate time and PC clock offset may drift up to 10 ms from the reference. However, the detected drift has a negligible impact on WAMPAC latency measurements when no strict synchronization is required by WAMPAC operation, and mainly affects communication latency measurements. For instance, if the Emulator is used, packet communication latencies to and from it are calculated based on timestamps from different timescales (internal computer clock and WS clock).

B. WAMPAC Emulator

The WAMPAC Emulator allows the emulation of fundamental WAMPAC functionalities to test the behavior of the Characterizer. Indeed, the Emulator implements the basic functions of the WS and can thus serve as a low-latency version of a real system. Therefore, the WAMPAC Emulator is capable of

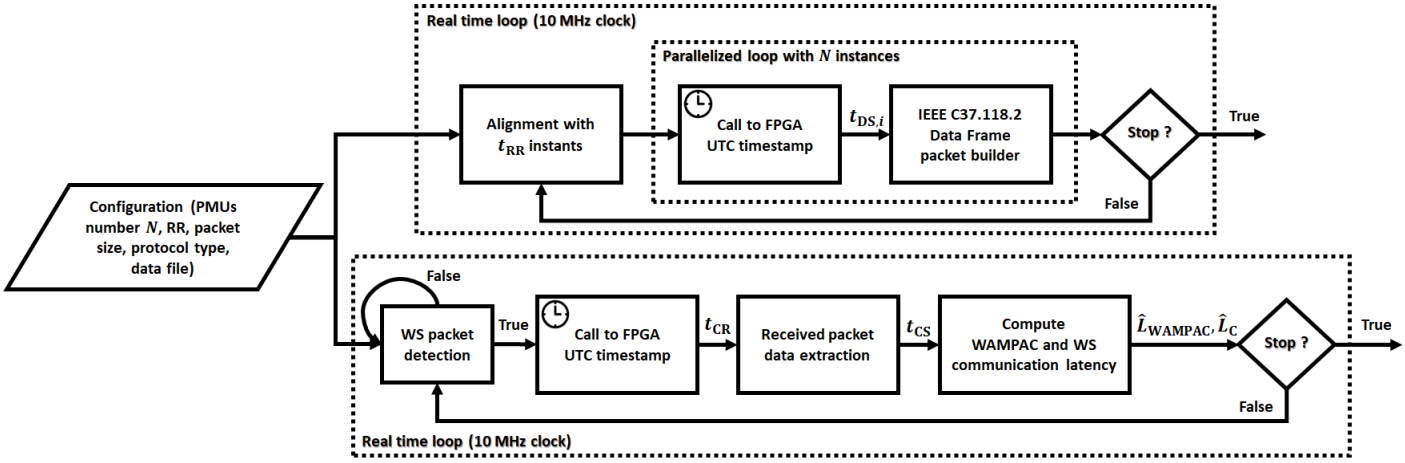

Fig. 3. Block diagram of the proposed WAMPAC Characterizer.

receiving the IEEE C37.118.2 packets from PMUs (both real and emulated) and can run in real time a WAMPAC logic. Indeed, a policy must be chosen to decide when the activation signal has to be sent (in an IEEE C37.118.2 command frame packet in this paper) for controlling the grid.

An example of WAMPAC policy based on phase-angle measurements, which is also adopted in the tests of Section IV, is the following rule:

$$\text{activation} = \begin{cases} \text{true} & \text{if } \max_{i=1,\dots,N} |\varphi_i - \varphi_{i_{\text{ref}}}| > \delta_\varphi \\ \text{false} & \text{otherwise} \end{cases} \quad (5)$$

where the activation signal is generated and the corresponding command frame is sent if the phase-angle difference between the phase-angle measurement $\varphi_{i_{\text{ref}}}$ of a PMU i_{ref}, chosen as reference, and at least one of the measured phase angles φ_i included in the other PMU packets with the same timestamp is greater than a threshold δ_φ. On the contrary, i.e., under normal conditions, the command packet is not sent. The simple PAM application considered here is inspired by that used by the Italian TSO in [31]. Furthermore, this simple policy appears effective when some real cases, such as the incident that occurred at the Continental Europe Synchronous Area (CESA) on January 8th 2021, are considered [37]. In that situation, shortly before the failure, the system was already operating close to the point of angular instability, with voltage phase-angle differences close to $90°$ between Western and Eastern Europe [37]. This event led to the separation of CESA into two sub-areas and demonstrated once again the importance of monitoring phase-angle differences between voltage phasors measured with PMUs at different locations, as they can be considered an early index of power transmission network stress [6], and thus may help in taking automatic countermeasures through purposely designed WAMPAC systems.

The Emulator is also capable of calculating the IEEE C37.118.2 packet latency $\widehat{L}_{\text{D},i}$ for the incoming PMU streams defined as the difference between the arrival time of input packet $t_{\text{DR},i}$ and the sending time $t_{\text{DS},i}$ written into the same packet, i.e.,

$$\widehat{L}_{\text{D},i} = t_{\text{DR},i} - t_{\text{DS},i} \quad (6)$$

In this way, the Emulator can analyze the behavior of the PMUs emulated by the WAMPAC Characterizer in more detail, providing additional insight. The Emulator also estimates its own processing latency $\widehat{L}_{\text{WE}}$ as:

$$\widehat{L}_{\text{WE}} = t_{\text{CS}} - \max(t_{\text{DR},1}, t_{\text{DR},2}, \dots, t_{\text{DR},i}, \dots, t_{\text{DR},N}) \quad (7)$$

$\widehat{L}_{\text{WE}}$, as the WS processing time defined in Section II, represents the difference between the instant when the activation signal is sent (t_{CS}) and the arrival time of the last PMU packet among those with the same timestamp that trigger the policy.

Analogous to that illustrated for the Characterizer, the block diagram of the Emulator implementation is shown in Fig. 4. Also in this case, configuration operations are carried out on the parameters that can be set. In the same way as the Characterizer, the Emulator uses an RT cycle, regulated by the internal clock at 10 MHz, containing an internal cycle with N parallel instances distributed appropriately to the available processors.

Within the parallelized loop, PMU packets are received. The arrival time $t_{\text{DR},i}$ is saved and the data frame is decoded, extracting $t_{\text{DS},i}$. After obtaining the required data for all the incoming PMUs, the WAMPAC policy is applied (voltage and current synchrophasors, frequency and ROCOF can be considered according to the adopted strategy). If the policy condition is verified, a timestamp is recorded and the command frame is built.

The output packet was assembled according to the standard [4], using the bits of the CMD field reserved for user designation (identified with bits 15-12 set to 0 and bits 11-8 not equal to 0), to obtain a specific command for WAMPAC application.

Therefore, in accordance with the WAMPAC policy summarized in (5), this functionality has been used to send the control packet to activate the control strategy.

After the packet is sent, the WAMPAC Emulator processing latency $\widehat{L}_{\text{WE}}$ and the PMU communication latencies $\widehat{L}_{\text{D},1}$, $\widehat{L}_{\text{D},2}$, ..., $\widehat{L}_{\text{D},i}$, ..., $\widehat{L}_{\text{D},N}$ are calculated, according to (7) and (6), respectively. If the policy results in a false condition, only the PMUs communication latencies are computed. Lastly, the RT cycle is re-executed according to the stop condition.

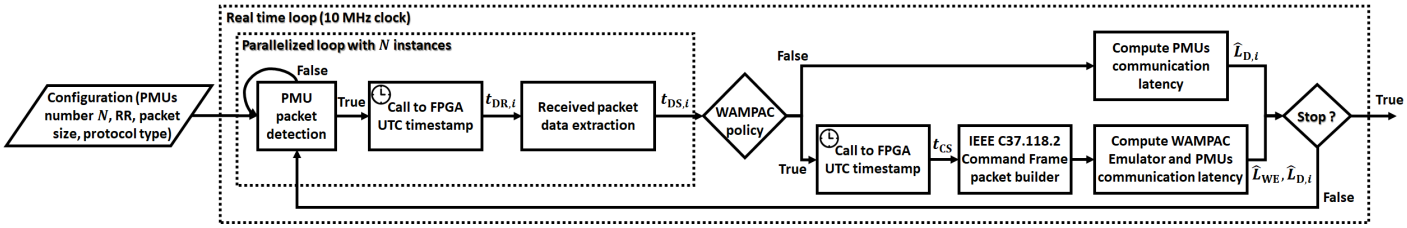

Fig. 4. Block diagram of the WAMPAC Emulator.

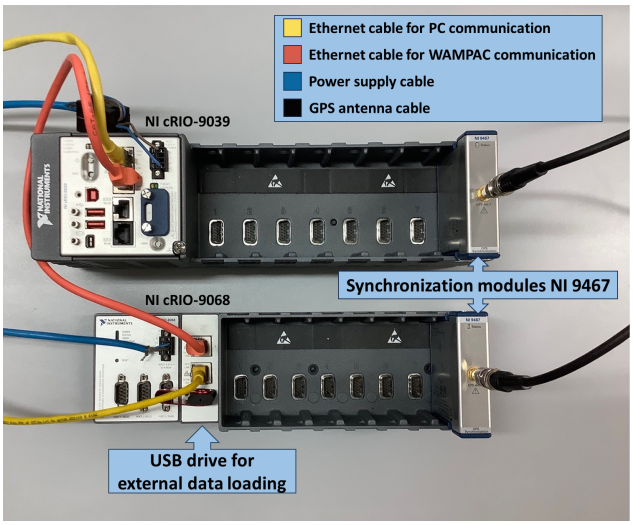

Fig. 5. Picture of the used hardware and connections for the laboratory characterization.

For the implementation of the WAMPAC Emulator, another modular system was used, the NI cRIO-9039, provided with NI Linux real-time operating system, Intel Atom quad-core processor with maximum frequency 1.91 GHz and 2 GB RAM, and equipped with the NI 9467 module for GPS synchronization. As in the WAMPAC Characterizer, the synchronization allows timestamping the packets arrival and departure with respect to UTC and consequently to calculate the aforementioned latencies in a coherent way.

Figure 5 shows the modular devices used for the test setup during laboratory characterization and highlights the involved connections.

IV. Tests and Results

In this section, two selected test scenarios and the corresponding test setup are presented. Then, the most significant results are reported and discussed.

A. Test Scenarios and Setup

The first test scenario corresponds to the Characterizer latency laboratory assessment involving the Emulator introduced in Section III and, in particular, in Section III-B. The test setup corresponds to Fig. 2. In the performed tests, PMU data correspond to synthetic synchrophasors generated directly by the WAMPAC Characterizer. Measured values were kept stable: for an emulated PMU assumed as reference, phase angle was constantly equal to $0°$ while, for the other emulated PMUs, it is kept at $110°$, so that the phase-angle difference was always greater than $\delta_\varphi = 100°$, which is considered as the event detection threshold. In this condition, the WAMPAC Emulator sends, for each set of packets from the emulated PMUs (each timestamp), a control packet. This allows characterizing the device in the worst stress condition, when it produces a latency measurement approximately every T_{RR}. This choice also permits a statistical assessment of latency measurement characteristics within short time intervals and is thus adopted from here on for the tests.

The second test scenario is an experimental latency measurement test for a WAMPAC system based on the OpenPDC tool by Grid Protection Alliance (GPA) [38]. This is an open-source software application intended as the core module in the WAMS and WAMPAC systems design. The OpenPDC is a data concentrator whose primary function is to time align and collect the synchrophasor data provided by the PMUs. It enables the management of data from PMUs in the field by also allowing outbound streams to be sent, concentrating data from the various incoming packets. Due to its capabilities, this software has been used in various scientific projects and real-field implementations [39], [40].

The test architecture is shown in Fig. 6(a). The OpenPDC is hosted in a workstation equipped with two Intel Xeon Gold 6128 processors, 256 GB RAM, and Windows 10 OS.

To simulate various behaviors of a real network, a network emulator has been also used within the previously described architecture, thus modified as shown in Fig. 6(b). It is then possible to test applications in the presence of real network issues such as higher latencies, jitter and packet loss conditions. Network emulation with a similar approach has been used in several scientific works [10], [41]. The network emulator has been hosted in a workstation, with an Intel E8500 processor, 8 GB RAM and a Linux operating system (Ubuntu 16.04 LTS), running the NetEm software [42]. The adopted workstation is equipped with several Ethernet network interfaces; two of them have been configured in transparent bridge mode, in order to influence the input and output data streams. As shown in Fig. 6(b), the network emulator has been connected between the WS and the Characterizer to simulate an additional latency to reach the actuator.

An additional test setup, shown in Fig. 6(c), has been used to test the same WAMPAC architecture in a mixed scenario, i.e., with the Characterizer and real PMUs connected to the same OpenPDC. In this case, the aim is to measure L_T, thus including the PMU reporting latency of a commercial PMU and the latency of PMUs emulated by the Characterizer.

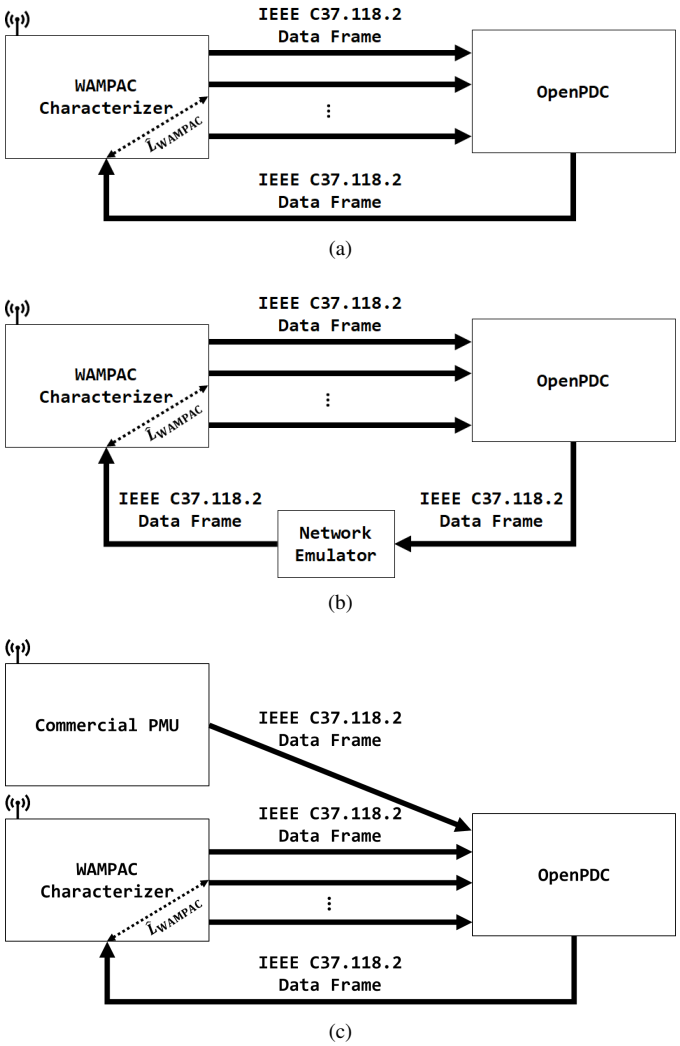

Fig. 6. Schematic diagram of the experimental test architecture (a), integration of the network emulator (b) and of a commercial PMU (c).

B. Validation of the WAMPAC Characterizer

The first series of tests is devoted to the characterization of the metrological performance of the WAMPAC Characterizer in a controlled laboratory environment (see Section IV-A). The tests have been divided into short and long tests.

The Characterizer was initially verified with 15-minute (short-term) tests in which all configuration parameters were varied in order to check the instrument operation for every possible combination. Specifically, the following options have been used:

- 2 to 10 PMUs;
- RRs of 10, 25, 50, 100 and 200 fps, i.e., with a packet every 100, 40, 20, 10 and 5 ms. This corresponds to 9000, 22500, 45000, 90000, and 180000 measures for the 15-minute tests;
- 3 packet types with small (50 bytes), medium (94 bytes) and large (458 bytes) payload size. These different sizes mostly depend on the number of synchrophasors included in the data frame (3, 6 and 45, respectively);
- UDP or TCP transport layer protocol.

For the characterization tests, the architecture presented in Fig. 2 has been used to assess the Characterizer's latency measurement accuracy. In particular, the results are summarized through the maximum value (indicated as 'Max' in the following tables), the mean value ('μ') and the standard deviation ('σ') of $\widehat{L}_{\text{WAMPAC}}$ values provided by the Characterizer. Tables I and II present the results of the tests performed using the NI cRIO-9068, RR of 50 and 100 fps, and 2 or 10 PMUs.

In particular, Table I reports the results obtained when the UDP protocol is set, while Table II refers to TCP protocol. It is possible to observe the remarkable stability obtained with RT hardware: up to a RR of 100 fps, a maximum latency of about 1 ms was measured in the UDP case and about 1.3 ms in the TCP case. It is important to highlight that the obtained latency values include not only the effect of the Characterizer, but also the latency concerning the Emulator, thus guaranteeing that found maximum values are actually upper bounds for the latency introduced by the Characterizer.

TABLE I
WAMPAC LATENCY RESULTS FOR UDP PROTOCOL AND RT HARDWARE

Configuration		Reporting rate [fps]					
		50			100		
Number of PMUs	Payload size [B]	Latency index [ms]					
		Max	μ	σ	Max	μ	σ
2	50	0.90	0.64	0.03	0.93	0.64	0.03
	94	0.89	0.65	0.03	0.90	0.65	0.03
	458	0.96	0.70	0.03	1.02	0.69	0.03
10	50	0.87	0.64	0.03	0.95	0.63	0.03
	94	0.96	0.65	0.03	1.00	0.64	0.03
	458	0.98	0.70	0.03	1.05	0.70	0.03

TABLE II
WAMPAC LATENCY RESULTS FOR TCP PROTOCOL AND RT HARDWARE

Configuration		Reporting rate [fps]					
		50			100		
Number of PMUs	Payload size [B]	Latency index [ms]					
		Max	μ	σ	Max	μ	σ
2	50	1.03	0.74	0.03	1.04	0.73	0.04
	94	1.03	0.74	0.04	1.01	0.74	0.04
	458	1.08	0.79	0.04	1.10	0.79	0.04
10	50	1.24	0.74	0.05	1.28	0.73	0.05
	94	1.26	0.75	0.05	1.38	0.74	0.05
	458	1.31	0.80	0.05	1.30	0.80	0.05

It can be seen from the test results that μ remains almost constant as the number of emulated PMUs increases in the RT case. This, considering also that σ is practically constant too, means that the RT device, even if the number of operations to perform is higher, manages to keep its performance almost unchanged in the various tests until time constraints can be respected. This is also confirmed by Fig. 7, which reports the histograms of $\widehat{L}_{\text{WAMPAC}}$ measured when RR $=$ 100 fps, small payload size, UDP protocol, and 2, 5 and 10 emulated PMUs are considered (red, green and blue bars, respectively). From the figure, it can be seen that the histograms are mostly overlapping.

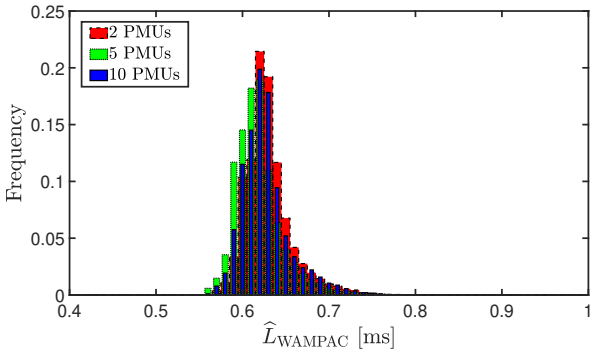

Fig. 7. Histograms of the WAMPAC latency measured for 15 minutes, considering 2, 5 and 10 PMUs, RR at 100 fps and UDP small packet size.

On the other hand, with a RR of 200 fps, the system exhibits a state of heightened strain due to an increased number of concurrent requests. The load level of the CPU shows peaks at 100 % occupancy and, for this reason, the device has some slowdowns, leading to higher maxima of measured latency in some tests. These correspond to few outliers that do not significantly affect μ and σ but reduce Characterizer's capability to accurately monitor a given system. Higher RRs are not manageable with the considered RT platform.

Tables I and II also show that, as the packet size increases, μ slightly increases, as expected. In addition, μ is generally lower using UDP protocol than using TCP. This is because UDP is a lighter and faster protocol than TCP.

Furthermore, the TCP packet usually has a larger header than UDP packet (in our case, the TCP header is 20 bytes while that of UDP packets is 8 bytes) considering the same number of synchrophasors transmitted.

Table III shows an extract of the results achieved with GP hardware with UDP protocol and the same test length (see Table I for comparison). For these tests, the Characterizer software was run in a workstation equipped with two Intel Xeon Gold 6128 processors, 256 GB RAM, and Windows 10 OS. To synchronize the workstation, NetTime software [43] with a 15-minute refresh interval was used. The results are much more variable and maximum values are quite pronounced due to frequent interrupts that can slow down code execution (the worst found result is about 90 ms for 7 PMU, small packet and RR $=$ 200 fps). In this case, σ reflects the increased variability, whereas μ can be even lower in some tests than that obtained with RT hardware. Indeed, the used workstation provides more computing power than the employed RT system but no determinism is guaranteed. As for TCP, all these considerations still hold but, as in the RT case, with generally higher μ values than UDP (the results are not reported here for brevity).

To clarify the differences in the results obtained with the RT device and the GP device, Fig. 8 shows the trends of the measured WAMPAC latency in case of 2 emulated PMUs over 15 minutes. From the figure, the higher stability of the values obtained with RT hardware is evident, whereas GP latency measurements are characterized by the presence of outliers. Therefore, the characterization of a WAMPAC system

TABLE III
WAMPAC LATENCY RESULTS FOR UDP PROTOCOL AND GP HARDWARE

Configuration		Reporting rate [fps] 50			100		
Number of PMUs	Payload size [B]	Latency index [ms] Max	μ	σ	Max	μ	σ
2	50	30.10	0.55	0.18	11.52	0.47	0.07
	94	7.45	0.54	0.10	45.19	0.49	0.40
	458	8.64	0.50	0.09	4.49	0.48	0.06
10	50	24.52	0.81	0.15	9.17	0.73	0.09
	94	4.24	0.78	0.09	49.05	0.74	0.28
	458	26.64	0.80	0.18	35.50	0.76	0.19

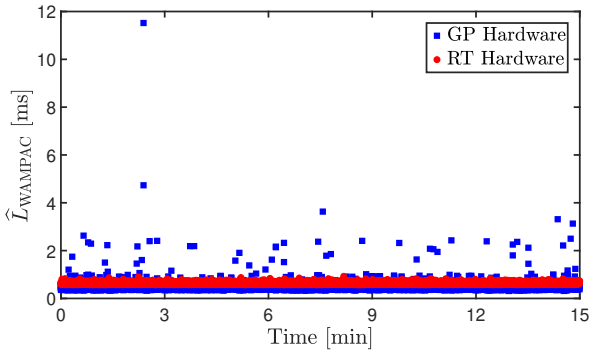

Fig. 8. WAMPAC latency measured by RT and GP hardware, considering 2 PMUs, RR at 100 fps and UDP small packet size.

using the RT system shows, as expected, stable performance over time. Nonetheless, the comparison reveals that the GP implementation, even if not highly accurate, can be considered a useful tool for preliminary compliance tests of WAMPAC functionalities and investigation of the system behavior.

To get further into the Characterizer assessment, the various contributions to $\widehat{L}_{\mathrm{WAMPAC}}$ are isolated and shown in Fig. 9, considering 10 emulated PMUs, 50 fps, UDP protocol and medium packet size. $\widehat{L}_{\mathrm{WAMPAC}}$ and $\widehat{L}_{\mathrm{C}}$ (latency of the control packet) measurements, obtained with the Characterizer, are reported as a function of time together with the $\widehat{L}_{\mathrm{WE}}$, measured internally by the Emulator, and the maximum $\widehat{L}_{\mathrm{D},i}$ obtained by the Emulator among the PMUs. Table IV shows μ and σ of the individual latency components measured during the test. These results indicate that the latency introduced by the WAMPAC Emulator can be safely considered to be lower than 0.1 ms. Obviously, this latency contribution is related to the complexity of the policy and the number of PMUs, but, thanks to the RT performance, it is still an order of magnitude below other latency contributions.

To verify the functionality in the presence of a network issue, an extra communication delay in a test over 15 minutes has also been added. The above test configuration has been used and the data stream of the 10th PMU is delayed by 1 ms starting from the middle of the test duration and leaving the other PMU streams unaffected. The test shows that PMU communication latency (and consequently also WAMPAC latency) has an instantaneous increment equal to the extra delay, as expected.

This article has been accepted for publication in IEEE Transactions on Instrumentation and Measurement. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIM.2023.3334360

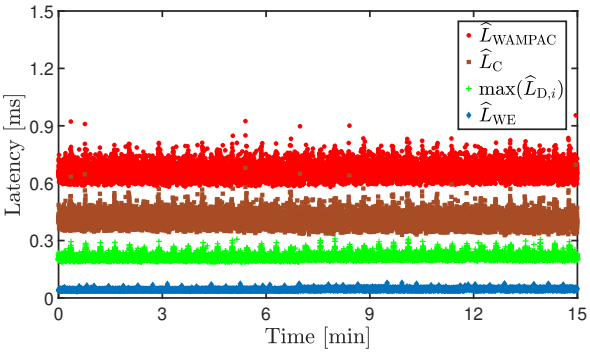

Fig. 9. From the top to the bottom: WAMPAC latency (red dots), control packet latency (brown squares), WAMPAC Emulator processing latency (green crosses), and max PMU communication latency (blue diamonds), considering 10 PMUs, 50 fps and UDP medium packet size.

TABLE IV
LATENCY COMPONENTS FOR 10 PMUS, 50 FPS, MEDIUM PACKET SIZE, AND UDP PROTOCOL

Latency Component	Latency Index [ms]	
	μ	σ
$\widehat{L}_{\mathrm{WAMPAC}}$	0.65	0.03
$\widehat{L}_{\mathrm{C}}$	0.40	0.03
$\widehat{L}_{\mathrm{WE}}$	0.04	< 0.01
$\max(\widehat{L}_{\mathrm{D},i})$	0.20	0.01

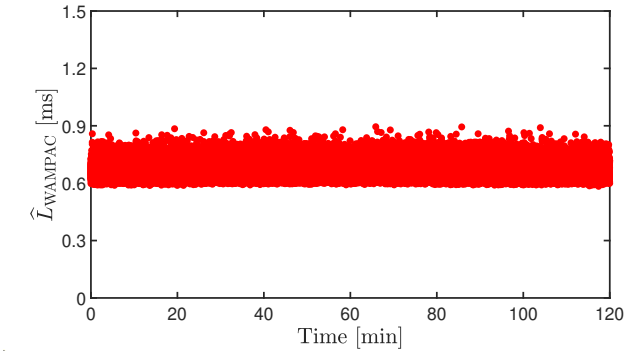

Fig. 10. WAMPAC latency measured for 2 hours, considering 2 PMUs, RR at 50 fps and UDP medium packet size.

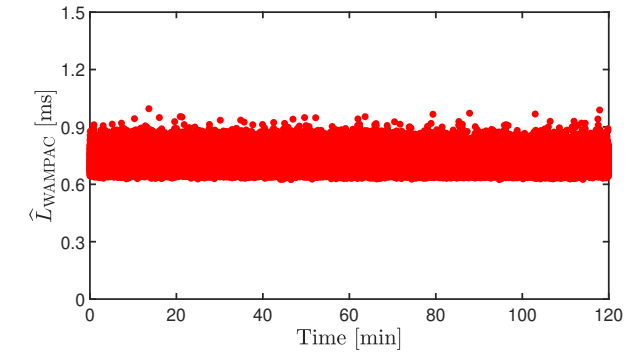

Fig. 11. WAMPAC latency measured for 2 hours through the integration of the network emulator without extra delay, considering 2 PMUs, RR at 50 fps, and UDP medium packet size.

The second set of characterization tests corresponds to long-duration tests, which have been carried out using the RT hardware. The Characterizer configuration is composed of 2 emulated PMUs, 50 fps, and medium packet size. In order to verify the stability and accuracy of the implemented system in the long run, 2-hour tests are considered with both UDP and TCP protocols. Figure 10 reports the latency measured by the Characterizer for the UDP case and reveals that it remains stable even during long tests. Almost the same results as in Table I, with values always below 0.90 ms (1.03 ms for TCP), were achieved: in particular, $\mu = 0.66$ ms and $\sigma = 0.03$ ms for the UDP case, while $\mu = 0.75$ ms and $\sigma = 0.03$ ms for the TCP case. The TCP case can also be compared with [31], leading to an accuracy improvement of one order of magnitude, from about 10 ms to 1 ms. Furthermore, these results can be used to interpret the accuracy of the measurements performed in the experimental tests of Section IV-C.

Afterwards, tests in the presence of network delay have been performed. A preliminary test was conducted to verify that the network emulator does not introduce significant extra delays (in addition to those set) into the measurement and control chain. For this reason, the network emulator has been configured to provide no extra delay. The test was carried out with both UDP and TCP protocols, but for the sake of brevity, only the UDP case is shown in Fig. 11 (results obtained with TCP are similar). In this test, $\mu = 0.70$ ms and $\sigma = 0.04$ ms have been obtained. As expected, both the mean and standard deviation are slightly larger than in previous tests without the network emulator, but the additional latency is negligible (the network emulator introduces an average latency of approximately 0.04 ms).

Finally, two additional characterization tests were carried out with UDP and TCP protocols by setting a significant delay in the network emulator in order to emulate the communication delay between the WS and the WAMPAC actuator. Considering that the maximum delay of the round-trip time in a real test case reported in [31] was 36 ms, 18 ms has been chosen as a representative one-way network delay, thus emulating the latency of a real communication network. For the UDP case, $\mu = 18.70$ ms and $\sigma = 0.03$ ms have been achieved. As expected, the obtained average value is shifted by 18 ms compared to the previous ones.

In all the short- and long-duration tests, no packet loss occurred, which is a crucial aspect to consider for employing the instrument in the field.

C. Experimental Tests on a Real WAMPAC System

In this subsection, the experimental tests conducted on the real WS based on OpenPDC and described in Section IV-A are presented. First, the architecture shown in Fig. 6(a) has been used and 2-hour tests have been carried out with both UDP and TCP protocols. Fig. 12 reports the results for the UDP case and a substantially similar behavior has been obtained with TCP. A linear trend with a positive slope can be observed interleaved with higher latency bursts. The linear trend lasts about 100 min and yields a variation of about 20 ms. This behavior is

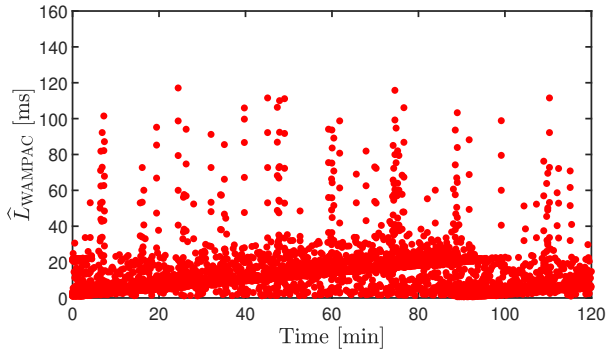

Fig. 12. WAMPAC latency measured for 2 hours through the connection to OpenPDC, considering 2 PMUs, RR at 50 fps and UDP medium packet size.

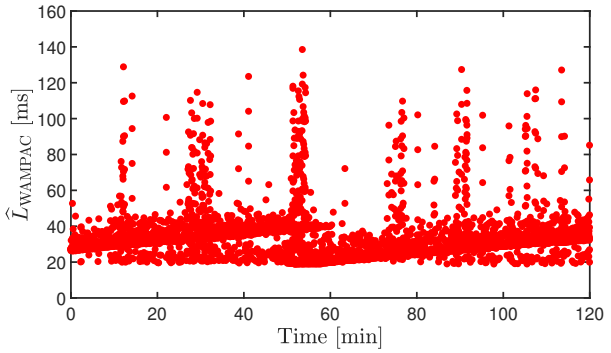

Fig. 13. WAMPAC latency measured for 2 hours through the connection to OpenPDC and network emulator with an additional delay of 18 ms, considering 2 PMUs, RR at 50 fps, and TCP medium packet size.

thus representative of OpenPDC operation. L_{WAMPAC} goes up to 120 ms in the UDP case and 140 ms in the TCP case for a few instants, while progressively smaller delays follow such spikes. In these tests, an average of 9.47 ms and a standard deviation of 6.16 ms were obtained for the UDP case, while $\mu = 12.19$ ms and $\sigma = 5.76$ ms were found in the TCP case. These results are intrinsically tied to the workstation and OpenPDC application configuration and characteristics. They also depend on the concurrent processes running on the workstation and can be partially improved by increasing the OpenPDC process priority. Several tests have been performed in this regard, highlighting the role of a non-real-time system in the latency trends. However, an analysis of the application configuration and performance is beyond the scope of this paper.

Moreover, two tests were carried out with the architecture in Fig. 6(b), introducing an additional delay of 18 ms through the NetEm like in Section IV-B. The results are shown in Fig. 13 for the TCP case, leading to $\mu = 29.38$ ms and $\sigma = 5.33$ ms. The latency values appear substantially shifted by the added delay with respect to the prior results, but the pattern is the same as in the previous test.

Finally, a further test was carried out using a commercial PMU and two PMUs emulated by the WAMPAC Characterizer connected to the OpenPDC, as shown in Fig. 6(c). The chosen commercial PMU uses an M-Class algorithm for synchrophasor estimation with RR $= 50$ fps and TCP packets of 92 bytes. Such PMU was characterized beforehand in terms of reporting latency, finding an average $\mu_{\text{PMU}} = 132.43$ ms and a standard deviation of 0.08 ms, for about 30 minutes, since the standard [2] requires at least 20 minutes of observation. Therefore, to make the scenario more realistic, the same latency was configured for the PMUs emulated by the Characterizer. The test results (not reported here for the sake of brevity) confirm the conclusions of previous tests (e.g., that in Fig. 12) while highlighting the Characterizer flexibility. Indeed, the average of $\widehat{L}_T$ is almost shifted by μ_{PMU} with respect to $\widehat{L}_{\text{WAMPAC}}$ of the above test, as expected, and the standard deviation is almost the same. Latency peaks are more erratic and depend on the specific test configuration, but the influence of L_{PMU} on the measurements is still evident.

The proposed measurement platform proved to be effective, with an accuracy well-suited for testing WAMPAC applications in real operating scenarios. This measurement instrument can then help TSOs during the prototyping and implementation phases, so that configuration and testing can rely on an accurately measured WAMPAC latency. This is a key factor, because, when WAMPAC latency is not assessed properly, automated control may not work and, in any case, cannot be optimized. The results also show that the GP version does not allow the accuracy required to verify real WAMPAC constraints, but it can help in preliminary tests.

V. Conclusion

This paper has proposed a WAMPAC Characterizer that is flexible and adaptable to the different types of WAMPAC systems. Its accuracy allows testing different WAMPAC scenarios, with several PMUs and with various RRs, packet configurations and communication network delays. While evaluating the WAMPAC system under test and its promptness, the proposed platform does not introduce any significant uncertainty-related decision risks. The proposed measurement platform can be employed for both short- and long-duration tests also when WAMPAC systems rely on UTC synchronization for their critical tasks and is thus a promising tool for TSOs at design, prototyping and pre-production stage.

References

[1] AA. VV., *Phasor Measurement Units and Wide Area Monitoring Systems*, 1st ed., A. Monti, C. Muscas, and F. Ponci, Eds. Academic Press, 2016.

[2] *IEEE/IEC International Standard - Measuring relays and protection equipment - Part 118-1: Synchrophasor for power systems - Measurements*, IEC/IEEE 60255-118-1:2018, Dec. 2018.

[3] *IEEE Standard for Phasor Data Concentrators for Power Systems*, IEEE Std C37.247-2019, Sep. 2019.

[4] *IEEE Standard for Synchrophasor Data Transfer for Power Systems*, IEEE Std C37.118.2-2011 (Revision of IEEE Std C37.118-2005), Dec. 2011.

[5] G. Frigo, A. Derviškadić, Y. Zuo, and M. Paolone, "PMU-based ROCOF measurements: Uncertainty limits and metrological significance in power system applications," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 10, pp. 3810–3822, 2019.

[6] V. Terzija *et al.*, "Wide-area monitoring, protection, and control of future electric power networks," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 80–93, 2011.

[7] M. Almas and L. Vanfretti, "Impact of time-synchronization signal loss on PMU-based WAMPAC applications," in *2016 IEEE Power and Energy Society General Meeting (PESGM)*, Boston, US, Jul. 2016.

[8] I. Ivanković, I. Kuzle, and N. Holjevac, "Multifunctional WAMPAC system concept for out-of-step protection based on synchrophasor measurements," *International Journal of Electrical Power & Energy Systems*, vol. 87, pp. 77–88, 2017.

[9] S. S. Negi, N. Kishor, K. Uhlen, and R. Negi, "Event detection and its signal characterization in PMU data stream," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 6, pp. 3108–3118, 2017.

[10] P. Castello, C. Muscas, P. A. Pegoraro, and S. Sulis, "Active phasor data concentrator performing adaptive management of latency," *Sustainable Energy, Grids and Networks*, vol. 16, pp. 270–277, 2018.

[11] A. Derviškadić, P. Romano, M. Pignati, and M. Paolone, "Architecture and experimental validation of a low-latency phasor data concentrator," *IEEE Transactions on Smart Grid*, vol. 9, no. 4, pp. 2885–2893, 2018.

[12] D. Anand et al. (2017, Jan.) NIST special publication 1500-08: Timing challenges in the smart grid. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.1500-08.pdf

[13] Y. Bansal and R. Sodhi, "A half-cycle fast discrete orthonormal S-transform-based protection-class μPMU," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 9, pp. 6934–6945, 2020.

[14] S. M. Blair, M. H. Syed, A. J. Roscoe, G. M. Burt, and J.-P. Braun, "Measurement and analysis of PMU reporting latency for smart grid protection and control applications," *IEEE Access*, vol. 7, pp. 48 689–48 698, 2019.

[15] S. M. Blair, N. Matheson, R. Munro, and C. D. Booth, "A new platform for validating real-time, large-scale WAMPAC systems," in *Protection, Automation and Control World Conference*, Glasgow, UK, Jun. 2019.

[16] F. Ye and A. Bose, "Multiple communication topologies for PMU-based applications: Introduction, analysis and simulation," *IEEE Transactions on Smart Grid*, vol. 11, no. 6, pp. 5051–5061, 2020.

[17] M. S. Thomas, N. Senroy, and A. S. Rana, "Analysis of time delay in a wide-area communication network," in *2014 6th IEEE Power India International Conference (PIICON)*, Delhi, IN, Dec. 2014.

[18] P. Castello, P. Ferrari, A. Flammini, C. Muscas, P. A. Pegoraro, and S. Rinaldi, "A distributed PMU for electrical substations with wireless redundant process bus," *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 5, pp. 1149–1157, 2015.

[19] I. Ivanković, V. Terzija, and S. Skok, "Transmission network angle stability protection based on synchrophasor data in control centre," *Journal of Energy*, vol. 67, no. 3, pp. 36–40, 2018.

[20] F. Zhang, Y. Sun, L. Cheng, X. Li, J. H. Chow, and W. Zhao, "Measurement and modeling of delays in wide-area closed-loop control systems," *IEEE Transactions on Power Systems*, vol. 30, no. 5, pp. 2426–2433, 2015.

[21] M. S. Almas and L. Vanfretti, "RT-HIL implementation of the hybrid synchrophasor and GOOSE-based passive islanding schemes," *IEEE Transactions on Power Delivery*, vol. 31, no. 3, pp. 1299–1309, 2016.

[22] M. Naglic, M. Popov, M. A. M. M. van der Meijden, and V. Terzija, "Synchro-measurement application development framework: An IEEE standard C37.118.2-2011 supported MATLAB library," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 8, pp. 1804–1814, 2018.

[23] M. Banafer and S. R. Mohanty, "Traveling wave-based primary protection and fault localization scheme for MTDC grid considering IEC 61869-9 measurement standard," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–16, 2023.

[24] EPRI. (2013) Synchrophasor communication infrastructure - Impacts on latency – Part II. [Online]. Available: https://www.epri.com/research/products/000000003002000604

[25] EPRI. (2012) Wide area monitoring, protection, and control systems (WAMPAC) - Standards for cyber security requirements. [Online]. Available: https://www.smartgrid.epri.com/doc/ESRFSD.pdf

[26] S. Vahidi, M. Ghafouri, M. Au, M. Kassouf, A. Mohammadi, and M. Debbabi, "Security of wide-area monitoring, protection, and control (WAMPAC) systems of the smart grid: A survey on challenges and opportunities," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 2, pp. 1294–1335, 2023.

[27] A. Ashok, M. Govindarasu, and J. Wang, "Cyber-physical attack-resilient wide-area monitoring, protection, and control for the power grid," *Proceedings of the IEEE*, vol. 105, no. 7, pp. 1389–1407, 2017.

[28] A. Musleh, C. Konstantinou, M. Sazos, A. Keliris, A. Al-Durra, and M. Maniatakos, "GPS spoofing effect on phase angle monitoring and control in an RTDS based hardware-in-the-loop environment," *IET Cyber-Physical Systems: Theory & Applications*, vol. 2, pp. 180–187, 2017.

[29] M. S. Almas, L. Vanfretti, R. S. Singh, and G. M. Jonsdottir, "Vulnerability of synchrophasor-based WAMPAC applications' to time synchronization spoofing," *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 4601–4612, 2018.

[30] Y. Wang and J. P. Hespanha, "Distributed estimation of power system oscillation modes under attacks on GPS clocks," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 7, pp. 1626–1637, 2018.

[31] P. Castello *et al.*, "Latency characterization of a wide area monitoring protection and control application in the italian transmission system," in *2022 IEEE 12th International Workshop on Applied Measurements for Power Systems (AMPS)*, Cagliari, IT, Sep. 2022.

[32] P. Castello, J. Liu, C. Muscas, P. A. Pegoraro, F. Ponci, and A. Monti, "A fast and accurate PMU algorithm for P+M class measurement of synchrophasor and frequency," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 12, pp. 2837–2845, 2014.

[33] P. Castello *et al.*, "An active phasor data concentrator suitable for control and protection applications," in *2019 International Conference on Smart Grid Synchronized Measurements and Analytics (SGSMA)*, College Station, TX, US, May 2019.

[34] *Communication networks and systems for power utility automation - Part 90-5: Use of IEC 61850 to transmit synchrophasor information according to IEEE C37.118*, IEC TR 61850-90-5:2012, May 2012.

[35] *Instrument transformers - Part 13: Stand-alone merging unit (SAMU)*, IEC 61869-13, Feb. 2021.

[36] S. M. S. Hussain, S. M. Farooq, and T. S. Ustun, "A security mechanism for IEEE C37.118.2 PMU communication," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 1, pp. 1053–1061, 2022.

[37] European Network of Transmission System Operators for Electricity (ENTSO-E), "Final report on the separation of the Continental Europe power system on 8 January 2021 - Main report," Tech. Rep., Jul. 2021.

[38] Grid Protection Alliance, OpenPDC, Access Date 30.05.2023. [Online]. Available: https://github.com/GridProtectionAlliance/openPDC

[39] P. Castello, C. Muscas, P. A. Pegoraro, and S. Sulis, "Monitoring system based on phasor measurement units with variable reporting rates," *ACTA IMEKO*, vol. 7, pp. 62–69, 2018.

[40] A. Nechifor, M. Albu, R. Hair, and V. Terzija, "An enterprise platform for wide area monitoring of load dynamics using synchronized measurement technology," *IFAC-PapersOnLine*, vol. 49, no. 27, pp. 85–90, 2016.

[41] K. Hammar and R. Stadler, "Intrusion prevention through optimal stopping," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2333–2348, 2022.

[42] S. Hemminger, "Network emulation with NetEm," in *Proc. Linux Conf.*, 2005.

[43] Mark Griffiths, NetTime, Access Date 30.05.2023. [Online]. Available: https://www.timesynctool.com

Paolo Castello (Member, IEEE) received the M.S. degree in electronic engineering and the Ph.D. degree in electronic and computer engineering from the University of Cagliari, Cagliari, Italy, in 2010 and 2014, respectively.

He is currently an Assistant Professor with the Department of Electrical and Electronic Engineering, University of Cagliari. He has co-authored 51 scientific articles published in international journals and conference proceedings. His research interests include an algorithm for synchrophasor estimation, synchronized and distributed measurement system, and characterization of measurement devices as phasor measurement units and power quality meters.

Dr. Castello is a member of the IEEE Instrumentation and Measurement Society

Giacomo Gallus (Student Member, IEEE) received the M.S. degree (*cum laude*) in Electronic Engineering from the University of Cagliari, Cagliari, Italy, in 2021, where he is currently pursuing the Ph.D. degree with the Electrical and Electronic Measurements Group of the Department of Electrical and Electronic Engineering. His research activities focus on synchrophasor measurements, synchronization techniques and Wide Area Monitoring, Protection and Control (WAMPAC) systems.

Paolo Attilio Pegoraro (Senior Member, IEEE) received the M.S. (*summa cum laude*) degree in telecommunication engineering and the Ph.D. degree in electronic and telecommunication engineering from the University of Padova, Padua, Italy, in 2001 and 2005, respectively.

From 2015 to 2018 he was an Assistant Professor with the Department of Electrical and Electronic Engineering, University of Cagliari, Cagliari, Italy, where he is currently Associate Professor. He has authored or co-authored over 140 scientific papers. His current research interests include the development of new measurement techniques for modern power networks, with attention to synchronized measurements and state estimation.

Dr. Pegoraro is a member of IEEE IMS TC 39 (Measurements in Power Systems) and of IEC TC 38/WG 47. He is an Associate Editor of the IEEE Transactions on Instrumentation and Measurement and the General Chair of the IEEE International Workshop on Applied Measurements for Power Systems (AMPS).

Sara Sulis (Senior Member, IEEE) received the M.S. degree in electrical engineering and the Ph.D. degree in industrial engineering from the University of Cagliari, Cagliari, Italy, in 2002 and 2006, respectively. She is currently an Associate Professor of electrical and electronic measurements with the University of Cagliari.

She has authored or co-authored more than 120 scientific articles. Her current research interests include distributed measurement systems designed to perform state estimation and harmonic sources estimation of distribution networks.

Dr. Sulis is a member of the Instrumentation and Measurement Society, the IEEE TC 39 “Measurements in Power Systems,” and the CENELEC TC 38 “Instrument Transformers.” She is an Associate Editor of the IEEE Transactions on Instrumentation and Measurement.