

**METHOD AND SYSTEM FOR CHARACTERISING A PROGRAMMABLE LOGIC
CONTROLLER (PLC) AND/OR ATTACK DETECTION IN A NETWORKED
CONTROL SYSTEM**

5

[0001] This application claims the benefit of priority of Singapore Patent Application No. 10202006737U, filed on 15 July 2020, the content of which being hereby incorporated by reference in its entirety for all purposes.

10

TECHNICAL FIELD

[0002] The present invention generally relates to a method and a system for characterising a programmable logic controller (PLC) in a networked control system, and a method and a system for attack detection in a networked control system.

15

BACKGROUND

[0003] An industrial control system (ICS) is a networked control system comprising sensors, actuators, controllers (i.e., programmable logic controllers (PLCs)) and communication networks configured to control one or more physical processes in an industry, such as water treatment, water distribution, smart grid, autonomous transportation, and so on. For example, an ICS uses sensors to remotely measure the system state and feed sensor measurements to PLCs. PLCs then send control actions to actuators based on the sensor measurements. PLCs also share local state measurements with other PLCs via a messaging protocol. However, industrial control systems (ICSs) are an attractive target for cyber attacks due to the critical nature of ICS infrastructures, and therefore, require security measures for safe operations. Recent research efforts in ICS security stem from a traditional IT infrastructure perspective. For example, network-based intrusion attack detection has been a proposed solution. However, such conventional network traffic based intrusion attack detection methods would fail when an attacker impersonates a PLC since there would be no change in network traffic patterns. Furthermore, conventional commercial ICS communication protocols may lack data integrity checks, resulting in no data integrity guarantees. For example, in a number of such conventional ICS communication protocols, no authentication measures are implemented, and hence, an attacker may manipulate data transmitted across the PLCs and field devices (e.g., actuators).

[0004] Although solutions grounded in cryptography, such as those that use transport layer security (TLS), hash-based message authentication codes (HMACs) or other authentication and/or integrity guarantees have been advocated in the context of ICS, historically, such countermeasures are not widespread due to limitations in hardware and relative computational cost of such protocols. Since many ICSs run legacy hardware, and may continue to do so indefinitely, the problem of raising the bar against authentication attacks by non-cryptographic means is a practical one. A recent study reported reveals that a large number of PLCs are connected to the Internet and contain vulnerabilities related to authentication. Also, the use of commercial off the shelf (COTS) devices in an ICS, and software backdoor, can lead to full control over PLCs. Stuxnet is a famous example of a malware attack where PLCs were hijacked and malicious code altered the PLC's configuration. A range of malware and network-based attacks were designed and executed against PLCs.

[0005] A need therefore exists to provide a method and a system for characterising a PLC in a networked control system, and a method and a system for attack detection in the networked control system, that seek to overcome, or at least ameliorate, problem(s) relating to conventional methods and systems for PLC authentication and/or attack detection in a networked control system, and more particularly, enabling or improving authentication of PLCs and/or attack detection in the networked control system in a practical and effective manner. It is against this background that the present invention has been developed.

SUMMARY

[0006] According to a first aspect of the present invention, there is provided a method of characterising a PLC in a networked control system using at least one processor, the networked control system comprising a plurality of PLCs, including the PLC, and a communication network layer based on which the plurality of PLCs communicate with each other, the method comprising:

- obtaining network traffic data from the communication network layer;
- determining scan cycle related timing profile information associated with the PLC based on the network traffic data obtained; and
- generating characterising information associated with the PLC based on the determined scan cycle related timing profile information for characterising the PLC.

[0007] According to a second aspect of the present invention, there is provided a PLC characterising system for characterising a PLC in a networked control system, the networked

control system comprising a plurality of PLCs, including the PLC, and a communication network layer based on which the plurality of PLCs communicate with each other, the PLC characterising system comprising:

a memory; and

5 at least one processor communicatively coupled to the memory and configured to perform the method of characterising the PLC in the networked control system according to the above-mentioned first aspect of the present invention.

[0008] According to a third aspect of the present invention, there is provided a method of attack detection in a networked control system using at least one processor, the networked
10 control system comprising a plurality of PLCs and a communication network layer based on which the plurality of PLCs to communicate with each other, the method comprising:

obtaining second network traffic data from the communication network layer;

determining second scan cycle related timing profile information associated with a PLC of the plurality of PLCs based on the second network traffic data obtained;

15 generating second characterising information associated with the PLC based on the determined second scan cycle related timing profile information; and

detecting whether the networked control system is subject to an attack in relation to the PLC based on the second characterising information associated with the PLC and reference characterising information associated with the PLC.

20 **[0009]** According to a fourth aspect of the present invention, there is provided an attack detection system for attack detection in a networked control system using at least one processor, the networked control system comprising a plurality of PLCs and a communication network layer based on which the plurality of PLCs to communicate with each other. The attack detection system comprising: a memory; and at least one processor communicatively coupled
25 to the memory and configured to perform the method of attack detection in the networked control system according to the above-mentioned third aspect of the present invention.

[0010] According to a fifth aspect of the present invention, there is provided a computer program product, embodied in one or more non-transitory computer-readable storage mediums, comprising instructions executable by at least one processor to perform the method of
30 characterising a PLC in a networked control system according to the above-mentioned first aspect of the present invention.

[0011] According to a sixth aspect of the present invention, there is provided a computer program product, embodied in one or more non-transitory computer-readable storage mediums,

comprising instructions executable by at least one processor to perform the method of attack detection in a networked control system according to the above-mentioned third aspect of the present invention.

5 BRIEF DESCRIPTION OF THE DRAWINGS

[0012] Embodiments of the present invention will be better understood and readily apparent to one of ordinary skill in the art from the following written description, by way of example only, and in conjunction with the drawings, in which:

FIG. 1 depicts a schematic flow diagram of a method of characterising a PLC in a
10 networked control system, according to various embodiments of the present invention;

FIG. 2 depicts a schematic block diagram of a PLC characterising system for characterising a PLC in a networked control system, according to various embodiments of the present invention;

FIG. 3 depicts a schematic flow diagram of a method of attack detection in a networked
15 control system using, according to various embodiments of the present invention;

FIG. 4 depicts a schematic block diagram of an attack detection system for attack detection in a networked control system, according to various embodiments of the present invention;

FIG. 5 depicts a schematic block diagram of an exemplary computer system which may
20 be used to realize or implement the PLC characterising system and the attack detection system (or the PLC characterising and attack detection system), respectively, according to various embodiments of the present invention;

FIG. 6 depicts a schematic drawing showing an overview of an example ICS network architecture, according to various example embodiments of the present invention;

FIG. 7 depicts an example logical flow of steps involved during a PLC scan cycle, along
25 with an example ladder logic, according to various example embodiments of the present invention;

FIG. 8 depicts a table (Table 1) presenting results for MSG instructions from/to PLCs and their respective timing analysis, according to various example embodiments of the present
30 invention;

FIG. 9 depicts a schematic drawing illustrating message queuing in a PLC, according to various example embodiments of the present invention;

FIG. 10 depicts an overview of a method for device identification and attack detection, according to various example embodiments of the present invention;

FIG. 11 depicts a table (Table 2) presenting a list of example features that may be extracted, individually or in combination, according to various example embodiments of the present invention;

FIG. 12 depicts plots showing statistical features of the estimated scan cycle vector for three PLCs in the SWaT testbed, according to various example embodiments of the present invention;

FIG. 13 depicts a table (Table 3) presenting results of the multiclass classification with respect to chunk size vs classification accuracy, according to various example embodiments of the present invention;

FIG. 14 depicts a table (Table 4) presenting results of k-fold cross validation using multi-class classifier, according to various example embodiments of the present invention;

FIG. 15 depicts a table (Table 5) presenting results showing stability from run-to-run and across temperature range, whereby data from each PLC is labeled as class-1 and all the data from rest of the five PLCs are labeled as class-2, according to various example embodiments of the present invention;

FIG. 16 depicts a table (Table 6) showing the accuracy improvement due to change in the control logic of PLC 4, whereby DF denotes Default Profiles and MF denotes Modified Profiles, according to various example embodiments of the present invention;

FIG. 17 depicts a table (Table 7) presenting results for a multi-class classification in relation to the EPIC Testbed Performance Evaluation, according to various example embodiments of the present invention;

FIG. 18 depicts a table (Table 8) showing the attack detection performance, whereby MN denotes Masquerader Naïve; MPDK denotes Masquerader Partial Distribution Knowledge; and MFDK denotes Masquerader Full Distribution Knowledge, according to various example embodiments of the present invention;

FIG. 19 depicts plots for the estimated scan cycle for PLC 1 under attack and normal operation, according to various example embodiments of the present invention;

FIGS. 20A to 20C show estimated scan cycle time profiles of six PLCs under a powerful masquerade attack, according to various example embodiments of the present invention;

FIG. 21 shows a plot of ROC for Masquerader Partial Distribution Knowledge, according to various example embodiments of the present invention;

FIG. 22 shows a plot of ROC for MFDK Masquerader Full Distribution Knowledge, according to various example embodiments of the present invention;

FIG. 23 illustrates an example watermark for PLC 1 in SWaT, according to various example embodiments of the present invention;

5 FIG. 24 depicts an example design or configuration of a PLC watermarking method, according to various example embodiments of the present invention;

FIG. 25 depicts an example request-response messaging in the PLCs, according to various example embodiments of the present invention;

10 FIG. 26 depicts a table (Table 9) presenting the K-S test results for normal data in relation to chunk size vs classification accuracy for PLC 1, according to various example embodiments of the present invention;

FIG. 27 depicts Gaussian distribution approximation of PLC1 under normal operation for different runs of the experiment, according to various example embodiments of the present invention;

15 FIG. 28 depicts a table (Table 10) presenting K-S test results with and without adding the watermark, with accuracy for PLC 1 for a chunk size of 120 samples shown, according to various example embodiments of the present invention;

FIG. 29 depicts a closed loop feedback system model configured to model a process of message exchange between PLC request and response messages, according to various example
20 embodiments of the present invention;

FIG. 30 depicts a plot of empirical cumulative distributions for a masquerade attack on PLC1 and watermark, according to various example embodiments of the present invention;

FIG. 31 depicts a table (Table 11) presenting results for the K-S test for attack data vs watermark attack detection accuracy for all PLCs in the SWaT testbed for a chunk size of 120
25 samples, whereby MFDK denotes Masquerade Full Distribution Knowledge;

FIG. 32 depicts the response time for a message instruction in PLC3 in a SWaT testbed, according to various example embodiments of the present invention;

FIG. 33 depicts results from an experiment with a watermarked delay of 40 ms, according to various example embodiments of the present invention; and

30 FIG. 34 depicts a masquerade attack and watermark time series, according to various example embodiments of the present invention.

DETAILED DESCRIPTION

[0013] Various embodiments of the present invention provide a method and a system for characterising a programmable logic controller (PLC) in a networked control system, and a method and a system for attack detection in a networked control system. In various
5 embodiments, the method of characterising a PLC in a networked control system and the method of attack detection in a networked control system may be combined or integrated as a method of characterising a PLC and attack detection in a networked control system. Similarly, in various embodiments, the system (PLC characterising system) for characterising a PLC in a networked control system and the system (attack detection system) for attack detection in a
10 networked control system may be combined or integrated as a system (PLC characterising and attack detection system) for characterising a PLC and attack detection in a networked control system.

[0014] For example, the networked control system may be implemented in any industries (industrial applications) as an industrial control system (ICS) as desired or as appropriate that
15 requires an industrial process control, such but not limited to, water treatment, water distribution, smart grid, autonomous transportation, and so on. The networked control system may comprise sensors, actuators, controllers (i.e., programmable logic controllers (PLCs)) and communication networks configured to control one or more physical processes in an industry. For example, an ICS uses sensors to remotely measure the system state and feed sensor
20 measurements to PLCs. PLCs then send control actions to actuators based on the sensor measurements. PLCs also share local state measurements with other PLCs via a messaging protocol. However, as explained in the background, ICSs are an attractive target for cyber attacks due to the critical nature of ICS infrastructures, and therefore, require security measures for safe operations. In this regard, as discussed in the background, conventional methods and
25 systems for attack detection in a networked control system suffer from various problems such as practicality and/or effectiveness problems. Accordingly, various embodiments of the present invention provide a method and a system for characterising a PLC in a networked control system, and a method and a system for attack detection in the networked control system, that seek to overcome, or at least ameliorate, problem(s) relating to conventional methods and
30 systems for PLC authentication and/or attack detection in a networked control system, and more particularly, enabling or improving authentication of PLCs and/or attack detection in the networked control system in a practical and effective manner.

[0015] In various embodiments, an attack on the networked control system may refer to any type of security or malicious attack on the networked control system known in the art and need not be described herein, such as an attack on one or more PLCs in the networked control system. Various possible types of attack on a networked control system are known in the art, and for illustration purpose, example types of attack will be described later below according to various example embodiments of the present invention.

[0016] FIG. 1 depicts a schematic flow diagram of a method 100 of characterising a PLC in a networked control system using at least one processor, according to various embodiments of the present invention. The networked control system comprising a plurality of PLCs, including the above-mentioned PLC, and a communication network layer based on which the plurality of PLCs communicate with each other. The method 100 comprises: obtaining (at 102), obtaining network traffic data from the communication network layer; determining (at 104) scan cycle related timing profile information associated with the PLC based on the network traffic data obtained; and generating (at 106) characterising information (which may also be referred to as characteristic information) associated with the PLC based on the determined scan cycle related timing profile information for characterising the PLC.

[0017] Accordingly, the method 100 of characterising a PLC in a networked control system is advantageously based on scan cycle related timing profile information associated with the PLC, which has been found to be based on hardware and software characteristics of the PLC, thereby enabling or improving authentication of the PLC in a practical and effective manner. These advantages or technical effects, and/or other advantages or technical effects, will become more apparent to a person skilled in the art as the method 100 of characterising a PLC, as well as corresponding PLC characterising system for characterising a PLC, is described in more detail according to various embodiments and example embodiments of the present invention

[0018] In various embodiments, the scan cycle related timing profile information associated with the PLC comprises timing information associated with a series of request messages indicated as sent by the PLC to the communication network layer in the network traffic data obtained. For example, the series of request messages may be a sample of request messages indicated as sent by the PLC obtained in the network traffic data. In various embodiments, each of the series of request messages indicated as sent by the PLC may have stored therein or assigned thereto an identity of the PLC that sent the request message.

- [0019] In various embodiments, the timing information associated with the series of request messages indicated as sent by the PLC is inter message timing information associated with the series of request messages indicated as sent by the PLC.
- [0020] In various embodiments, the method 100 further comprises introducing a time delay to each of a plurality of request messages sent by the PLC.
- [0021] In various embodiments, the time delay is a predetermined time delay.
- [0022] In various embodiments, the time delay is a random time delay.
- [0023] In various embodiments, the random time delay is determined based on a clock of the PLC.
- [0024] In various embodiments, the above-mentioned generating (at 106) characterising information associated with the PLC comprises extracting one or more time and/or frequency domain features from the timing information associated with the series of request messages indicated as sent by the PLC. In this regard, the characterising information comprises the extracted one or more time and/or frequency domain features.
- [0025] In various embodiments, the extracted one or more time and/or frequency domain features is one or more time and frequency domain features.
- [0026] In various embodiments, the set of request messages has a sample size in a range of 30 to 500. In various embodiments, the sample size may be in a range of 60 to 200 or 100 to 150. In various embodiments, the sample size may be about 120.
- [0027] In various embodiments, the PLC is configured to recursively perform a scan cycle comprising an input reading stage, a control logic execution stage, and an output writing stage. In this regard, request messages sent by the PLC are sent at the control logic execution stage.
- [0028] In various embodiments, the plurality of PLCs are configured to communicate with each other based on a request-response model.
- [0029] In various embodiments, the method 100 further comprises: labeling the extracted one or more time and/or frequency domain features of the characterising information with an identity information of the PLC to produce labeled features associated with the PLC; and training a machine learning model based on the labeled features associated with the PLC to produce a PLC classifier configured for PLC identification.
- [0030] FIG. 2 depicts a schematic block diagram of a PLC characterising system 200 for characterising a PLC in a networked control system, according to various embodiments of the present invention, corresponding to the method 100 of characterising a PLC in a networked control system as described hereinbefore with reference to FIG. 1 according to various

embodiments of the present invention. The PLC characterising system 200 comprises: a memory 202; and at least one processor 204 communicatively coupled to the memory 202 and configured to perform the method 100 of characterising the PLC in the networked control system as described herein according to various embodiments of the present invention.

5 Accordingly, in various embodiments, the at least one processor 204 is configured to: obtain network traffic data from the communication network layer; determine scan cycle related timing profile information associated with the PLC based on the network traffic data obtained; and generate characterising information associated with the PLC based on the determined scan cycle related timing profile information for characterising the PLC.

10 **[0031]** It will be appreciated by a person skilled in the art that the at least one processor 204 may be configured to perform various functions or operations through set(s) of instructions (e.g., software modules) executable by the at least one processor 204 to perform various functions or operations. Accordingly, as shown in FIG. 2, the system 200 may comprise a network traffic data module (or a network traffic data circuit) 206 configured to obtain network
15 traffic data from the communication network layer; a scan cycle related timing profile information determining module (or scan cycle related timing profile information determining circuit) 208 configured to determine scan cycle related timing profile information associated with the PLC based on the network traffic data obtained; and a characterising information generating module (a characterising information generating circuit) 210 configured to generate
20 characterising information associated with the PLC based on the determined scan cycle related timing profile information for characterising the PLC.

[0032] It will be appreciated by a person skilled in the art that the above-mentioned modules are not necessarily separate modules, and one or more modules may be realized by or implemented as one functional module (e.g., a circuit or a software program) as desired or as
25 appropriate without deviating from the scope of the present invention. For example, two or more of the network traffic data module 206, the scan cycle related timing profile information determining module 208 and the characterising information generating module 210 may be realized (e.g., compiled together) as one executable software program (e.g., software application or simply referred to as an “app”), which for example may be stored in the memory
30 202 and executable by the at least one processor 204 to perform various functions/operations as described herein according to various embodiments of the present invention.

[0033] In various embodiments, the PLC characterising system 200 corresponds to the method 100 of characterising a PLC as described hereinbefore with reference to FIG. 1,

therefore, various functions or operations configured to be performed by the least one processor 204 may correspond to various steps or operations of the method 100 of characterising a PLC as described herein according to various embodiments, and thus need not be repeated with respect to the PLC characterising system 200 for clarity and conciseness. In other words, various embodiments described herein in context of the methods are analogously valid for the corresponding systems, and vice versa.

[0034] For example, in various embodiments, the memory 202 may have stored therein the network traffic data module 206, the scan cycle related timing profile information determining module 208 and/or the characterising information generating module 210, which respectively correspond to various steps (or operations or functions) of the method 100 of characterising a PLC as described herein according to various embodiments, which are executable by the at least one processor 204 to perform the corresponding functions or operations as described herein.

[0035] FIG. 3 depicts a schematic flow diagram of a method 300 of attack detection in a networked control system using at least one processor, according to various embodiments of the present invention. Similarly, the networked control system comprises a plurality of PLCs and a communication network layer based on which the plurality of PLCs to communicate with each other. The method 300 comprises: obtaining (at 302) second network traffic data from the communication network layer; determining (at 304) second scan cycle related timing profile information associated with a PLC of the plurality of PLCs based on the second network traffic data obtained; generating (at 306) second characterising information (which may also be referred to as second characterising information) associated with the PLC based on the determined second scan cycle related timing profile information; and detecting (at 308) whether the networked control system is subject to an attack in relation to the PLC based on the second characterising information associated with the PLC and reference characterising information associated with the PLC.

[0036] In various embodiments, the reference characterising information associated with the PLC is characterising information generated according to the method 100 of characterising a PLC as described herein according to various embodiments.

[0037] In various embodiments, in the same or similar manner as the method 100, the second scan cycle related timing profile information associated with the PLC comprises second timing information associated with a second series of request messages indicated as sent by the PLC to the communication network layer in the second network traffic data obtained. In this regard, a request message sent by the PLC would be indicated as sent by the PLC, and

furthermore, for example, a request message sent by an attacker imitating the PLC may also be indicated (faked by the attacker) as sent by the PLC.

5 [0038] In various embodiments, in the same or similar manner as the method 100, the second timing information associated with the second series of request messages sent by the PLC is inter message timing information associated with the second series of request messages indicated as sent by the PLC.

[0039] In various embodiments, in the same or similar manner as the method 100, the method 300 further comprises introducing a time delay to each of a second plurality of request messages sent by the PLC.

10 [0040] In various embodiments, in the same or similar manner as the method 100, the time delay is a predetermined time delay.

[0041] In various embodiments, in the same or similar manner as the method 100, the time delay is a random time delay.

15 [0042] In various embodiments, in the same or similar manner as the method 100, the random time delay is determined based on a clock of the PLC.

[0043] In various embodiments, in the same or similar manner as the method 100, the above-mentioned generating (at 306) second characterising information associated with the PLC comprises extracting one or more second time and/or frequency domain features from the second timing information associated with the second series of request messages indicated as
20 sent by the PLC. In this regard, the second characterising information comprises the extracted one or more second time and/or frequency domain features.

[0044] In various embodiments, in the same or similar manner as the method 100, the extracted one or more second time and/or frequency domain features is one or more time and frequency domain features.

25 [0045] In various embodiments, in the same or similar manner as the method 100, the second series of request messages has a sample size in a range of 30 to 500. In various embodiments, the sample size may be in a range of 60 to 250 or 100 to 150. In various embodiments, the sample size may be about 120.

30 [0046] In various embodiments, in the same or similar manner as the method 100, the PLC is configured to recursively perform a scan cycle comprising an input reading stage, a control logic execution stage, and an output writing stage. In this regard, request messages sent by the PLC are sent at the control logic execution stage.

[0047] In various embodiments, in the same or similar manner as the method 100, the plurality of PLCs are configured to communicate with each other based on a request-response model.

5 [0048] In various embodiments, the above-mentioned detecting (at 308) whether the networked control system is subject to an attack in relation to the PLC is based on the second characterising information associated with the PLC and the PLC classifier produced according to the method 100 of characterising a PLC as described herein according to various embodiments.

10 [0049] In various embodiments, the above-mentioned detecting (at 308) whether the networked control system is subject to an attack in relation to the PLC comprises: producing, by the PLC classifier, a classification result based on the second characterising information associated with the PLC; and determining whether the networked control system is subject to an attack in relation to the PLC based on the classification result from the PLC classifier.

15 [0050] FIG. 4 depicts a schematic block diagram of an attack detection system 400 for attack detection in a networked control system, according to various embodiments of the present invention, corresponding to the method 300 of attack detection in a networked control system as described hereinbefore with reference to FIG. 3 according to various embodiments of the present invention. The attack detection system 400 comprises: a memory 402; and at least one processor 404 communicatively coupled to the memory 402 and configured to perform the
20 method 300 of attack detection in the networked control system as described herein according to various embodiments of the present invention. Accordingly, in various embodiments, the at least one processor 404 is configured to: obtain second network traffic data from the communication network layer; determine second scan cycle related timing profile information associated with a PLC of the plurality of PLCs based on the second network traffic data
25 obtained; generate second characterising information associated with the PLC based on the determined second scan cycle related timing profile information; and detect whether the networked control system is subject to an attack in relation to the PLC based on the second characterising information associated with the PLC and reference characterising information associated with the PLC.

30 [0051] In the same or similar manner as the PLC characterising system 200 as described hereinbefore, it will be appreciated by a person skilled in the art that the at least one processor 404 may be configured to perform various functions or operations through set(s) of instructions (e.g., software modules) executable by the at least one processor 404 to perform various

functions or operations. Accordingly, as shown in FIG. 4, the attack detection system 400 may comprise a network traffic data module (or a network traffic data circuit) 406 configured to obtain second network traffic data from the communication network layer; a scan cycle related timing profile information determining module (or scan cycle related timing profile information determining circuit) 408 configured to determine second scan cycle related timing profile information associated with a programmable logic controller (PLC) of the plurality of PLCs based on the second network traffic data obtained; a characterising information generating module (a characterising information generating circuit) 410 configured to generate second characterising information associated with the PLC based on the determined second scan cycle related timing profile information; and an attack detection module (or an attack detection circuit) 412 configured to detect whether the networked control system is subject to an attack in relation to the PLC based on the second characterising information associated with the PLC and reference characterising information associated with the PLC.

[0052] In the same or similar manner as the PLC characterising system 200 as described hereinbefore, it will be appreciated by a person skilled in the art that the above-mentioned modules of the attack detection system 400 are not necessarily separate modules, and one or more modules may be realized by or implemented as one functional module (e.g., a circuit or a software program) as desired or as appropriate without deviating from the scope of the present invention. For example, two or more of the network traffic data module 406, the scan cycle related timing profile information determining module 408, the characterising information generating module 410 and the attack detection module 412 may be realized (e.g., compiled together) as one executable software program (e.g., software application or simply referred to as an “app”), which for example may be stored in the memory 402 and executable by the at least one processor 404 to perform various functions or operations as described herein according to various embodiments of the present invention.

[0053] In various embodiments, the attack detection system 400 corresponds to the method 300 of attack detection as described hereinbefore with reference to FIG. 3, therefore, various functions or operations configured to be performed by the least one processor 404 may correspond to various steps or operations of the method 300 of attack detection as described herein according to various embodiments, and thus need not be repeated with respect to the attack detection system 400 for clarity and conciseness. In other words, various embodiments described herein in context of the methods are analogously valid for the corresponding systems, and vice versa.

[0054] For example, in various embodiments, the memory 402 may have stored therein the network traffic data module 406, the scan cycle related timing profile information determining module 408, the characterising information generating module 410 and the attack detection module 412, which respectively correspond to various steps (or operations or functions) of the method 300 of attack detection as described herein according to various embodiments, which are executable by the at least one processor 404 to perform the corresponding functions or operations as described herein.

[0055] Furthermore, as described hereinbefore in various embodiments, the method 100 of characterising a PLC in a networked control system and the method 300 of attack detection in a networked control system may be combined or integrated as a method of characterising a PLC and attack detection in a networked control system. Similarly, in various embodiments, the PLC characterising system 200 for characterising a PLC in a networked control system and the attack detection system 400 for attack detection in a networked control system may be combined or integrated as a PLC characterising and attack detection system for characterising a PLC and attack detection in a networked control system. Accordingly, in the PLC characterising and attack detection system, it will be understood by a person skilled in the art that various components or modules of the PLC characterising system 200 and the attack detection system 400 may be combined or integrated together as desired or as appropriate. For example, it will be appreciated by a person skilled in the art that the memory 202 and the memory 402 may be realised by the same component, the processor 202 and the processor 404 may be realised by the same component, the network traffic data module 206 and the network traffic data module 406 may be realised by the same module, the scan cycle related timing profile information determining module 208 and the scan cycle related timing profile information determining module 408 may be realised by the same module, and the characterising information generating module 210 and the characterising information generating module 410 may be realised by the same module.

[0056] A computing system, a controller, a microcontroller or any other system providing a processing capability may be provided according to various embodiments in the present disclosure. Such a system may be taken to include one or more processors and one or more computer-readable storage mediums. For example, the PLC characterising system 200 described hereinbefore may include a processor (or controller) 204 and a computer-readable storage medium (or memory) 202 which are for example used in various processing carried out therein as described herein. A memory or computer-readable storage medium used in various

embodiments may be a volatile memory, for example a DRAM (Dynamic Random Access Memory) or a non-volatile memory, for example a PROM (Programmable Read Only Memory), an EPROM (Erasable PROM), EEPROM (Electrically Erasable PROM), or a flash memory, e.g., a floating gate memory, a charge trapping memory, an MRAM (Magnetoresistive Random Access Memory) or a PCRAM (Phase Change Random Access Memory).

[0057] In various embodiments, a “circuit” may be understood as any kind of a logic implementing entity, which may be special purpose circuitry or a processor executing software stored in a memory, firmware, or any combination thereof. Thus, in an embodiment, a “circuit” may be a hard-wired logic circuit or a programmable logic circuit such as a programmable processor, e.g., a microprocessor (e.g., a Complex Instruction Set Computer (CISC) processor or a Reduced Instruction Set Computer (RISC) processor). A “circuit” may also be a processor executing software, e.g., any kind of computer program, e.g., a computer program using a virtual machine code, e.g., Java. Any other kind of implementation of the respective functions which will be described in more detail below may also be understood as a “circuit” in accordance with various alternative embodiments. Similarly, a “module” may be a portion of a system according to various embodiments in the present invention and may encompass a “circuit” as above, or may be understood to be any kind of a logic-implementing entity therefrom.

[0058] Some portions of the present disclosure are explicitly or implicitly presented in terms of algorithms and functional or symbolic representations of operations on data within a computer memory. These algorithmic descriptions and functional or symbolic representations are the means used by those skilled in the data processing arts to convey most effectively the substance of their work to others skilled in the art. An algorithm is here, and generally, conceived to be a self-consistent sequence of steps leading to a desired result. The steps are those requiring physical manipulations of physical quantities, such as electrical, magnetic or optical signals capable of being stored, transferred, combined, compared, and otherwise manipulated.

[0059] Unless specifically stated otherwise, and as apparent from the following, it will be appreciated that throughout the present specification, description or discussions utilizing terms such as “obtaining”, “producing”, “determining”, “generating”, “introducing”, “labeling”, “training”, “detecting”, “characterising” or the like, refer to the actions and processes of a computer system, or similar electronic device, that manipulates and transforms data represented as physical quantities within the computer system into other data similarly represented as

physical quantities within the computer system or other information storage, transmission or display devices.

[0060] The present specification also discloses a system (e.g., which may also be embodied as a device or an apparatus), such as the PLC characterising system 200 and the attack detection system 400, for performing various operations/functions of various methods described herein. Such a system may be specially constructed for the required purposes, or may comprise a general purpose computer or other device selectively activated or reconfigured by a computer program stored in the computer. The algorithms presented herein are not inherently related to any particular computer or other apparatus. Various general-purpose machines may be used with computer programs in accordance with the teachings herein. Alternatively, the construction of more specialized apparatus to perform various method steps may be appropriate.

[0061] In addition, the present specification also at least implicitly discloses a computer program or software/functional module, in that it would be apparent to the person skilled in the art that individual steps of various methods described herein may be put into effect by computer code. The computer program is not intended to be limited to any particular programming language and implementation thereof. It will be appreciated that a variety of programming languages and coding thereof may be used to implement the teachings of the disclosure contained herein. Moreover, the computer program is not intended to be limited to any particular control flow. There are many other variants of the computer program, which can use different control flows without departing from the scope of the invention. It will be appreciated by a person skilled in the art that various modules described herein (e.g., the network traffic data module 206, the scan cycle related timing profile information determining module 208 and/or the characterising information generating module 210 with respect to the PLC characterising system 200) may be software module(s) realized by computer program(s) or set(s) of instructions executable by a computer processor to perform the required functions, or may be hardware module(s) being functional hardware unit(s) designed to perform the required functions. It will also be appreciated that a combination of hardware and software modules may be implemented.

[0062] Furthermore, one or more of the steps of a computer program/module or method described herein may be performed in parallel rather than sequentially. Such a computer program may be stored on any computer readable medium. The computer readable medium may include storage devices such as magnetic or optical disks, memory chips, or other storage devices suitable for interfacing with a general purpose computer. The computer program when

loaded and executed on such a general-purpose computer effectively results in an apparatus that implements the steps of the methods described herein.

[0063] In various embodiments, there is provided a computer program product, embodied in one or more computer-readable storage mediums (non-transitory computer-readable storage medium(s)), comprising instructions (e.g., the network traffic data module 206, the scan cycle related timing profile information determining module 208 and/or the characterising information generating module 210) executable by one or more computer processors to perform the method 100 of characterising a PLC in a networked control system, as described herein with reference to FIG. 1 according to various embodiments. Accordingly, various computer programs or modules described herein may be stored in a computer program product receivable by a system therein, such as the PLC characterising system 200 as shown in FIG. 2, for execution by at least one processor 204 of the system 200 to perform various functions.

[0064] In various embodiments, there is provided a computer program product, embodied in one or more computer-readable storage mediums (non-transitory computer-readable storage medium(s)), comprising instructions (e.g., the network traffic data module 406, the scan cycle related timing profile information determining module 408, the characterising information generating module 410 and the attack detection module 412) executable by one or more computer processors to perform the method 300 of attack detection in a networked control system, as described herein with reference to FIG. 3 according to various embodiments. Accordingly, various computer programs or modules described herein may be stored in a computer program product receivable by a system therein, such as the attack detection system 400 as shown in FIG. 4, for execution by at least one processor 404 of the system 400 to perform various functions.

[0065] Software or functional modules described herein may also be implemented as hardware modules. More particularly, in the hardware sense, a module is a functional hardware unit designed for use with other components or modules. For example, a module may be implemented using discrete electronic components, or it can form a portion of an entire electronic circuit such as an Application Specific Integrated Circuit (ASIC). Numerous other possibilities exist. Those skilled in the art will appreciate that the software or functional module(s) described herein can also be implemented as a combination of hardware and software modules.

[0066] In various embodiments, the PLC characterising system 200 and the attack detection system 400 (or the PLC characterising and attack detection system) may each be realized by

any computer system (e.g., desktop or portable computer system) including at least one processor and a memory, such as a computer system 500 as schematically shown in FIG. 5 as an example only and without limitation. Various methods/steps or functional modules may be implemented as software, such as a computer program being executed within the computer system 500, and instructing the computer system 500 (in particular, one or more processors therein) to conduct various functions or operations as described herein according to various embodiments. The computer system 500 may comprise a computer module 502, input modules, such as a keyboard and/or a touchscreen 504 and a mouse 506, and a plurality of output devices such as a display 508, and a printer 510. The computer module 502 may be connected to a computer network 512 via a suitable transceiver device 514, to enable access to e.g., the Internet or other network systems such as Local Area Network (LAN) or Wide Area Network (WAN). The computer module 502 in the example may include a processor 518 for executing various instructions, a Random Access Memory (RAM) 520 and a Read Only Memory (ROM) 522. The computer module 502 may also include a number of Input/Output (I/O) interfaces, for example I/O interface 524 to the display 508, and I/O interface 526 to the keyboard 504. The components of the computer module 502 typically communicate via an interconnected bus 528 and in a manner known to the person skilled in the relevant art.

[0067] It will be appreciated by a person skilled in the art that the terminology used herein is for the purpose of describing various embodiments only and is not intended to be limiting of the present invention. As used herein, the singular forms "a", "an" and "the" are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will be further understood that the terms "comprises" and/or "comprising," when used in this specification, specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components, and/or groups thereof.

[0068] Any reference to an element or a feature herein using a designation such as "first", "second" and so forth does not limit the quantity or order of such elements or features, unless stated or the context requires otherwise. For example, such designations may be used herein as a convenient way of distinguishing between two or more elements or instances of an element. Thus, a reference to first and second elements does not necessarily mean that only two elements can be employed, or that the first element must precede the second element. In addition, a phrase referring to "at least one of" a list of items refers to any single item therein or any combination of two or more items therein.

[0069] In order that the present invention may be readily understood and put into practical effect, various example embodiments of the present invention will be described hereinafter by way of examples only and not limitations. It will be appreciated by a person skilled in the art that the present invention may, however, be embodied in various different forms or configurations and should not be construed as limited to the example embodiments set forth hereinafter. Rather, these example embodiments are provided so that this disclosure will be thorough and complete, and will fully convey the scope of the present invention to those skilled in the art.

[0070] Various example embodiments provide timing-based authentication on PLCs, and more particularly, scan cycle related timing profile based authentication on PLCs.

[0071] PLCs are a core component of an ICS. However, if a PLC is compromised or the commands sent across a network from the PLCs are spoofed, consequences may be catastrophic. In this regard, various example embodiments provide a method to authenticate PLCs (e.g., corresponding to the method 100 of characterising a PLC as described hereinbefore with reference to FIG. 1 according to various embodiments) that seeks to raise the bar against powerful attackers while being compatible with real-time systems. In various example embodiments, the method captures timing information for each PLC in a non-invasive manner. In this regard, various example embodiments note that the scan cycle is a unique feature of a PLC and the method is based on obtaining scan cycle related timing profile information associated with a PLC (which may be referred to as estimated scan cycle timing profile information associated with a PLC) passively by observing network traffic. For example, an attacker that spoofs commands issued by a PLC would deviate from the corresponding fingerprint produced. In various example embodiments, a PLC watermarking method is provided to detect replay attacks. According to various example embodiments, PLC watermarking may model the relation between the scan cycle and the control logic by modeling the input/output as a function of request/response messages of a PLC. To demonstrate the practicality and effectiveness of timing-based authentication methods on PLCs according to various example embodiments, such methods were also validated on an operational water treatment plant (SWaT) and smart grid (EPIC) testbeds, which will be described later below. The results obtained from experiments conducted demonstrate that PLCs can advantageously be distinguished based on their scan cycle timing characteristics according to various example embodiments.

[0072] According to various example embodiments, to enhance authentication in PLCs non-invasively and without disturbing their core functionality, two authentication methods are provided, namely, PLC fingerprinting and and PLC watermarking.

[0073] In relation to PLC fingerprinting, the PLC fingerprint is a function of its hardware and control functionality, that is, the timing characteristics of a PLC. In this regard, various example embodiments note that there is a unique feature of PLCs known as scan cycle. A scan cycle refers to the periodic execution of the PLC logic and input/output (I/O) read/write. In various example embodiments, this unique feature of a PLC is utilized to form a fingerprint of the PLC (e.g., corresponding to the characterising information associated with the PLC as described hereinbefore according to various embodiments). In particular, various example embodiments seek to create a fingerprint in a passive manner without disturbing the PLC's core functionality. In various example embodiments, scan cycle timing is estimated (or scan cycle related timing information is determined) in a non-invasive manner by monitoring the messages which are being exchanged between the PLCs. For example, uniqueness in the fingerprint is due to the hardware components such as clock, processor, I/O registers, and logic components, for example, control logic, message queuing, and so on. For example, an adversary may send malicious messages either by using an external device connected to the ICS network, or as Man-in-The-Middle (MiTM), to modify the messages, or from outside of the system to perform DoS attacks. These attacks, even if launched by a knowledgeable attacker, would be detected according to various example embodiments of the present invention since the timing profile resulting from the injected data would not match the reference pattern representative of the unique characteristics of a PLC. In general, it is shown that any attack on PLC messages is able to be detected according to various example embodiments of the present invention if it changes the statistics of estimated scan cycle timing distribution.

[0074] In relation PLC watermarking, a method is provided to detect advanced replay and masquerade attacks. In various example embodiments, the PLC watermarking is built on top of scan cycle time estimation and the dependency of such an estimate on the control logic. In various example embodiments, a random delay injected in the control logic and a PLC watermark may be reflected in the estimated scan cycle time. This leads to the detection of powerful masquerade and replay attacks because PLC watermark behaves as a nonce. Experimental results on a real-world water treatment (SWaT) testbed demonstrate the practicality and effectiveness in fingerprinting (or characterising) the timing pattern for PLC identification and attack detection. In various example embodiments, experiments were

performed on a total of six Allen Bradley PLCs available in the SWaT testbed and four Wago PLCs, four Siemens IEDs in EPIC testbed. Results demonstrate that PLC identification and attack detection can be performed with high accuracy. Moreover, it is also shown that although the method may raise false positives, the rate at which they are raised is practical, in the sense that it can be managed by a human operator without creating bottlenecks, or can be fed to metamodels that take into account other features (such as model-based countermeasures, intrusion detection system (IDS) alarms, and so on).

[0075] Accordingly, various example embodiments advantageously provide a non-cryptographic risk-based technique to authenticate PLCs and detect attacks (e.g., corresponding to the method of characterising a PLC and attack detection in a networked control system as described hereinbefore according to various embodiments). There have been several conventional research works on network intrusion detection systems using network traffic features. However, various example embodiments note that anomaly or attack detection in inter-arrival time of packets alone does not work well in practice. Accordingly, various example embodiments advantageously provide a PLC fingerprinting method to fingerprint PLCs by exploiting scan cycle timing information and a PLC watermarking method to detect a powerful cyber attacker that is aware of timing profiles used for fingerprinting, for example, replay attacks.

Architecture of an ICS

[0076] A typical ICS comprises field devices (e.g., sensors and actuators), control devices (e.g., PLCs), as well as SCADA, HMI and engineering workstations. In general, an ICS follows a layered architecture. FIG. 6 depicts a schematic drawing showing an overview of an example ICS network architecture 600, according to various example embodiments of the present invention. As shown in FIG. 6, there are three levels of communications. Level 0 is the field communication network and comprises field devices, for example, remote I/O units and communication interfaces to send/receive information to/from PLCs. Level 1 is the communication layer where PLCs communicate with each other to exchange data to make control decisions (e.g., corresponding to the communication network layer based on which the plurality of PLCs communicate with each other as described hereinbefore according to various embodiments). Level 2 is the supervisory control network and is where PLCs communicate with the SCADA workstation, HMI, historian server. The communication protocols in an ICS

have been proprietary until recently when the focus shifted to using the enterprise network technologies for ease of deployment and scalability, such as the Ethernet and TCP/IP.

PLC Architecture and the Scan Cycle

5 [0077] A PLC may comprise a central unit (which may be referred to as a processor), a program and data memory unit, input/output (I/O) interfaces, communication interfaces and a power supply. I/O interface connects the PLC with input devices (e.g., sensors and switches) and output devices (e.g., actuators). The communication interfaces are used to communicate with other devices on the network (e.g., a human-machine interface (HMI)), an engineering
10 workstation, a programming device and other PLCs.

[0078] Scan Cycle Time (T_{SC}): The PLCs are part of real-time embedded systems and have to perform time-critical operations. To optimize this objective, there is the concept of control loop execution in the PLCs. A PLC has to perform its operations continuously in a loop called the scan cycle. There are three key steps in a scan cycle, namely, 1) reading the inputs, 2)
15 executing the control logic and 3) writing the outputs. A scan cycle is in the range of milliseconds (ms) with a strict upper bound referred to as the watchdog timer, else the PLC enters fault mode. The duration of the scan cycle time is based on a number of factors including the speed of the processor, the number of I/O devices, processor clock, and the complexity of the control logic. Therefore, with the variations in the hardware and control logic, these tasks
20 take variable time even for the same type of machines, resulting in device fingerprints. FIG. 7 shows an example logical flow of steps involved during a PLC scan cycle, along with an example ladder logic, according to various example embodiments of the present invention. For example, expression for the scan cycle time can be written as,

$$T_{SC} = T_{IN} + T_{CL} + T_{OP}$$

25 Equation (1)

where T_{SC} denotes the scan cycle time of a PLC, T_{IN} denotes the input read time, T_{CL} denotes the control logic execution time, and T_{OP} denotes the output write time. In this regard, various example embodiments seek to estimate the scan cycle time (T_{SC}) (or determine scan cycle related timing information) in a non-invasive manner and create a hardware and software
30 fingerprint (e.g., corresponding to the characterising information associated with the PLC as described hereinbefore according to various embodiments) based on the uniqueness of the scan cycle in each PLC. For better understanding, the relationship of the network communication to the scan cycle will now be elaborated below.

Monitoring the Scan Cycle on the Network

[0079] It is possible to capture the scan cycle information using a system call in the PLC but various example embodiments note that such an approach would not be useful to detect network layer attacks. Accordingly, various example embodiments seek to obtain the scan cycle timing information (or scan cycle related timing information) outside the PLCs and in a passive manner. To this end, various example embodiments estimate the scan cycle (or determine scan cycle related timing information) over the network and refer to it as the estimated scan cycle time (T_{ESC}) (or scan cycle related timing information). In various example embodiments, communication between PLCs is based on a request-response model. The message exchange between different PLCs may be programmed using the message instruction (MSG) on a ladder rung using the control logic as shown in FIG. 7. Since the MSG instruction is executed in the control logic which is step 2 of the scan cycle shown in FIG. 7, this MSG instruction would occur at this specific point in the scan cycle. In various example embodiments, the scan cycle time (T_{SC}) is estimated by observing the MSG requests being exchanged among PLCs on the network layer.

Scan Cycle vs. Time (IAT) of MSG Instructions

[0080] Various example embodiments note that the scan cycle and the IAT (inter-arrival time) of MSG are not equivalent. In this regard, exploiting MSG instructions in the control logic to obtain scan cycle information is challenging. For example, if the MSG instructions were executed each scan cycle, then by monitoring the IAT of MSG instructions alone would have provided the scan cycle information. In this regard, a measurement experiment was conducted, for which results are reported in Table 1 shown in FIG. 8. In particular, Table 1 show results for MSG instructions from/to PLCs and their respective timing analysis. $E[\cdot]$ denotes the expected value of a particular variable; T_{SC} denotes *ScanCycle*, T_{Resp} denotes *MSGResponseTime*, T_{ESC} denotes *EstimatedScanCycle*, *MV* denotes *motorizedvalve*, *FIT* denotes *flowmeter*, *LIT* denotes *levelsensor*; and η denotes the ratio between a scan cycle and estimated scan cycle time, lower bounded by at least 1 scan cycles. In particular, $[T_{SC}]$ represents the mean of the scan cycle time measured inside a PLC and $[T_{ESC}]$ represents the mean of the MSG instructions IAT. As can be observed from Table 1, it turns out that MSG instructions IAT instead is equal to a multiple of the scan cycle time, which may be referred to herein as estimated scan cycle time or scan cycle related timing information. In other words,

such a scan cycle related timing information is a timing information obtained that is related to (or corresponds to) the actual scan cycle time and may be referred to as an estimated scan cycle time in the sense that it is a timing information obtained when trying to determine or estimate the scan cycle time, although the actual value obtained may be far from the actual scan cycle time as shown in Table 1. Since messages are analyzed at the network layer, various example embodiments determine the relationship between the scan cycle of a PLC and what is observed at the network layer. On the network, a message would be seen at the following intervals,

$$T_{ESC} = T_{Proc} + T_{Txn} + T_{Prog} + T_{SC} + T_{Que},$$

Equation (2)

where T_{Proc} denotes the packet processing delay at a PLC, T_{Txn} and T_{Prog} denote the packet transmission and propagation delays respectively, and T_{Que} denotes the queuing delay. The relationship between T_{SC} and T_{ESC} may thus be simplified to:

$$T_{ESC} = T_{OverHead} + T_{SC},$$

Equation (3)

where $T_{OverHead} = T_{Proc} + T_{Txn} + T_{Prog} + T_{Que}$ is the time it takes for a packet to get processed at the PLC, enter a message queue for transmission and finally get transmitted on the network.

[0081] Various example embodiments note that the transmission and propagation delays are fixed per route and does not influence the variation in delays of the packets, while variable queuing delay has a significant effect on the packet delay timings and its reception on the network. Since the network configuration and the number of connected devices for an ICS network are fixed, the propagation delays can be measured and treated as constant. Accordingly, various example embodiments found that the significant effect on the estimation of the scan cycle is due to the queuing delay. In this regard, the randomness in the queuing delay depends on the network traffic directed to a particular PLC and its processor usage. Accordingly, various example embodiments seek to quantify that delay and determine if it still reveals the information about the scan cycle. It is true that the message instructions are scanned in each scan cycle but their execution depends on two conditions as specified in the following:

Condition 1: The response for the previous message request has been received. When this condition is satisfied, the Rung condition-in is set to True as shown in Figure 9. In particular, FIG. 9 depicts a schematic drawing illustrating message queuing in a PLC; and

Condition 2: The message queue has an empty slot. That is, the MSG.EW bit is set to ON as shown in FIG. 9.

[0082] Condition 1 means that the response for the last message has been received by the PLC. This process can take multiple scan cycle times. An example analysis to find the time it takes to obtain a response for a previously sent message, is shown in Table 1 in FIG. 8. This data is based on experimental setup using the SWaT testbed. The response time was calculated after a request has been received by the destination PLC and then the corresponding response has arrived at the source PLC. In Table 1, it can be seen that the response time from PLC 2 to PLC 3 is 11.062 *ms* on average. From Table 1, it can also be seen that the scan cycle time for PLC 3 measured using a system call is 4.117 *ms*. This means that it takes multiple scan cycles to obtain the response back to PLC 3. When the response has arrived the Rung condition-in in FIG. 9 becomes True. The message instruction is scanned each scan cycle but it does not get executed if Rung condition-in is not set to True. In FIG. 9, the black circle with a number at the bottom represents a scan cycle count. At the first scan cycle, when Rung condition-in becomes True, MSG.EN (message enable) bit is set to ON. Then the message is ready to enter the message queue and it checks MSG.EW (message enable wait) bit and if it is full the message keeps waiting until MSG.EW is set to ON and message can enter the queue and get transmitted on the network. As shown in FIG. 9, it can take several scan cycle times to complete the whole process. Therefore, the two above-mentioned conditions must be fulfilled to transmit the messages between PLCs. Since everything can be measured in terms of scan cycles, it would be possible to recover this scan cycle information from the network layer and use it as a hardware and software fingerprint for each PLC.

THREAT MODEL

[0083] An attacker can compromise a plant either by remotely entering the control network, physically damaging the PLCs/components of PLCs, or intercepting traffic as man-in-the-middle (MiTM). It is assumed that the adversary aims to sabotage a plant by compromising the communication between the PLCs and from PLCs to other devices such as HMI, SCADA or historian server. Once an intrusion has happened an adversary can choose to spoof the messages by using fake IDs, suspend the messages (denial of service) making the PLCs unavailable, intercepting and modifying the traffic (e.g., MiTM attack or a masquerade attack by suspending a legitimate PLC and sending fake messages on behalf of the legitimate PLC), which will ultimately falsify the current plant's state and lead potentially to unsafe states. The following example attack scenarios are considered: Denial of Service (DoS), Man-in-the-Middle (MiTM),

and Masquerading. Note that a masquerading attack can be realized by an MiTM attack that also drops the original packets produced by a given PLC.

[0084] For the masquerading attack three types of attackers are considered, namely, 1) Naive, which tries to imitate a PLC but has no knowledge about the estimated scan cycle of the PLC, 2) Powerful Partial Distribution Knowledge (PDK), which tries to imitate a PLC and knows the mean of the estimated scan cycle of a PLC, and 3) Powerful Full Distribution Knowledge (FDK), which tries to imitate a PLC and knows the full distribution of the estimated scan cycle. A customized python script was developed using pycomm library to get precise control of message transmission during the masquerading attack.

10

DESIGN CONSIDERATIONS

[0085] Various example embodiments address the following question: Is it possible to authenticate messages from each PLC in a non-intrusive and passive manner?

[0086] Various example embodiments seek to create fingerprints for the PLCs based on hardware and software characteristics of the PLCs. Once the PLCs have been identified the next step may detect a range of network attacks on the Level 1 network communication. FIG. 10 depicts an overview of a method 1000 for device identification and attack detection, according to various example embodiments of the present invention (e.g., corresponding to the method of characterising a PLC and attack detection in a networked control system as described hereinbefore according to various embodiments). The method 1000 may begin with network traffic data collection. The collected data may then be processed to estimate the scan cycle time (e.g., corresponding to determining scan cycle related timing profile information associated with the PLC based on the network traffic data obtained as described hereinbefore according to various embodiments). Next, the estimated scan cycle time may be used to extract a set of time and/or frequency domain features (e.g., corresponding to extracting one or more time and/or frequency domain features from the timing information associated with the series of request messages as described hereinbefore according to various embodiments). The extracted features may then be combined and labeled with a PLC ID. A machine learning algorithm may then be used for PLC classification and attack detection. In experiments conducted, the traffic collector was deployed at the Level 1 network (also known as SCADA control network) switch with mirror port to monitor all the network traffic at Level 1. Data was collected for all the six PLCs deployed in the SWaT testbed. The list of request messages together with the requesting and responding PLCs is provided in Table 1 in FIG. 8.

25
30

[0087] In various example embodiments, the PLCs may be profiled using time and frequency domain features of the estimated scan cycle samples (e.g., corresponding to the timing information associated with the series of request messages as described hereinbefore according to various embodiments). For example, Fast Fourier Transform may be used to convert data to the frequency domain and extract the spectral features. A list of example features along with the corresponding description is presented in Table 2 in FIG. 11. In particular, Table 2 presents a list of example features that may be used, individually or in combination, according to various example embodiments of the present invention. The information value (IV) indicated in Table 2 helps to choose features based on the values that contribute significantly to the classification accuracy and those not bringing any unique information for classification would be dropped. Data re-sampling is done to find out the sample size with which high classification accuracy can be achieved. This, in turn, would provide information about the time the method needs to make a classification decision.

[0088] The experiments were carried out in a state-of-the-art water treatment facility and a smart grid testbed. The method was tested on six different Allen Bradley PLCs, four Siemen IEDs and four Wago PLCs.

Identification of PLCs

[0089] As shown in FIG. 10, the present method compares a PLC data with a pre-created model and if the profile is matched it returns the PLC ID. During the testing phase, if the profile does not match to the pre-trained model an alarm is raised and a potential attack is declared. For better understanding, experimental results are presented in the following in the form of questions.

[0090] *Q1: Proof of fingerprint. Is the estimated scan cycle a good candidate for a fingerprint?* FIG. 12 depicts plots showing statistical features of the estimated scan cycle vector for three PLCs in the SWaT testbed, according to various example embodiments of the present invention. On the leftmost plot, the time series data of the estimated scan cycle time is plotted. It can be observed from the middle plot that the distributions of the estimated scan cycle time of the PLCs have distinctive behavior but still a few PLCs overlap. The rightmost plot shows two time-domain features namely mean and variance of the estimated scan cycle time. By using these two features, the six PLCs can be easily distinguished. This visual representation is a proof for the existence of scan cycle based fingerprint. These two features are used for the ease of visualization, however, to see the performance of the proposed technique the dataset is

analyzed in a systematic manner using machine learning and the complete feature set. The results are shown in Table 3 in FIG. 13 and discussed in the following. In particular, Table 3 shows results of the multiclass classification with respect to chunk size vs classification accuracy.

5 [0091] Q2: *PLC Identification/Attack Detection Delay. What is the right amount of data to identify PLCs with the highest accuracy?* It is observed that 120 samples are a good trade-off between accuracy and detection time with an accuracy of 96.12%. On average it takes just 3.6 seconds to make a detection decision.

10 [0092] Q3: *How will the amount of training and testing data affect PLC identification performance?* Results in Table 4 in FIG. 14 show that the accuracy of the chosen classifier function is stable for the range of data divisions and does not depend on the choice of size of the dataset. In particular, Table 4 shows results of k-fold cross validation using multi-class classifier, according to various example embodiments of the present invention. The results provide a practical insight into the case when limited data is available to train the machine learning model.

15 [0093] Q4: *Is the fingerprint stable for different runs of the experiments?* The data collected at 22 degree Celsius in scenario 1 is used to train a machine learning model and test it with the data collected in scenario 2 at 33 degree Celsius. The first row in Table 5 in FIG. 15 shows the results for this experiment. In particular, Table 5 shows results show stability from run-to-run and across temperature range, whereby data from each PLC is labeled as class-1 and all the data from rest of the five PLCs are labeled as class-2. Table 5 ensures that the fingerprint is stable for different runs and temperature variations. The use of the binary classifier also demonstrates the scalability of the method.

20 [0094] Q5: *With an increase in the number of PLCs, how accurately PLCs can be classified?* At this point a different line of argument is considered, that is, it is not necessary to compare hundreds of PLCs with each other. Since the source (expected) PLC of a message is known, the job is to verify if this message is really being generated by that particular PLC or being sent by an attacker device or being spoofed at the network layer. In Table 5, the first row shows the accuracy for PLC identification based on this binary classification. Another question is, can attacks be detected using such supervised machine learning models? To address this question, SVM was used as a one-class classifier according to various example embodiments of the present invention. In Table 5, the second row shows the results with one-class SVM (OC-SVM), to identify a particular PLC resulting in higher accuracy. Using OC-SVM makes the argument

25

30

of scalability even stronger since a model can be created by using just the normal data of a PLC. In the following section, the performance of attack detection using one-class classification model is discussed.

5 *Practical application of the present method in an ICS*

[0095] One advantageous feature of the method is that it is the combination of PLC hardware and control logic execution time. Therefore, it is possible to create a unique fingerprint even for similar PLCs with the similar control logic which is probable in an industrial control system. In the Table 3, it can be seen that the multiclass accuracy to uniquely identify all the six PLCs in SWaT testbed is 93.54% for a sample size of 100. Upon investigation, it was observed that PLC 3 and PLC 4 have a very similar profile hence lower classification accuracy. Considering that the fingerprint is the combination of the hardware and the control logic, an experiment to force a distinguishing fingerprint is proposed. To remove the collision between the PLC 3 and PLC 4, an extra delay was added to the ladder logic of the PLC 4 without affecting the normal operation. The classification accuracy for multiclass classification of six PLCs is increased to 99.06%. The results are summarized in Table 6 in FIG. 16. In particular, Table 6 shows the accuracy improvement due to change in the control logic of PLC 4, whereby DF denotes Default Profiles and MF denotes Modified Profiles.

20 *Generalising to Other ICSs and Devices*

[0096] The method has been tested in another physical process (electric power grid testbed), EPIC, employing different type of devices (WAGO PLCs and Siemens IEDs (Intelligent Electronic Devices)). The EPIC was divided into four main sectors: Generation, Transmission, Micro-Grid and Smart Home. Each sector comprises various electrical equipment such as motors, generators and load banks. These equipment can be monitored and managed by different digital control components such as PLCs, Intelligent Electronic Device (IED) through different communication medium. Generic Object Oriented Substation Event (GOOSE), MODBUS Serial, TCP/IP and Manufacturing Messaging Specification (MMS) were employed from the IEC 61850 standard communication networks and systems in substations. Siemens Protection Relay Intelligent Electronic Device (IED) which provides Protection, Instrumentation and Metering functionality. Four Siemens 7SR242 series IEDs were used. Accordingly, both the devices being used are diversified, as well as the communication protocols.

[0097] Results are shown in Table 7 in FIG. 17. In particular, Table 7 shows the results for a multi-class classification in relation to the EPIC Testbed Performance Evaluation, according to various example embodiments of the present invention. Timing profiles obtained for all four devices compared against the signature of each device and the accurate identification percentage is reported in Table 7. For the same four PLCs with identical control logic a 70% device identification result is encouraging. Moreover, it is to be noted from the above section (*Practical application of the present method in an ICS*) that by modifying the control logic in a benign manner it is possible to achieve accuracy close to 100%.

10 *Attack Detection*

[0098] A powerful masquerader with the knowledge of the network traffic pattern can try to maintain the normal network traffic statistics. Such a masquerade attacker would deceive the conventional network traffic based intrusion detection methods. The present method is based on the hardware and software characteristics of the devices, which are hard for an attacker to replicate.

[0099] *Q6: How well does the proposed technique perform to detect powerful network attacks on PLCs?* The intuition behind the attack detection is that an attack on the network even if not affecting the network traffic statistics, must cause deviation to the estimated scan cycle fingerprint profile of the associated PLC. In Table 8 in FIG. 18, the attack detection rate (TPR) and attack missing rate (FNR) are shown. In particular, Table 8 shows the attack detection performance, whereby MN denotes Masquerader Naïve; MPDK denotes Masquerader Partial Distribution Knowledge; and MFDK denotes Masquerader Full Distribution Knowledge.

[00100] The first row shows the attack detection performance for a MiTM attacker which intercepts the traffic between the PLCs and then forwards the compromised messages. It is observed that all the attacks on all PLCs are detected with 100% accuracy. To understand these attack scenarios consider FIG. 19 which shows plots for the estimated scan cycle for PLC 1 under attack and normal operation, according to various example embodiments of the present invention. For the masquerading attack, three types of attackers were considered, namely, 1) Naive, which tries to imitate a PLC but has no knowledge about the estimated scan cycle of the PLC; 2) Powerful Partial Distribution Knowledge (PDK), which tries to imitate a PLC and knows the mean of the estimated scan cycle of a PLC, and 3) Powerful Full Distribution Knowledge (FDK), which tries to imitate a PLC and knows the full distribution of the estimated scan cycle.

[00101] A powerful masquerader tries to imitate a PLC by sending fake messages at the exact time using its knowledge. Now this powerful attacker could not be detected by the conventional network traffic pattern based methods because the number of packets, packet length, header information and other network profiles would all be the same as normal operation.

5 In contrast, the present method is able to detect this attack because the attacker deviates from the fingerprinted profile. For example, in the rightmost plot in FIG. 19, it can be seen that the profile under this masquerading attack deviates massively from the normal fingerprint profile although the number of packets and other network configurations are not that different. This result is reflected in Table 8 in the third row where except one case all the attacks are detected
10 with 100% accuracy. Amid the high accuracy, one can make the attacker even more powerful by providing it with the complete distribution of the estimated scan cycle vector. Row 4 in Table 8 contains results for such an attacker. For PLC 1 and PLC 3, the attacker was able to imitate the PLCs perfectly thus avoiding detection. Even though attacker has the full distribution knowledge but still due to attacker's hardware imperfections for some scenarios
15 high detection rates were obtained. To reinforce this result, FIGs. 20A to 20C show all the PLCs under this powerful masquerade attack. It can be observed from the top row how similar attacked data time series is to the normal data. This result is very significant in the sense that the attacker does not change the network statistics and sends the fake messages pretending to be one of the legitimate PLCs. It is the unique characteristic (scan cycle, queuing load)
20 fingerprint of the PLC which allows attack detection.

[00102] The performance evaluation of the classifier will now be discussed, according to various example embodiments of the present invention. A one-class SVM was used to detect attacks. To visualize the performance of the classifier a Receiver Operating Curve (ROC) is plotted using TPR (attacks rightfully detected) and FPR (normal data detected as an attack).
25 SVM model measures the confidence that the test data belongs to the data on which model was trained. ROC plot for the last two attack scenarios from Table 8 are shown in FIG. 21 and FIG. 22, respectively. In particular, FIG. 21 shows a plot of ROC for Masquerader Partial Distribution Knowledge, and FIG. 22 shows a plot of ROC for MFDK Masquerader Full Distribution Knowledge. From FIG. 21, it can be seen that for PLC 1 and PLC 3 if we are
30 willing to accept a very high FPR then the attack detection could also be made. For PLC 2 the detection rate was not 100% and hence as seen in FIG. 22 an increase in FPR can result in high TPR. The key takeaway from this analysis of the classifier is that the classifier according to various example embodiments of the present invention was not trained for very high FPR to

demonstrate the 100% attack detection rate. High detection rate is the result of underlying present detection technique based on the estimated scan cycle time (e.g., corresponding to the method 300 of attack detection as described hereinbefore according to various embodiments).

[00103] Accordingly, it can be observed from FIGs. 20A to 20C that the attacker was able to imitate the message transmission behavior of a PLC but still got exposed in most cases due to the limitation in the attacker's own hardware. Nevertheless, various example embodiments do not assume any limitations on attacker's part and consider that an attacker is capable of perfectly imitating the PLCs and is also able to do a replay attack. In the following, a PLC watermarking method for detecting intrusions from such powerful attackers will now be described according to various example embodiments of the present invention.

PLC WATERMARKING

[00104] PLC Watermarking exploits the relationship of PLC's unique feature of Scan Cycle Time and the network layer data request messages exchanged among the PLCs. In various example embodiments, PLC watermarking seeks to extend a static fingerprint as discussed hereinbefore to a dynamic, randomly generated scheme to tackle a powerful attacker. In various example embodiments, randomness in the watermark is generated by, 1) using the clock of the PLC to inject random delay and/or 2) injecting the watermark signal for a random number of scan cycle count (i.e., number of scan cycles for which a particular watermark is to be added). For example, such a watermark can be added through a system call sampling PLC clock via a single ladder logic instruction. An example is shown in FIG. 23 where a constant watermark signal is injected labeled as the watermark. In particular, FIG. 23 illustrates an example watermark for PLC 1 in SWaT. The y-axis depicts the change in T_{ESC} because of the watermark and x-axis the change in the number of packets due to the watermark.

[00105] The request-response model for data exchange among the PLCs facilitated the design of PLC watermarking. Various example embodiments observe that, 1) the request messages are controlled by Scan Cycle Time and 2) the time of arrival of response messages is a function of request message arrival time. These two observations led various example embodiments to, 1) affect the message control by manipulating the scan cycle time and 2) exploit the feedback loop for request-response channel to inject and observe a watermark signal, respectively. FIG. 24 presents the first hypothesis related to the scan cycle time. In particular, FIG. 24 depicts an example design or configuration of the PLC watermarking method according to various example embodiments of the present invention. A watermark is injected at the end

of the control logic (i.e., end of scan cycle) to introduce delays to the transmitted data request messages on the network. FIG. 25 shows the result for the second hypothesis depicting that the timing profile for the response messages has the similar distribution as the request messages. In particular, FIG. 25 depicts an example request-response messaging in the PLCs, according to various example embodiments of the present invention.

[00106] In FIG. 24, an example of communication between two PLCs is shown. For simplicity, PLC 1 is assumed to transmit messages to PLC 2, representing explicit message exchange between PLCs. Under normal operation, PLC 1 sends a request to PLC 2 labeled as Req_1 and after some time gets the response labeled as $resp_1$. The second request is labeled as Req_2 . The time between these two requests, and similar subsequent requests, establish a profile for estimated scan cycle time as shown in previous sections. A random delay, labeled $T_{Watermark}$, can be added making request 2 at a later time labeled as $\overline{Req}_2$. The time difference between Req_1 and $\overline{Req}_2$, and the subsequent packets using $T_{Watermark}$, constitutes a profile for PLC Watermarking. The plot on the right-hand side in Figure 11 depicts the distributions for the case of the normal operation of the PLCs and with a watermark. An example of such a watermark is shown in 23 for the case of PLC1 in SWaT testbed. According to various example embodiments, it is important to respect the real-time constraint also known as watchdog timer: $[max T_{SC}]_{Watermark} < T_{watchdog}$. In various example embodiments, watermark upper bounds are empirically determined by scan cycle measurements for each PLC.

Distinguishing Watermarked Signal from Normal

[00107] In various example embodiments, the goal is to investigate how the watermark could be distinguished from the normal profile, and how to build such a watermark. The result in FIG. 25 shows a Gaussian approximation for estimated scan cycle time T_{ESC} . Gaussian distribution possesses useful properties, e.g., scaling and shifting of a Gaussian random variable preserves its distribution.

[00108] *Proposition 5.1.* Linear transformation of a random variable. For a random variable $\mathbf{X}$ with mean μ and standard deviation σ as defined by a Gaussian distribution, then for any $a, b \in \mathbb{R}$ distribution, then for any $a, b \in \mathbb{R}$

$$\mathbf{Y} = a\mathbf{X} + b$$

(Equation 4)

then, $\mathbf{Y}$ is a Gaussian random variable with mean $a\mu + b$ and standard deviation $a^2\sigma$.

[00109] *Hypothesis Testing.* The estimated scan cycle T_{ESC} vector under normal operation can be represented as r_k for the k^{th} PLC and for the watermark as $r_k^* = \beta(r_k) + \alpha$, where α is the random delay introduced as a watermark and β denotes the scaling of the distribution due to the change in the control logic. Intuitively it means that the estimated scan cycle pattern in r_k is offset with a constant value α , an obvious consequence of which is change in the mean of the random variable. Using the above-mentioned proposition 5.1, it can be seen that the resultant vector ($r_k^* = \beta(r_k) + \alpha$) is a linearly transformed version of r_k . The mean for such a change is $\bar{r}_k^* = \alpha + (\beta)\bar{r}_k$ and variance $S_{r^*}^2 = (\beta)^2 S_r^2$. For this watermark response protocol, two hypotheses need to be tested, $\mathcal{H}_0$ the without watermark mode and $\mathcal{H}_1$ the with watermark mode using a K-S test.

Kolmogorov-Smirnov (K-S) Test

[00110] In various example embodiments, a two-sample K-S test was used. The K-S statistics quantifies a distance between the empirical distribution functions of two samples. Under a replay attack samples would look like the original trained model without watermark. In the absence of any attack watermark would be preserved and null hypothesis would be rejected.

[00111] *K-S test based model training.* In the following, the empirical distributions for the estimated scan cycle time T_{ESC} will be derived and a reference model is obtained without watermarking. There are thousands of samples captured from the PLCs in a matter of a few minutes. If all the captured samples from an experiment are considered it results in a smooth empirical distribution but then the time to make a decision also increases by a few minutes. Therefore, a trade-off between the speed of detection and detection performance is desired. Results are depicted in a tabular form in Table 9 in FIG. 26 for PLC 1. In particular, Table 9 show the K-S test results for normal data in relation to chunk size vs classification accuracy for PLC 1, according to various example embodiments of the present invention. For a chunk size of 60, a TNR of 100% was achieved but, to be little more conservative, a value of 120 chunk size is chosen in the following analysis.

[00112] Two example use cases are shown in FIG. 27. In particular, FIG. 27 depicts Gaussian distribution approximation of PLC1 under normal operation for different runs of the experiment, whereby $D_{m,n}$ is the maximum distance between two distributions. For the graph on the left, it can be seen that for two different samples of Scan Cycle Time for PLC1 under normal operation are very similar, that is, both the samples are drawn from the same distribution. On the right,

one sample is taken from normal operation and the second sample from the watermarked Scan Cycle Time of PLC1. It is observed that the distance metric Dm , is greater as compared to the plot on the left hand side, thus these two samples are drawn from different distributions. This is the key intuition to detect replay attacks in the presence of a watermark signal.

- 5 **[00113]** *K-S test based model testing.* In various example embodiments, testing was done for the dataset from normal operation and for PLC Watermarking. It can be seen from Table 10 shown in FIG. 28 that the K-S test produces 100% true negative rate that is classifying normal data as normal. In particular, Table 10 shows K-S test results with and without adding the watermark, with accuracy for PLC 1 for a chunk size of 120 samples shown. Second and third
10 columns present results for the case of a static watermark signal. The second column shows the result of injecting a watermark delay of 20 ms, while the third column shows the results obtained by injecting a watermark delay of 40 ms. In both cases, with high accuracy, the watermark signals could be classified. The third case is that of a random watermark created using the clock of the PLC. Such a high accuracy of detection motivates the use of K-S test for attack detection.

15

Watermark Modeling: A Closed Loop Feedback System

- [00114]** It can be observed that there is a strong relationship between PLC request and response messages as depicted in FIGs. 24 and 25. This process of message exchange can be modeled as a closed loop feedback system from the perspective of control theory. An example
20 closed loop feedback system model is depicted in FIG. 29, according to various example embodiments of the present invention.

[00115] *Definition 5.1.* The Inter Arrival Time (IAT) of MSG requests and responses, respectively, is defined as the system state referred to as x_k , where k is the message number.

- [00116]** *Definition 5.2.* The response MSG time is treated as the output of a system, such as
25 an output of a sensor, defined as y_k .

[00117] *Definition 5.3.* The dynamics of the scan cycle, PLC hardware and logic complexity govern the dynamics of request messages being sent to other PLCs. These dynamics which are reflected in estimated scan cycle control the timings of receiving the response message. This control action is denoted as u_k , where k is the message number.

- 30 **[00118]** Using subspace system identification methods, the process dynamics can be modeled and represented in a state space form as follows,

$$x_{k+1} = Ax_k + Bu_k + v_k, \text{ and}$$

(Equation 5)

$$y_k = Cx_k + \eta_k.$$

(Equation 6)

[00119] This is a state space system model which can be generally used to model dynamics of a physical process. In the system of Equations (5) and (6), y_k is the output of the system which is response message timing profile. This output is a function of request messages that act as a control input. FIG. 25 shows the relationship of response message with the request message timing profile which is driven by the scan cycle time. v_k and η_k are identical and independently distributed sources of noise due to communication channels. Matrices A , B and C capture the input-output relationship and models the communication among PLCs. The system of Equations in (5) and (6) can model the underlying system but it can be subject to powerful attacks for example replay attacks or masquerade attacks. An attacker can learn the system behavior and replay the collected data while real system state might be reporting different measurements.

[00120] *Definition 5.4.* PLC Watermarking Δu_k : The output (i.e., the response MSG time) depends on the request MSG profile. However, request MSG timing profile depends on the scan cycle and other communication overheads. These factors together constitute the control input u_k . A watermark is added to this control input such that the effects of the added watermark are observable on the output of the system, i.e., the response MSG timing profile y_k .

[00121] *Proposition 5.2.* Replay attack can be detected using the PLC Watermarking technique given the system model of Equations (5) and (6).

[00122] *Proof:* The PLC watermarking technique injects a watermark defined as Δu_k in the control signal u_k . A replay attack will use the normal data and system model as defined in Equations (5) and (6). An attacker, unaware of the watermark, would be exposed as follows,

$$x_{k+1} = Ax_k + B(u_k + \Delta u_k) + v_k,$$

(Equation 7)

$$y_{k+1} = Cx_{k+1} + \eta_{k+1}.$$

(Equation 8)

[00123] Substituting x_{k+1} in the above equation results in,

$$y_{k+1} = CAx_k + CBu_k + v_{k+1} + \eta_{k+1} + CB\Delta u_k.$$

(Equation 9)

[00124] The last term in the above equation is the watermark signal which is generated randomly using the PLC clock. This watermark signal will expose a replay attack.

[00125] The effects of the watermark in detecting attacks are considered next. FIG. 30 shows an example of powerful masquerade attack. In particular, FIG. 30 depicts a plot of empirical cumulative distributions for a masquerade attack on PLC1 and watermark, according to various example embodiments of the present invention. These cumulative distributions show that due to the watermark a masquerader exposes itself via a K-S test. In this case, the defender is expecting the presence of a watermark in the response received from the other PLC but it did not get that and raised an alarm. The same result is shown in Table 11 shown in FIG. 31. In particular, Table 11 shows the results for the K-S test for attack data vs watermark attack detection accuracy for all PLCs in the SWaT testbed for a chunk size of 120 samples, whereby MFDDK denotes Masquerade Full Distribution Knowledge. The results are for all PLCs in the six stages of SWaT. Replay and powerful masquerade attacks are detected with high accuracy using the watermark signals as shown in FIG. 30. A high true positive rate (i.e., attacks declared as attacks), as compared to the results in Table 8, points to the effectiveness of PLC Watermarking as an attack detection technique. In the following, the threat model is further strengthened by assuming that the attacker has the knowledge of the system model and attempts to estimate the watermark signal. FIG. 32 depicts the response time for a message instruction in PLC3 in SWaT testbed, according to various example embodiments of the present invention.

System State Estimation

[00126] While modeling the system in a state space form, the system states can be estimated using Kalman filter. This formulation helps in detecting MiTM attackers which adds same delay in the request (input) and the response (output) messages and is also useful in quantifying the contribution of the watermark signal in the response message by normalizing the input and output in terms of the residual.

[00127] *Definition 5.5.* Let the response MSG timing measurements under a replay attack be y_k^a , control (request MSG) signal under replay attack u_k^a , and the state estimate x_k^a at time step k , $0 < k \leq T$ for an attack time period T .

[00128] *Proposition 5.3.* Given the system of equations for normal system model as defined in Equations (21)-(22) and replay attack defined in Equations (23)-(24) to be described later below, it can be shown that replay attack would not be detected.

[00129] *Proof:* The residual vector under an attack is given as,

$$r_{k+1}^a = y_{k+1}^a - \hat{y}_{k+1}.$$

(Equation 10)

[00130] During the replay attack for times $0 < k \leq T$, where T is the time for the readings being replayed, $y_{k+1}^a = y_{k+1}$, resulting in $r_{k+1}^a = r_{k+1}$. Therefore, this results in no detection and the alarm rate reduces to the false alarm rate of the detector in use.

[00131] *Theorem 5.1.* Given the system model in Equations (9) and (6), Kalman filter in Equations (15) and (13) to be described later below, and watermarked signal Δu_k , it can be shown that the residual vector is driven by the watermark signal and can be given as, $r_{k+1} = [CA - CLC](x_k^a - \hat{x}_k^{wm}) + CB(u_k^a - u_k) - CB\Delta u_k + Cv_k + \eta_{k+1}$.

[00132] The term $-CB(\Delta u_k)$ quantifies the effects of watermark. Based on this it could be investigated if it is possible to recover the watermark signal from the response message, and if not then the system is declared as under attack. FIG. 33 shows results from an experiment with a watermarked delay of 40ms. In particular, FIG. 33 shows a delay of 40ms is used as a watermark for PLC1. In the top pane it can be seen that the watermarked response is different from the estimate of response MSG IAT using the system model developed earlier. In the bottom, a density plot for the residual vector is shown which can be seen as being deviated from the zero mean under the effect of the watermark. The amount this distribution deviate depends on the watermark.

A More Powerful Attacker

[00133] In the following analysis we consider the case that attacker is trying to estimate the watermark signal as well.

[00134] *Definition 5.6.* A watermark signal which is chosen randomly in each iteration of experiment is defined as the dynamic watermark.

[00135] *Theorem 5.2.* An attacker attempting to estimate the watermark signal in the presence of dynamic watermark signal, would be detected because of the hardware delays to switch between the watermarks. System model results in $r_{k+1} = (CA - CLC)(x_k^a - \hat{x}_k^{wm}) + CB(u_k^a - u_k) + CB(\Delta u_k^a - u_k) + Cv_k + \eta_{k+1}$, where Δu_k^a is attacker's learned watermark signal. Proof is similar to the proof of the above-mentioned theorem 5.1.

[00136] In the result from the theorem 5.2 the first three terms are important. The attacker's goal is to make $x_k^a = x_k^{wm}$, $u_k^a = u_k$ and $\Delta u_k^a = \Delta u_k$. If this can be achieved, the attacker can hide amid the use of the watermark signal Δu_k . The attacker needs finite time to obtain the estimate (Δu_k^a) of the change in the dynamic watermark signal. While doing so, the attacker will be exposed due to the zero transition time required to detect and switch to the new random watermark signal.

[00137] *Security Argument.* The present watermark is an instance of the control logic shown to alter the execution time and ultimately the scan cycle time. Therefore, by challenging a PLC at a randomly selected time t and duration δ , one would expect the estimated scan cycle measurements to contain the effects of the watermark. If not, then one would suspect that the measurements received, at the network, to be spoofed. An attacker is aware of such a watermark, but at the same time is spoofing the messages at the network layer, therefore needs to consistently reflect the watermark profile starting at time t and for duration δ . However, the attacker needs to wait Δ seconds to recognize that the estimated scan cycle profile has been changed at the beginning and at the end of the watermark. Therefore, the attacker can at most react consistently with the expectations of the watermark at time $(t + \Delta)$ and stop at $(t + \delta + \Delta)$. As is shown in the previous sections, Δ time units needed to detect if a change in the profile is significant, is approximately 3 seconds in our case study, and can be leveraged to detect incoherent responses of the attacker to the watermark.

[00138] A dynamic random watermark signal is shown in FIG. 34. In particular, FIG. 34 depicts a masquerade attack and watermark time series, according to various example embodiments of the present invention. The top pane shows the response MSG timing profile for a random watermark signal. In the bottom, a masquerader's strategy is shown for PLC1. From the top pane it can be seen that the watermark is random while it does not affect the system performance as it is a tightly bounded function. A masquerade attacker knows the original system model and inputs/outputs but cannot completely follow the dynamic watermark, thus being exposed.

[00139] Accordingly, a timing based fingerprinting method is provided according to various example embodiments for industrial PLCs. The timing based fingerprinting method was used to estimate the scan cycle time (or determine the scan cycle related timing information). Based on the timing based fingerprinting method, it is observed that PLCs could be uniquely identified without any modifications to the control logic. It is possible to create unique fingerprints for the same model of PLCs. Furthermore, powerful attackers with the knowledge of scan cycle time and replay attacks are able to be detected by the PLC watermarking technique according to various example embodiments.

SYSTEM MODELING

Kalman filter

[00140] Given the system model in Equations (5) and (6), the state of a system can be estimated based on the available output y_k , using a linear Kalman filter with the following structure,

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + L_k(\bar{y}_k - C\hat{x}_k), \quad (\text{Equation 11})$$

with estimated state $\hat{x}_k \in \mathbb{R}^n$, $\hat{x}_1 = E[x(t_1)]$, where $E[\cdot]$ denotes the expectation, and gain matrix $L_k \in \mathbb{R}^{n \times m}$. The estimation error is defined as $e_k := x_k - \hat{x}_k$. In Kalman filter matrix L_k is designed to minimize the covariance matrix $P_k = E[e_k e_k^T]$ in the absence of attacks.

[00141] Equation (11) is an overview of the system model where the Kalman filter is being used for estimation. The estimator makes an estimate at each time step based on the previous readings up to x_{k+1} and the sensor reading y_k . the estimator gives $\hat{x}_k$ as an estimate of state variable x_k . Thus, an error can be defined as,

$$e_k = \hat{x}_k - x_k, \quad (\text{Equation 12})$$

where $\hat{x}_{k|j}$ denotes the optimal estimate for x_k given the measurements $y_1, \dots, y_j$. Let P_k denote the error covariance, $Cov(e_k) = E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]$, and $\hat{P}_{k|j}$ the estimate of P_k given $y_1, \dots, y_j$.

[00142] Prediction equation for state variable using Kalman filter can be written as,

$$\hat{x}_{k+1|k} = A\hat{x}_{k|k} \quad (\text{Equation 13})$$

$$P_{k+1|k} = AP_{k|k}A^T + Q, \quad (\text{Equation 14})$$

where $\hat{x}_{k|k}$ is the estimate at time step k using measurements up to time k and $\hat{x}_{k+1|k}$ is $(k+1)^{th}$ the prediction based on previous k measurements. Similarly, $P_{k|k}$ is the error covariance estimate until time step k . Q is the process noise covariance matrix. The next step in Kalman filter estimation is time update step using Kalman gain L_k .

$$L_k = P_{k|k-1}C^T(CP_{k|k}C^T + R)^{-1} \quad (\text{Equation 15})$$

$$\bar{x}_{k+1|k} = \hat{x}_{k+1|k} + L_k(y_k - C\hat{x}_{k+1|k})^{-1}$$

(Equation 16)

$$\bar{P}_{k+1|k} = (I - L_k C) \hat{P}_{k+1|k},$$

(Equation 17)

- 5 where $\bar{x}_{k+1|k}$ and $\bar{P}_{k+1|k}$, are the updates for the $k + 1$ time step using measurements y_i from the i^{th} sensor and Kalman gain L_k . R is the measurement noise covariance matrix. The initial state can be selected as $x_0 = x_0$ with $P_0 = E[(\hat{x}_0 - x_0)(\hat{x}_0 - x_0)^T]$. Kalman gain L_k is updated at each time step but after a few iterations it converges and operates in a steady state. Kalman filter is an iterative estimator and $\hat{x}_{k|k}$ in equation 13 comes from $\bar{x}_{k-1|k}$ in equation 16. It is assumed that the system is in a steady state before attacks are launched. Kalman filter gain is represented by L in steady state.

Residuals and hypothesis testing

- [00143] The estimated system state is compared with timing measurements $\bar{y}_k$ which may have the presence of an attacker. The difference between the two should stay within a certain threshold under normal operation, otherwise, an alarm is triggered. The residual random sequence r_k , $k \in \mathbb{N}$ is defined as,

$$r_k := y_k - C \hat{x}_k = C e_k + \eta_k + \delta_k.$$

(Equation 18)

- 20 [00144] If there are no attacks, the mean of the residual is

$$E[r_{k+1}] = CE[e_{k+1}] + E[\eta_{k+1}] = \bar{r}_{m \times 1} = 0_{m \times 1}.$$

(Equation 19)

where $\bar{r}_{m \times 1}$ denotes $m \times 1$ matrix composed of mean of residuals under normal operation, and the co-variance is given by

$$25 \quad \Sigma := E[r_{k+1} r_{k+1}^T] = CPC^T + R_2.$$

(Equation 20)

- [00145] For this residual, hypothesis testing is done, $\mathcal{H}_0$ the normal mode, i.e., no attacks, and $\mathcal{H}_1$ the faulty mode, i.e., with attacks. The residuals are obtained using this data together with the state estimates. Thus, the two hypotheses can be stated as follows,

$$30 \quad \mathcal{H}_0 : \begin{cases} E[r_k] = \bar{r}_{m \times 1}, \\ E[r_k r_k^T] = \Sigma, \end{cases} \quad \text{or} \quad \mathcal{H}_1 : \begin{cases} E[r_k] \neq \bar{r}_{m \times 1}, \\ E[r_k r_k^T] \neq \Sigma, \end{cases}$$

- [00146] The system model under the normal operation and in the presence of a replay attack can be given as,

[00147] Normal Operation:

$$\begin{cases} X_{k+1} = Ax_k + Bu_k + v_k, \\ y_k = Cx_k + \eta_k, \end{cases} \quad \text{system model.} \quad (\text{Equation 21})$$

$$\begin{cases} \hat{X}_{k+1} = A\hat{x}_k + Bu_k + L_k(y_k - C\hat{x}_k), \\ \hat{y}_k = C\hat{x}_k, \end{cases} \quad \text{state estimation.} \quad (\text{Equation 22})$$

5

[00148] Replay Attack:

$$\begin{cases} X_{k+1}^a = Ax_k^a + Bu_k^a + v_k, \\ y_k^a = Cx_k^a + \eta_k, \end{cases} \quad \text{system model.} \quad (\text{Equation 23})$$

$$\begin{cases} \hat{x}_{k+1}^a = A\hat{x}_k^a + Bu_k^a + L_k(y_k^a - C\hat{x}_k^a), \\ \hat{y}_k^a = C\hat{x}_k^a, \end{cases} \quad \text{state estimation.} \quad (\text{Equation 24})$$

10

THEOREM PROOF

Theorem

15 [00149] *Theorem 5.1:* Given the system model in Equations (9) and (6), Kalman filter (15) and (13), and watermarked inputs Δu_k , it can be shown that the residual vector is driven by the watermark signal and can be given as, $r_{k+1} = [CA - CLC](x_k^a - \hat{x}_k^w m) + CB(u_k^a - u_k) - CB\Delta u_k + Cv_k + \eta_{k+1}$.

[00150] *Proof:* In the system model of Equation (5), attacker has access to the normal timing measurements and control (request) signals and can replay those. Assuming that an adversary has access to the system model, Kalman filter gain and the parameters of the detector. During the replay attack using its knowledge an attacker tries to replay the data resulting in a system state described as,

$$x_{k+1}^a = Ax_k^a + Bu_k^a + v_k \quad (\text{Equation 25})$$

25

and attacker's spoofed sensor measurements as,

$$y_k^a = Cx_k^a + \eta_k \quad (\text{Equation 26})$$

$$y_{k+1}^a = C[Ax_k^a + Bu_k^a + v_k] + \eta_{k+1}$$

30

(Equation 27)

$$y_{k+1}^a = CAx_k^a + CBu_k^a + Cv_k + \eta_{k+1} \quad (\text{Equation 28})$$

[00151] However, using a watermarking signal in the control input u_k , the defender's state estimate becomes,

$$\hat{x}_{k+1}^{wm} = A\hat{x}_k^{wm} + Bu_k + B\Delta u_k + L(y_k^a - C\hat{x}_k^{wm}), \quad (\text{Equation 29})$$

where Δu_k is the watermark signal.

$$\hat{y}_{k+1}^{wm} = C\hat{x}_{k+1}^{wm}, \quad (\text{Equation 30})$$

$$\hat{y}_{k+1}^{wm} = C[A\hat{x}_k^{wm} + Bu_k + B\Delta u_k + L(y_k^a - C\hat{x}_k^{wm})], \quad (\text{Equation 31})$$

$$\hat{y}_{k+1}^{wm} = CA\hat{x}_k^{wm} + CBu_k + B\Delta u_k + CL(Cx_k^a - C\hat{x}_k^{wm}), \quad (\text{Equation 32})$$

$$\hat{y}_{k+1}^{wm} = CA\hat{x}_k^{wm} + CBu_k + B\Delta u_k + CLCx_k^a - CLC\hat{x}_k^{wm}, \quad (\text{Equation 33})$$

[00152] The residual vector in the presence of a replay attack is given as,

$$r_{k+1} = y_{k+1}^a - \hat{y}_{k+1}^{wm} \quad (\text{Equation 34})$$

$$r_{k+1} = CAx_k^a + CBu_k^a + Cv_k + \eta_{k+1} - CA\hat{x}_k^{wm} - CBu_k - CB\Delta u_k - CLCx_k^a + CLC\hat{x}_k^{wm}, \quad (\text{Equation 35})$$

$$r_{k+1} = (CA - CLC)x_k^a - (CA - CLC)\hat{x}_k^{wm} + CB(u_k^a - u_k) - CB(\Delta u_k) + Cv_k + \eta_{k+1}, \quad (\text{Equation 36})$$

$$r_{k+1} = (CA - CLC)(x_k^a - \hat{x}_k^{wm}) + CB(u_k^a - u_k) + Cv_k + \eta_{k+1} - CB(\Delta u_k), \quad (\text{Equation 37})$$

[00153] The last term above is the watermark signal and the first term is the error. For a stable system the spectral radius of $(CA - CLC < 1)$ and the error converges to zero.

PERFORMANCE METRICS

[00154] Each PLC is assigned a unique ID and multi-class classification is applied to identify it among all the PLCs. Identification accuracy is used as a performance metric. Let c denote the total number of classes, TP_i the true positive for class c_i when it is rightly classified, FN_i the false negative defined as the incorrectly rejected, FP_i the false positive as incorrectly accepted,

and FP_i the true negative as the number of correctly. The overall accuracy (acc) is defined as follows.

$$acc = \frac{\sum_{i=1}^c TP_i + \sum_{i=1}^c TN_i}{\sum_{i=1}^c TP_i + \sum_{i=1}^c TN_i + \sum_{i=1}^c FP_i + \sum_{i=1}^c FN_i}$$

(Equation 38)

- 5 **[00155]** While embodiments of the invention have been particularly shown and described with reference to specific embodiments, it should be understood by those skilled in the art that various changes in form and detail may be made therein without departing from the scope of the invention as defined by the appended claims. The scope of the invention is thus indicated by the appended claims and all changes which come within the meaning and range of
- 10 equivalency of the claims are therefore intended to be embraced.