基于依赖路径分析的 PLC 变量关联关系分析方法

姚文君[1),2),3)] 方滨兴[1),2),3)] 吴国栋[1),2),3)] 孙彦斌[1),2),3)] 田志宏[1),2),3)]

[1)](广州大学网络空间安全学院 广州 510000)

[2)](广东省工业控制系统攻防对抗重点实验室 广州 510000)

[3)](广州大学黄埔研究院 广州 510000)

摘 要 可编程逻辑控制器(PLC)的变量负责传感器数据接收、执行器数据发送及中间变量存储，其关联关系易被攻击者利用。PLC 蜜点作为一种安全防御工具，通过模拟真实工业控制系统(ICS)环境吸引和分析攻击行为，可以融入变量关联关系以提升仿真真实性。为此，挖掘 PLC 变量间的关联关系对 ICS 攻防至关重要。传统静态代码分析忽视了代码运行时的动态特征，而数据采集与监视控制系统(SCADA)日志因果分析仅依赖统计特性，难以全面反映 PLC 变量真实的关联关系。为克服这些局限，本文提出一种基于依赖路径分析的 PLC 变量关联关系分析方法（A PLC variable correlation analysis method based on dependency path analysis, VCA）。VCA 融合静态与动态特征，首先解析 PLC 代码为抽象语法树并构建程序依赖图，再利用 SCADA 日志时序数据，通过互信息为依赖边赋权重，最后构建变量权重的加权邻接矩阵，计算路径总权重以量化关联关系。实验表明，VCA 组装线工控系统上准确识别了 16 组强关联关系的 PLC 变量对，相较于现有研究具有明显优势。VCA 还为设计 PLC 蜜点时提供 ICS 功能域知识，模拟变量依赖逻辑，提升仿真真实性与诱捕效果。

关键词 工业控制系统；PLC 变量；关联分析；程序依赖图；PLC 蜜点

中图法分类号 TP309

A PLC variable correlation analysis method based on dependency path analysis

YAO Wen-Jun[1),2),3)] FANG Bin-Xing[1),2),3)] WU Guo-Dong[1),2),3)] SUN Yan-Bin[1),2),3)] TIAN Zhi-Hong[1),2),3)]

[1)](Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, 510000)

[2)](Guangdong Key Laboratory of Industrial Control System Security, Guangzhou, 510000)

[3)](Huangpu Research School of Guangzhou University, Guangzhou, 510000)

Abstract Programmable Logic Controllers (PLCs) play a critical role in Industrial Control Systems (ICS), overseeing essential operations such as acquiring data from sensors, issuing commands to actuators, and managing intermediate variable storage. The industrial control process is driven by changes in the values of PLC variables, as these PLC variables are intricately linked through correlations that govern the system's functionality. However, these relationships also represent potential weaknesses, as attackers can exploit them to compromise industrial processes. PLC honeypoints have emerged as an innovative security mechanism. By emulating genuine ICS environments, these honeypoints lure attackers, enabling the observation and analysis of malicious

本课题得到国家自然科学基金项目（No.U2468204, U2436208, 62372129, 62272119），广东省重点研发项目（2024B0101010002），广东省工业控制系统攻防对抗重点实验室项目（2024B1212020010），山东省重点研发计划项目（2025CXGC010901），广州市基础和应用基础研究项目（2024A04J9969）和广州大学研究生基础创新项目(1112333014)资助. 姚文君,博士研究生，CCF学生会员，主要研究领域为工控安全、代码分析.方滨兴，博士，教授，博士生导师，主要研究领域为计算机网络，信息安全. 吴国栋，硕士研究生，主要研究方向代码逆向、工控安全. 孙彦斌(**通信作者**)，博士，教授，博士生导师，CCF 高级会员，主要研究方向为网络安全、工控安全. 田志宏（**通信作者**)，博士，教授，博士生导师. CCF杰出会员. 主要研究方向为网络攻防对抗、APT 检测与溯源、工控安全.

behaviors. Leveraging variable correlations in designing PLC honeypoints can enhance the simulation's realism and effectiveness in deceiving adversaries. Consequently, uncovering these correlations in PLC variables is a critical task for both attacking and defending ICS infrastructure. Many research efforts on PLC variable correlations have largely followed two paths: PLC code-based analysis and causal analysis of Supervisory Control and Data Acquisition (SCADA) logs analysis. However, these traditional methods for analyzing PLC variable correlations have notable shortcomings. PLC code-based analysis, a static code analysis approach, examines the structure of PLC code to identify dependencies. While effective in revealing potential relationships, it fails to capture the dynamic features that emerge during runtime, such as variable interactions influenced by real-time conditions. This limitation can lead to an incomplete understanding of how variables truly interact in an operational ICS. Alternatively, causal analysis based on SCADA logs relies on statistical properties derived from historical data. Although this method provides insights into variable behavior over time, its dependence on statistical correlations often misses deeper causal connections, resulting in an inadequate representation of the true dependencies within the system. To address these limitations, this paper proposes a PLC variable correlation analysis method based on dependency path analysis (VCA). VCA distinguishes itself by integrating both static and dynamic features to provide a more accurate depiction of variable correlations. First, VCA parses the PLC code into an Abstract Syntax Tree (AST), a structured representation that outlines the code's syntax. From this, a Program Dependence Graph (PDG) is built to map out dependencies among code elements, establishing a foundation for static analysis. To incorporate runtime dynamics, VCA utilizes time-series data from SCADA logs. VCA employs mutual information to assign weights to the dependency edges in the PDG. These weights reflect the strength of correlations observed during system operation. Subsequently, a weighted adjacency matrix is constructed, and the total weights of paths between variables are calculated to quantify their relationships. Experimental results validate VCA's effectiveness. In an assembly line ICS, VCA successfully identified 16 pairs of strongly correlated PLC variables, surpassing the performance of prior methods. Furthermore, this paper discusses the application of VCA to PLC honeypoints. The design of PLC honeypoints must account for dynamic response mechanisms, update variable states based on their correlations, and deliberately expose certain variable correlations to lure attackers along intended attack paths. VCA provides ICS physical domain knowledge for PLC honeypoint design, simulating variable dependency logic to improve realism and deception efficacy.

Key words Industrial Control System; PLC variables; correlation analysis; program dependence graph; PLC honeypoint

1 引 言

可编程逻辑控制器(Programmable Logic Controller, PLC)变量在程序中负责了工业控制系统(Industrial Control System, ICS)中传感器数据的接收、执行器数据的发送和中间变量的存储。PLC变量之间已经被证明过是存在相互关联的关系[1]。攻击者可以利用这种相互关联的关系和PLC安全方面的脆弱性[2] [3]，劫持变量实施ICS攻击[4] [5] [6]。因此，挖掘出PLC变量之间的关联关系是极具意义的，不但可以帮助攻击者发现潜在的攻击点，也可以指导ICS防御系统发现入侵点或者故障源。例如在Triton/Trisis攻击事件中[7]，Triton恶意软件通过篡改控制逻辑器的内存变量，影响了一系列的其他变量，攻击了沙特阿拉伯一家石化工厂的安全仪表系统。通过将这些关联关系融入PLC蜜点设计，可以模拟真实的ICS环境，诱导攻击者暴露其意图和手法。

当前已经有相当的研究者人员致力于PLC变量关联关系的研究[4] [8]。识别PLC变量关联关系的一般方法是使用静态代码分析方法，去识别出PLC变量之间的相互影响关系[9] [10] [11]。这些方法可以准确地挖掘出一些PLC变量之间的因果关系，在ICS攻击或者防御上都取得了一定的应用。然而，这些方法存在局限性：它们仅仅是使用了代码的静态特征去识别出变量的影响关系，缺乏考虑代码的动态特征。例如在PLC代码中变量 *A*

和 *B* 构成依赖关系，但 ICS 实际运行中，构成变量 *A* 和 *B* 依赖关系的条件从不执行。例如变量 A 读取的是传感器的数据，但是该传感器在 ICS 运行时，一直没有读取外界信号。当前的 PLC 变量关联关系方法仍会将其视为有效关系，但这并不能识别出变量之间更加全面和深层次的影响关系。另外的一个研究方向是基于数据采集与监视控制系统(Supervisory Control And Data Acquisition, SCADA)日志的因果分析，通过利用机器学习或深度学习技术，对 SCADA 系统采集的日志数据进行因果关系挖掘，以揭示 PLC 变量间的关联性[12] [13]。然而，这类的方法仅依赖日志数据的统计特性，难以全面反映变量间的真实依赖关系。

为了解决上述问题，本文提出了一种基于依赖路径分析的 PLC 变量关联关系分析方法。该方法不仅利用了 PLC 代码的静态特征，还结合 SCADA 日志捕捉代码运行时的动态特征，从而更准确、全面地识别 PLC 变量间的关联关系。本文 VCA 方法具体过程是，首先将 PLC 解析为抽象语法树（Abstract Syntax Tree, AST），并在 AST 上添加构成依赖关系的依赖边，构造程序依赖图(Program Dependence Graph, PDG)；接着通过 SCADA 日志中的时序数据，利用互信息为 PDG 中的数据依赖边和控制依赖边定义并计算权重；最后，构建变量权重的加权邻接矩阵，计算依赖路径总权重，使用关联关系分数公式挖掘出变量间的关联关系。本文在搭建的组装线工控系统上开展实验，并验证本文方法在识别 PLC 变量关联关系的有效性和准确性。 实验结果表明，VCA 不仅准确识别了变量间的直接依赖，还捕捉了间接影响关系，并且能够全面且可靠地识别出 16 组 PLC 变量之间的强关联关系。VCA 相较于现有研究具有明显优势。此外，本文 VCA 方法可以有助于 ICS 防御系统发现入侵点或者故障源，对于指导 ICS 安全防御工作有着重要意义，尤其是在设计信息域的 PLC 蜜点时，可以提供 ICS 功能域知识，从而提高 PLC 蜜点的仿真真实性，增强对攻击者的诱捕效果。

本文研究工作的主要贡献如下：

(1) 提出了一种基于依赖路径分析的 PLC 变量关联关系分析方法，利用 PLC 代码的静态特征和 SCADA 日志中代码运行时的动态特征，在组装线工控系统实验中，VCA 成功识别出 16 组 PLC 变量的强关联关系，相较于现有研究具有明显优势。

(2) 构造了依赖边具备权重信息的程序依赖图，通过 SCADA 日志中的时序数据，利用互信息为 PDG 中的数据依赖边和控制依赖边计算依赖的权重信息。

(3) 探讨了 VCA 在工控安全中的应用，在设计信息域的 PLC 蜜点时，可以提供 ICS 功能域知识，提高 PLC 蜜点的仿真真实性，增强对攻击者的诱捕效果。

2 预备知识

2.1 PLC及PLC代码

PLC 是一种专用于工业自动化的计算设备，用于监控和控制制造过程和机械设备[14]。PLC 的核心功能通过控制逻辑语言来实现，它定义了在 ICS 中运行的规则。尽管有多家 PLC 制作商定义了不同语法规则的控制逻辑代码，例如西门子和施耐德等，这些类型的控制逻辑代码都必须符合 IEC 61131-3 标准[15]。PLC 控制逻辑代码的形式包含了 Ladder Diagram(LD), Structured Text (ST), Function Block Diagram (FBD), Instruction List (IL) 和 Sequential Function Chart (SFC)。这些类型的控制逻辑代码尽管表现形式不一致，但是都可以实现相同的控制逻辑意图。ST 代码与其他 4 种控制逻辑语言的形式对比，具有可读性强和灵活性高的优点，适合于编写复杂的控制算法。

2.2 PLC代码的执行方式

PLC 的代码执行采用周期性运行模式，通常称为循环扫描（Cyclic Scanning），如图 1 所示。

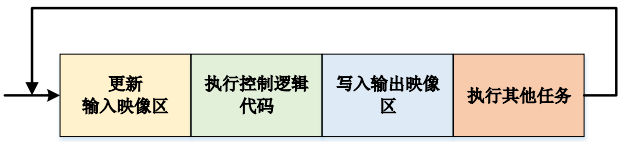

图 1 PLC 代码的循环扫描方式

图 1 中循环扫描过程包含三个主要阶段：更新输入映像区、执行控制逻辑代码、写入输出映像区。具体而言，PLC 首先从输入映像区读取传感器变量的状态，例如温度、压力或开关位置等的数值，并通过将外部输入信号映射至输入映像区，PLC 能够在每个周期开始时获取最新的环境数据。接着，PLC 执行预先编写的控制逻辑代码，根据输入的传感器数据和程序逻辑执行复杂

的计算与判断操作，包含对变量的状态进行逻辑运算、条件分支处理以及执行器指令的生成。最后，PLC 将逻辑运算结果写入输出映像区。输出映像区存储的信号随后被传输至执行器，例如电机、阀门等，以驱动设备动作或更新系统状态。此外，执行其他任务是指 PLC 的操作系统(runtime 等[16])执行其他任务，例如下载和删除块，接收和发送数据。最后，循环扫描重新开始，重复上述步骤。

3 VCA 方法设计

为了识别 PLC 变量中的关联关系，本文设计了一种基于依赖路径分析的 PLC 变量关联关系分析方法。在 PLC 控制逻辑中，传感器变量用以接收外界的传感器信号的信息，执行器变量用来存储控制逻辑运算的结果，并将该结果发送到执行器，驱动执行器。因此，在本文中，主要分析的是传感器变量到执行器变量、执行器到执行器潜在的关联关系。这里将这种关联关系表示为:

$$Relationship = f(v_{source}, v_{target}) \quad (1)$$

其中，f 是关联关系分析的方法；v_{source} 是源变量，由传感器结点 S 和执行器结点 A 的集合构成，即 $V_{SOURCE} = (S, A)$；v_{target} 是目标变量，由执行器结点 S 集合构成，即 $V_{TARGET} = (A)$.

本文 VCA 方法可以挖掘到传感器变量到执行器变量、执行器变量到执行器变量潜在的关联关系。这种关联关系可以被用来发现潜在的攻击点或故障源。如图 2 所示，本文 VCA 方法包含了 3 部分的内容：(1) 构造程序依赖图；(2) 计算依赖边权重; (3) 计算变量间关联关系分数。

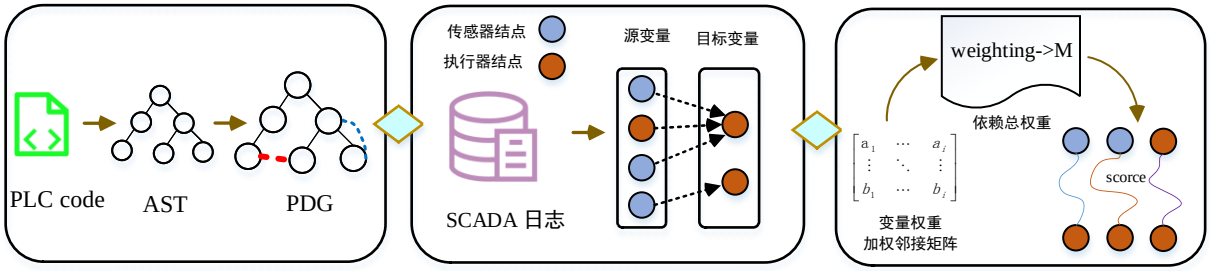

图 2 VCA 方法设计概览

VCA 方法首先将 PLC 代码解析为 AST，并在 AST 上添加构成依赖关系的依赖边，构造程序依赖图**(1)**；接着使用 SCADA 日志为依赖边定义权重，计算目标变量到源变量的依赖边的权重**(2)**; 最后，构建 PLC 变量的加权邻接矩阵，计算目标变量到源变量的依赖路径总权重，识别出变量间的关联关系**(3)**。

3.1 构造程序依赖图

程序依赖图是程序在程序中各个代码片段间数据依赖和控制依赖关系的反映。为了构造程序依赖图，在这个章节中，需要将 PLC 代码解析为 AST，接着设计在 AST 的遍历算法，识别出传感器结点和执行器结点，并将构成依赖关系的结点添加连接边。

3.1.1 解析 AST

AST 是编程代码在编译过程中的一种中间表示，通常由一些 token 构成[17]。这些 token 按照一定的连接方式，组合起来像一棵树的结构，这棵树被称为抽象语法树（AST）。在这个结构中，每个节点代表程序中的一个构造，如表达式、语句或声明等。在本文中，VCA 使用了一种名为 Parsing Expression Grammar (PEG)的工具[18] [19]将 ST 代码解析为 AST。在这些 AST 上，每一个 token 均为一个二元组<type,value>。例如在水罐 ICS 的 ST 代码中一部分代码如表 1 所示。

表 1 水罐 ICS 的 ST 代码片段

```
IF State = 1 AND LevelMeter <= 200 THEN
    Fill := Value_200;
    Discharge := Value_0;
END_IF;
```

将表 1 的代码片段代码解析成 AST，如图 3 所

示。

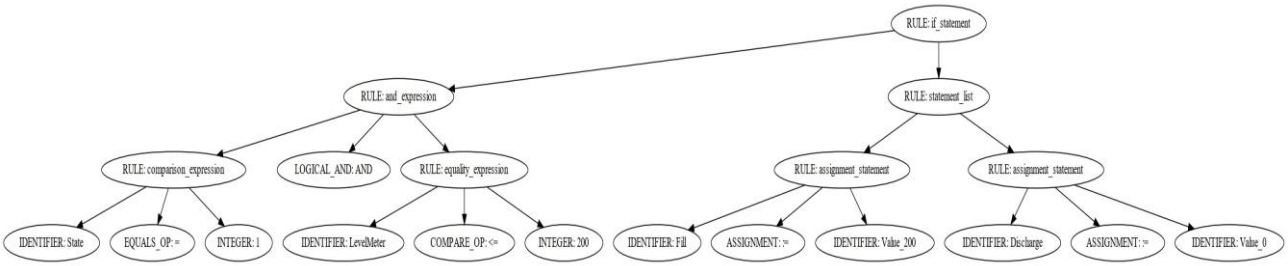

图 3　PLC 代码解析的 AST 实例

在图 3 中可以得知，该 AST 上的 token 的 type 有 *RULE* 、*ASSIGNMENT* 等类型，value 有 *if_statement*、*statement_list* 等类型。这 AST 的特点是 token 层次和连接关系分明，在上面遍历可以识别出传感器结点和执行器结点，并进一步连接它们的依赖边。

3.1.2 构造依赖图过程

PDG 是程序在程序中各个代码片段间数据依赖和控制依赖关系的反映。在构造程序依赖图之前，这里将其表示为 $G(V,E)$，其中：

a. $V=(V_{SOURCE},V_{TARGET})$ 由源变量结点集合和目标变量结点集合构成，表示了 PLC 代码中所有的源变量和目标变量。

b. $E=(e_{v_{source}}^{v_{target}})$ 是一个连接源变量结点 v_{source} 到目标变量结点 v_{target} 的有向边集合，表示 PLC 代码中目标变量到源变量的依赖关系，即目标变量依赖于源变量。

在本文中，为了挖掘变量之间的关联关系，将传感器结点 s_i 到执行器 a_j 的连接关系边、执行器结点 a_m 到执行器结点 a_n 的连接边设为依赖边。这里的依赖边包含了控制依赖边和数据依赖边。因为 PLC 的控制逻辑是根据传感器和执行器的状态值，去进行逻辑运算，从而将运算结果发送到执行器。因此，源变量结点 v_{source} 到目标变量结点 v_{target} 存在关联性的关系。

另一方面，VCA 方法构建的是 token 粒度的程序依赖图。token 粒度构成的 PDG 具备精细化分析的优点，使得每个细节（如变量、运算符、常量）都在依赖图中有明确的表示。构造 PDG 的过程需要设计 AST 的遍历算法，识别出传感器结点和执行器结点，再将构成依赖关系的结点连接。算法 1 描述了构造依赖图的过程。

算法 1.　构造 PDG.

输入：AST

输出：PDG

1: Initialize an empty *DFG, CFG;*

2: **FOR** each *token* **IN** the *AST* **DO**:

3:　*sensor_nodes*，*actuator_nodes* ← *IdentifyNodes(token)*

4:　*pair*←*ConstructPairs*({$(s_i,a_j)|s_i \in$*sensor_nodes*, $a_j \in$ *actuator_nodes*) ∪ {$(a_m,a_n)|a_m,a_n \in$*actuator_nodes*}};

5:　**IF** *CanConstructDataDependency(pair)* **THEN**:

6:　　*DFG*←*DFG*∪{(*pair*,data_dep)|*pair*=(*source*, *target*)};

7:　**IF** CanConstructControlDependency(*pair*) **THEN**:

8:　　*CFG*←*CFG*∪{(*pair*,cotr_dep)|*pair*=(*source*, *target*)};

9: **END FOR**

10: *PDG* ← *Combine*(*DFG*, *CFG*)

11: **RETURN** the *PDG*

算法 1 首先识别传感器变量和执行器变量的结点(第 3 行)。接着判断传感器结点和执行器结点、执行器结点和执行器结点是否构成了依赖关系，如果构成依赖关系，则连接它们的数据依赖边或者控制依赖边(第 4~8 行)。最后，该算法将所有数据和控制依赖关系组合到 PDG 中并返回。

此外，算法 1 构建的变量对具备传递性。即在变量对(a_m,a_n)和(a_n,a_{n+1})中，变量 a_n 是前一个变量对的目标变量，又是下一个变量对的源变量，那么可以构造新的变量对(a_m,a_{n+1})。变量 a_m 对变量 a_{n+1} 也构成了依赖关系。

图 4 是根据图 3 的 AST 实例，通过算法 1 去构建程序依赖图的结果展示。数据依赖边表示变量之间的直接数据传递关系，即源变量的值通过赋值或计算直接影响目标变量，例如图 4 中 *<IDENTIFIER: Value_200> 到 <IDENTIFIER: Fill>* 的红色指向边，表示数据依赖关系，反映在表 1 中 ST 代码 *“*Fill := Value_200*”* 中 *Value_200* 到 *Fill* 的数据依赖。控制依赖边表示条件语句对变量赋值的控制关系，即条件决定赋值是否执行，例如表 1 中“IF State = 1 AND LevelMeter <= 200 THEN

Fill := Value_200”中 *LevelMeter* 和 *State* 到 *Fill* 的依赖。在图 4 中，token<*IDENTIFIER: LevelMeter*>和<*IDENTIFIER: State*>到<*IDENTIFIER: Fill*>的*蓝色指向边，表示*了控制依赖关系。显然，这个程序依赖图是基于 token 粒度构建的。

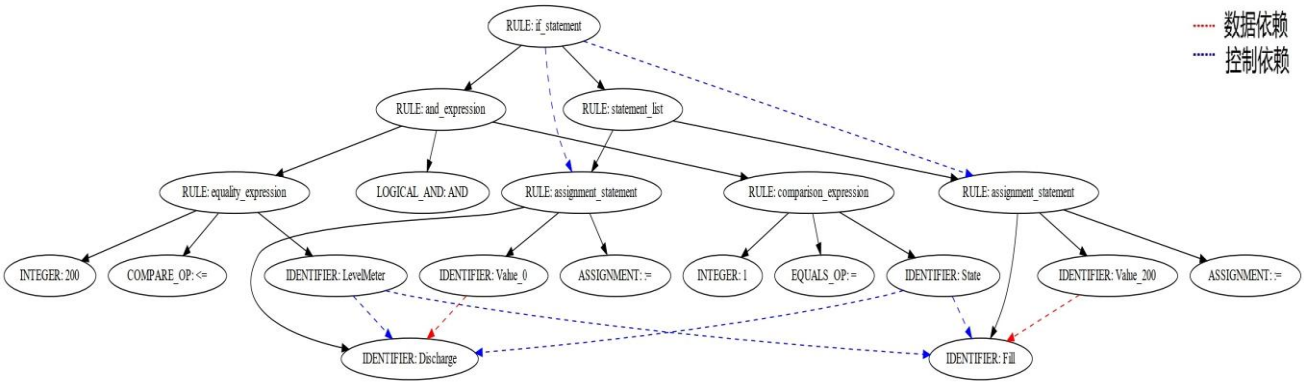

图 4 构建 token 粒度的 PDG 实例

3.2 计算依赖边权重

在 3.1 小节中已经构建了对 PLC 代码表征的程序依赖图。但这些表征属于静态的特征。由于 PLC 是在不断地从外界接口读入信息并进行处理的，这些信息是实时不断变化的。因此在程序依赖图中的依赖边是具备权重信息的。另一方面，SCADA 日志记录了变量的时序变化情况，可以分析这些时序数据的变化情况，计算变量之间的相关性和不确定性，从而为 PDG 中的依赖边赋予权重。PDG 中的依赖权重用 $W=\left(w_{v_{source}}^{v_{target}}\right)$ 表示，反应了依赖边 $e_{v_{source}}^{v_{target}}$ 所定义的依赖关系，即 SCADA 日志中传感器结点 s_i 到执行器结点 a_j 的连接关系，或执行器结点 a_m 到执行器结点 a_n 之间的关联强度的量化表达。本小节先对 SCADA 日志格式进行说明。SCADA 日志记录的时序数据采用结构化格式呈现，如表 2 所示。

表 2 集装线系统的 SCADA 数据格式示例

Time	s_1	…	s_i	a_1	…	a_j
t_1	v_{t1}^{s1}	…	v_{t1}^{si}	v_{t1}^{a1}	…	v_{t1}^{aj}
…	…	…	…	…	…	…
t_n	v_{tn}^{s1}	…	v_{tn}^{si}	v_{tn}^{a1}	…	v_{tn}^{aj}

表 2 中，“Time”字段表示记录数据采集的具体时间，格式为“YY-MM-DD HH:MM:SS”，精度为秒。其中，“YY”表示的是年份，“MM”表示的是月份，“DD”表示的是日，“HH”表示的是小时，“MM”表示的是分钟，“SS”表示的是秒。表 2 的单元格内容中，“s_1”字段表示的是 PLC 传感器变量名称，即 PLC 输入映像区的变量；“a_1”字段表示的是 PLC 执行器变量名称，即 PLC 输出映像区的变量。数值“v_{t1}^{s1}”表示的是在 t_1 时刻，变量 s_1 的值为 v_{t1}^{s1}。变量的数值类型包含了 INT（整数）、BOOL（布尔值）等。例如在章节 4“实验与评估”中，传感器变量 Lid at place 的数值类型为布尔值。

在计算依赖边权重之前，本小节首先给出依赖边的条件熵定义。对于依赖边 $e_{v_{source}}^{v_{target}}$ 的条件熵 $H(V_{TARGET}|V_{SOURCE})$ 表示在已知源变量 V_{SOURCE} 的条件下，目标变量 V_{TARGET} 的条件概率分布的熵对 V_{SOURCE} 的数学期望，即：

$$H(V_{TARGET}|V_{SOURCE})=\sum_{i=1}^{n}p_i H(V_{TARGET}|V_{SOURCE=v_{source\_i}}) \tag{2}$$

式中，$p_i=P(V_{SOURCE}=v_{source\_i}),i=1,2,\ldots,n$，表示在 SCADA 日志中，传感器变量或者执行器变量在作为源变量 v_{source_i} 时的概率分布。

进一步地，为了挖掘 SCADA 日志信息中源变量 v_{source} 到目标变量 v_{target} 的关联关系，这里使用互信息（Mutual Information, MI）对依赖边 $e_{v_{source}}^{v_{target}}$ 的权重 $w_{v_{source}}^{v_{target}}$ 进行定义。交叉互信息广泛应用于时序数据分析，反映的是一个随机变量由于已知另一个随机变量而减少的不肯定性， 在本文中适用于捕捉 SCADA 变量的相互性。 因此，依赖边的权重 $w_{v_{source}}^{v_{target}}$ 定义为：

$$I(w_{v_{source}}^{v_{target}})=H(V_{TARGET})-H(V_{TARGET}|V_{SOURCE}) \tag{3}$$

其中 V_{TARGET} 是 SCADA 日志中目标变量 v_{target_k}

的概率分布，V_{SOURCE} 是 SCADA 日志中源变量 v_{source_j} 的概率分布。

同时需要考虑到时序效应，因为 SCADA 变量间可能存在时延影响。例如源变量 v_{source_j} 的变化可能在一段时间后才能影响目标变量 v_{target_k}。传统互信息的计算忽略了时延效应，在这里引入时延 τ，计算源变量在时延 τ 对目标变量的影响。因此公式(3)所计算的依赖权重可以进一步完善。具体而言，使用交叉互信息计算不同时延下的依赖关系，即:

$$I(w_{v_{source}}^{v_{target}}) = H(V_{TARGET}) - H(V_{TARGET} \mid V_{SOURCE}^{\tau}) \quad (4)$$

式中，τ 表示时延的值，V_{SOURCE}^{τ} 表示延时后，目标变量 v_{target_k} 的概率分布。

通过 SCADA 日志中的时序数据，利用了互信息为 PDG 中的数据依赖边和控制依赖边定义并计算权重。这些权重不仅量化了变量依赖关系的强度，还能为进一步分析它们之间的关联关系提供平台。

3.3 计算变量间关联关系分数

尽管在上面中量化了源变量和目标量变量之间的依赖关系的强度，但是还不能直接应用于挖掘 PLC 变量影响关系。因为 PLC 变量之间可能存在多条可达路径，例如在数据依赖边 $e_{v_i}^{v_j}$，$e_{v_j}^{v_k}$，$e_{v_i}^{v_k}$ 中，在计算变量 v_i 对 v_k 的关联关系分数时，如果直接用依赖边 $e_{v_i}^{v_k}$ 的依赖权重计算是不合理的。因为还存在着 $e_{v_i}^{v_j}$ 和 $e_{v_j}^{v_k}$ 的依赖路径组合。这组合也可以构成变量 v_i 对 v_k 的可达路径，即通过中间变量 v_j，得到 v_i 对 v_k 的可达路径。另一方面，在依赖可达路径的计算中，可能存在路径爆炸的问题，即随着依赖图中结点和依赖边的数量增加，从一个变量到另一个变量的可达路径数量可能呈指数级的形式增长。为了解决这个问题，本小节通过计算依赖总权重的方式，用来计算变量之间关联关系分数，最终有效地挖掘出所有变量之间的关联关系。在计算关联关系分数之前，定义计算关联关系分数的公式：

$$score(v_i, v_j) = \sum_{k=1}^{n} \varphi(I(w_{v_i}^{v_j})_k) \quad (5)$$

式中，φ 表示的是计算 PLC 变量关联关系的分数，k 表示的是变量 v_i 对 v_j 的第 k 条可达路径，n 表示的是可达路径的总和。由公式(5)可知，PLC 变量的关联关系分数是变量之间所有依赖路径的权重之和。

进一步细化，挖掘 PLC 变量关联关系的具体步骤如下：

3.3.1 构建变量权重的加权邻接矩阵

利用程序依赖图中的依赖边及其权重构成一个加权邻接矩阵 $\boldsymbol{W}$，其中 $W[v_i][v_j]$ 表示从变量结点 v_i 到变量结点 v_j 的依赖边的权重。如果变量结点 v_i 到变量结点 v_j 不构成依赖关系，则 $W[v_i][v_j]$ 的值为 0。因此构建变量加权邻接矩阵为：

$$W[v_i][v_j] = \begin{cases} I(w_{v_i}^{v_j}), & \text{存在依赖边} \\ 0, & \text{不存在依赖边} \end{cases} \quad (6)$$

式中，$I(w_{v_i}^{v_j})$ 表示的是使用交叉互信息公式(4)计算不同时延下的依赖关系。

3.3.2 计算依赖路径总权重

变量结点 v_i 到变量结点 v_j 的依赖路径可能有一条，也有可能有多条。如果是多条依赖路径，需要计算所有依赖路径的总权重。计算变量的依赖权重之和 $\boldsymbol{M}$ 为：

$$M = (I - W)^{-1} \quad (1)$$

式中，$\boldsymbol{I}$ 是单位矩阵，$M[v_i][v_j]$ 表示从变量结点 v_i 到变量结点 v_j 的所有路径的总权重，反映了变量结点 v_i 到变量结点 v_j 的总关联关系。

3.3.3 设置关联阈值 threshold

通过设置一定的关联关系阈值系数，来进一步说明 PLC 变量的关联关系的强弱。判断 PLC 变量的关联关系的关联关系分数公式为：

$$Relationship = \begin{cases} score(v_i, v_j) \geq threshold, \text{强关联关系} \\ score(v_i, v_j) < threshold, \text{若关联关系} \end{cases} \quad (2)$$

式中，对于路径总权重大于关联关系阈值系数 threshold，本文将这种关联关系设为强关联关系，反之 ，设为弱关联关系。对于强关联关系的变量值，如果对变量结点 v_i 进行破坏或者篡改，那么将对变量结点 v_j 产生重要影响。

4 实验及评估

4.1 实验设置

本文构建了一个组装线系统。该系统由openPLC[20] 、FactoryIO [21]和一个 HMI 组成。组装线系统和本文方法运行的环境是 window 10 系统的 PC，运行了 16GB 内存和 AMD Ryzen 5@2.1 GHz 处理器。

该系统的传感器和执行器信息如表 3 所示。

表 3 组装线系统的传感器和执行器

变量类型	变量名称	数据类型	Modbus 数据地址(功能码)	作用
传感器	*X*	整数型	100(0x04)	感应 x 轴的位置
	Z	整数型	101(0x04)	感应 z 轴的位置
	Lid at place	布尔型	801(0x02)	盖板 A 传感器
	Base at place	布尔型	802(0x02)	底座 B 传感器
	Part leaving	布尔型	803(0x02)	产品离开传感器
	Item detected	布尔型	800(0x02)	真空检测信号
执行器	*Set X*	整数型	100(0x06)	机械手 X 轴的电机驱动
	Set Z	整数型	101(0x06)	机械手 Z 轴的电机驱动
	Lids conveyor 1	布尔型	802(0x05)	盖板 A 传送带
	Lids emitter	布尔型	807(0x05)	盖板 A 生成器
	Bases emitter	布尔型	808(0x05)	底座 B 生成器
	Bases conveyor 1	布尔型	803(0x05)	底座 B 传送带
	Bases conveyor 2	布尔型	805(0x05)	产品传送带
	Stop blade 1	布尔型	801(0x05)	盖板 A 挡板
	Stop blade 2	布尔型	804(0x05)	底座 B 挡板
	Grab	布尔型	806(0x05)	真空阀
	Parts remover	布尔型	809(0x05)	产品搬运

表 3 中描述了集装线系统中包含的传感器、执行器及其数值类型、Modbus 寄存器地址和对应的作用。

集装线系统的业务信息如表 4 所示。

表 4 组装线的业务流程及涉及到的变量

序号	业务过程	描述	涉及到的变量
1	初始状态	机械手回到 X 轴、Z 轴初始位置	*X*、 *Z*，*Set_X*、*Set_Z*
2	机械手垂直方向移动	机械手垂直向下运动	*Set_Z*
3	吸附盖板	机器手检测到盖板，并抓取该盖板	*Item_detected*、*Grab*
4	移动盖板到底座上面	机械手抓取盖板到底座上面	*X*、*Z*、*Set_X*、*Set_Z*、*Grab*
5	组装	将 A 和 B 组装成物体 C	*Grab*、*X*、*Z*、*Set_X*、*Set_Z*、*Item_detected*
6	传输产品	启动产品传送带，将产品传送走	*Base conveyor 2*、*Stop blade 2*
7	搬运产品	将产品搬走	*Parts remove*

表 4 中描述了组装线系统的业务过程以及涉及到的 PLC 变量。

4.2 实验过程

本文首先提取组装线工控系统的 PLC 控制逻辑代码，通过 VCA 方法将其解析为 AST 并构建 PDG；随后采集组装线工控系统连续 10 天的运行日志数据。利用 VCA 方法识别出存在关联关系的 PLC 变量，并将识别结果与文献[4]、文献[12]和文献[13]的方法进行对比分析，以此验证本文方法在关联关系挖掘上的准确性与有效性。

4.3 实验结果及分析

4.3.1 PLC 变量权重分析

SCADA 日志的格式如表 2 所示。在采集到 10 天的 SCADA 运行日志后，运用公式(4)对这些 SCADA 日志信息进行处理，得到变量之间的依赖权重值，即在运用 VCA 方法识别到 PLC 变量之间的依赖权重，结果如图 5 所示。

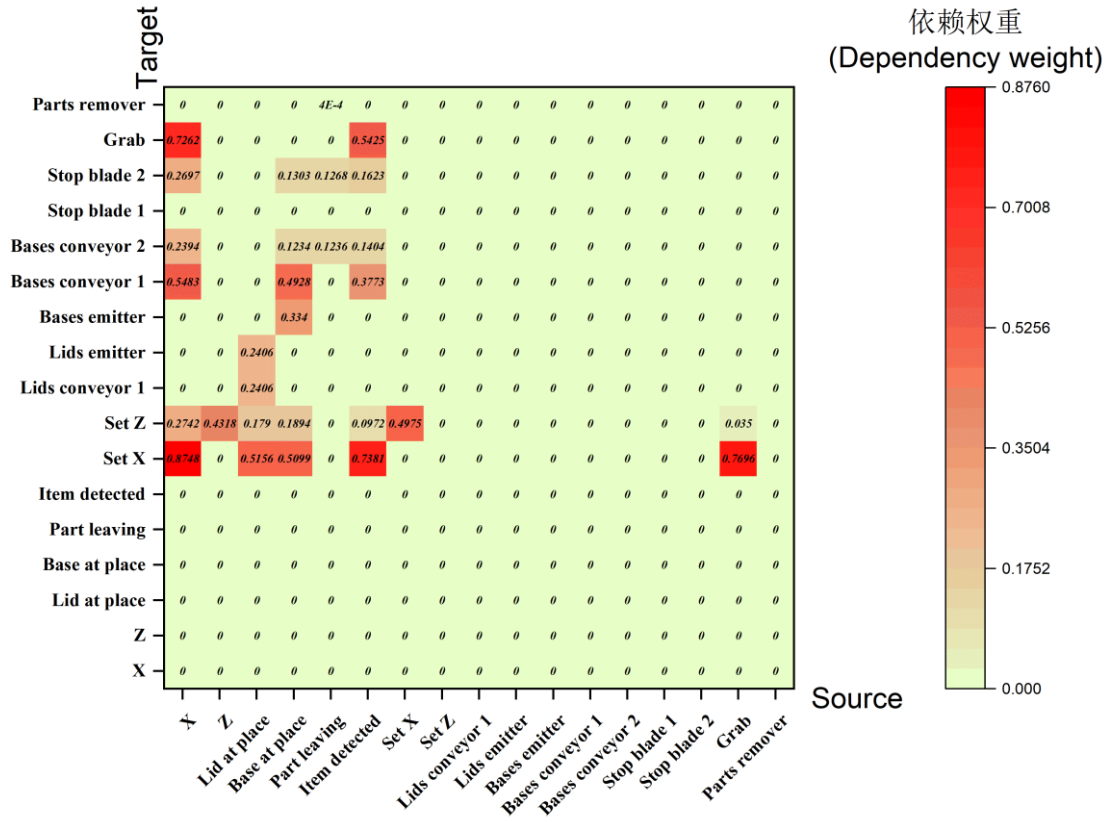

图 5　PLC 变量之间的依赖权重

在图 5 中，横轴表示的源变量，纵轴表示的是目标变量，方格颜色的深浅反映了目标变量到源变量的依赖权重分数。方格的红色越深，表示目标变量到源变量的依赖权重分数越高。这些权重分数通过公式(4)计算得出，反映了目标变量对源变量的依赖强度。例如，在横坐标为 *X*，纵坐标为 *Set X* 的方格数值为 0.8748，表示的是目标变量 *Set X* 到源变量 X 的依赖权重分数是 0.8748，显示出 *X* 对 *Set X* 具有较强的直接影响。

由图 5 可知，传感器变量之间和执行器变量到传感器的依赖权重分数都为 0，说明了它们之间不存在关联关系。这与工业控制系统的实际运行情况是符合的。传感器变量通常作为系统的独立输入，负责接收外部环境信号（如位置、状态等），不受其他变量的直接控制或影响，任何变量都不对其构成依赖关系。

尽管只有 *x* 轴和 *z* 轴上都有对应的位置传感器(传感器变量 *X* 和 *Z*)和机械手方向电机驱动(执行器变量 *Set X* 和 *Set Z*)，但是由图 5 可知，变量 *X* 对变量 *Set Z* 具有影响关系，而变量 *Z* 对 *Set X* 不具有影响关系。原因是在该 ST 构成的 PDG 中，可以找到变量 *X* 到变量 *Set Z* 的间接依赖路径(*X*→*Set X*→*Set Z*)，而 *Z* 到 *Set X* 不存在直接或间接的依赖路径，因此变量 *X* 对 *Set X* 和 *Set Z* 具有影响关系，而 *Z* 只对 *Set Z* 具有影响关系。

另一方面，图 5 也反应了 PLC 变量之间依赖的多样性。部分变量和多个变量构成依赖关系。例如，传感器变量 X 和多个目标变量（如 *Set X*: 0.8748、*Set Z*: 0.2742、*Bases conveyor 1*: 0.5483 等）具有非零的依赖权重分数。这反映了在组装线系统的控制逻辑和业务流程中，传感器变量 X 是一个关键输入变量，广泛参与多个执行器变量的控制过程。然而，部分变量不对任何变量构成依赖关系。例如，*Lids emitter* 作为源变量时，所有目标变量对其的依赖权重分数都为 0，表明其在系统中扮演输出的角色，而不是作为其他变量的控制输入。在组装线系统

中，这类变量通常用于反馈状态，而不直接驱动其他过程。

4.3.2 依赖路径总权重分析

在对组装线系统的 PLC 变量进行依赖总权重公式(7)进行计算，结果如图 6 所示。

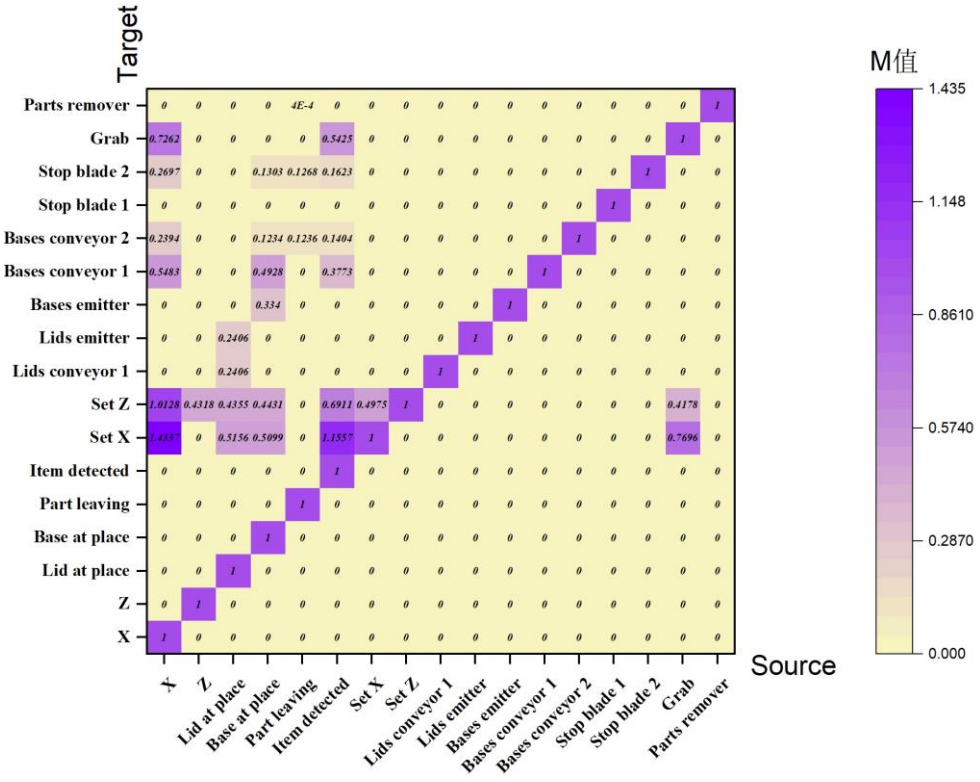

图 6 PLC 变量关联关系分数权重和矩阵

与图 5 相同，图 6 的横轴表示源变量，纵轴表示目标变量，方格颜色的深浅反映了目标变量到源变量的关联关系分数。方格的紫色越深，表示目标变量到源变量的关联关系分数越高。例如，在横坐标为 *Z*，纵坐标为 *Set X* 的方格数值为 0.4318，表示的是目标变量 *Set Z* 到源变量 *Z* 的依赖权重和是 0.4318，即关联关系分数为 0.4318。

图 6 与图 5 的数值分布趋势大体相同。例如，在图 5 和图 4 中，横坐标为 *Lid at place*，纵坐标为 *Set X* 的方格数值均为 0.5156，依赖权重和依赖权重和的数值相等。这是因为 *Set X* 到 *Lid at place* 的依赖路径只有一条，在计算依赖权重和时，直接使用了这 2 个变量的依赖权重分数。但是也存在这 2 个图在相同方格位置、不同数值的情况。例如，图 5 中横坐标为 *Grab*，纵坐标为 *Set X* 的方格数值为 0.4178；而图 4 中相同的这个位置方格数值为 0.035。这是因为变量 *Set X* 到变量 *Grab* 的依赖路径有多条，在计算依赖权重和时，叠加了所有依赖路径的权重的总值。整体而言，在相同方格位置的值，图 6 的值都比图 5 的值都要大。

4.3.3 变量关联关系分析

设置不同的阈值 threshold，可以得到不同具备强关联关系的变量对组合。在图 6 中，在进行依赖路径总权重计算时，得到变量 *Set Z* 对变量 *Z* 的依赖总权重为 0.4318，略高于 0.4。而在组装线工控系统中，变量 *Set Z* 对变量 *Z* 被确认为存在强关联关系。因此将阈值设为 0.4，能够有效区分变量间的强弱关联关系。在设置 0.4 的阈值 threshold 后，识别得到具备强影响关系的变量关联关系如图 7 所示。

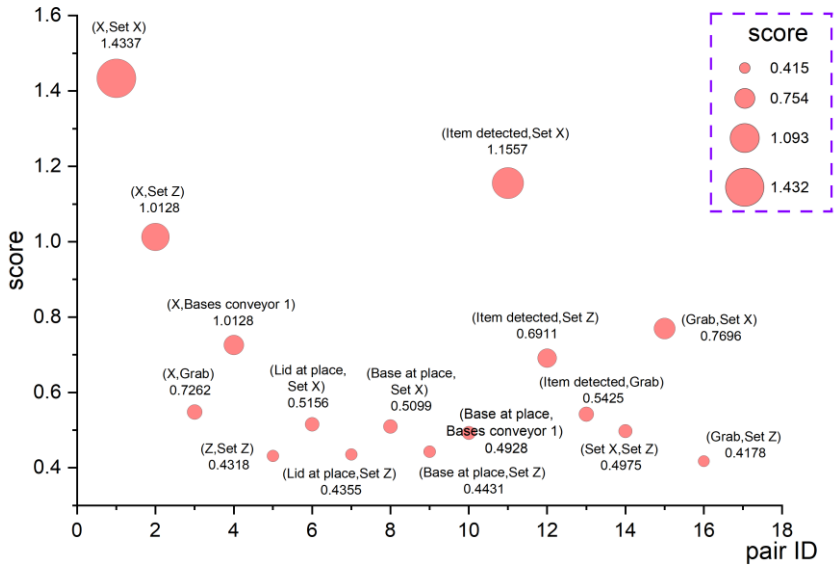

图 7　具备强影响关系的变量关联关系

图 7 中，红色圈表示本文识别得到具备强关联关系的变量对。红色圈越大，目标变量到源变量的关联关系分数越高。例如在第一个红色圈，表示目标变量 Set X 到源变量 X 的关联关系分数为 1.4337。在图 7 可知，本文方法识别到了 16 组具备强影响关联关系的变量对。本文方法和文献[4]、文献[12]、文献[13]识别出的依赖关系进行对比，结果如表 5 所示。

表 5 PLC 变量关联关系结果对比

强关联关系变量对 (源变量，目标变量)	文献 [4]	文献 [12]	文献[13]	本文 VCA 方法
(X, Set X)	√	√		√
(X, Set Z)	√		√	√
(X, Bases conveyor 1)	√			√
(X, Grab)		√	√	√
(X, Set Z)	√		√	√
(Lid at place, Set X)	√	√		√
(Lid at place, Set Z)			√	√
(Base at place, Set X)	√		√	√
(Base at place, Set Z)		√		√
(Base at place, Bases conveyor 1)	√	√		√
(Item detected, Set X)	√		√	√
(Item detected, Set Z)	√	√	√	√
(Item detected, Grab)		√	√	
(Set X, Set Z)	√		√	
(Grab, Set X)		√	√	
(Grab, Set Z)		√	√	

由表 5 可知，VCA 识别出 16 组 PLC 变量之间的强关联关系。而文献[4]识别了 9/16 组强关联关系的变量组；文献[12]识别了 7/16 组强关联关系的变量组；文献[13] 识别了 10/16 组强关联关系的变量组。具体而言，文献[4]解耦了大部分传感器变量到执行器变量的关联关系。但是对于执行器变量到执行器变量，例如*(Set X, Set Z)*变量对，该方法不能识别它们之间的关联关系。文献[12]和文献[13]虽然可以识别出传感器变量到执行器变量、执行器变量到执行器变量的关联关系，但是它们只能够识别出部分变量对的关联关系。作为对比，本文 VCA 方法可以全面识别传感器变量到执行器变量以及执行器变量之间的强关联关系。例如，本文方法成功识别了*(X, Set X)*、*(Item detected, Set Z)*等关键变量对的强关联关系，同时准确捕捉到*(Set X, Set Z)*、*(Grab, Set X)*等执行器变量间的关联，弥补了文献[4]的不足。例如，生成的 PDG 局部子图如图 8 所示。

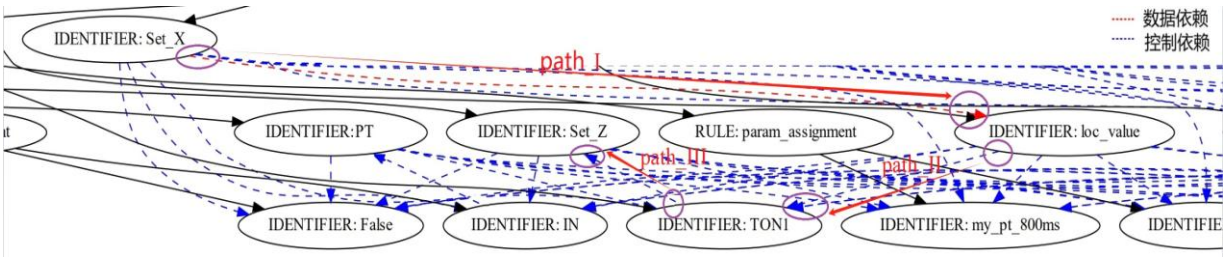

图 8 PDG 局部子图示例

在图 8 中，变量 *Set X* 到中间变量 *loc_value* 存在数据依赖关系，即 *token <IDENTIFIER: Set X>*到*<IDENTIFIER: loc_valuel>*存在数据依赖的路径，图 8 中标号 path_I 的路径(*SetX→loc_value*)；中间变量 *loc_value* 到定时器变量 *TON1* 存在控制依赖关系，即 *token <IDENTIFIER: loc_value> 到 <IDENTIFIER: TON1>*存在控制依赖的路径，图 8 中 path_II 的路径(*loc_value→TON1*)；定时器变量 *TON1* 到变量 *Set Z* 存在控制依赖关系，即 *token <IDENTIFIER: TON1>*到*<IDENTIFIER:Set Z>*存在控制依赖的路径，图 8 中 path_III 的路径(*TON1→Set Z*)。根据依赖的传递性，得出依赖关系(*SetX→loc_value→TON1→SetZ*)。因此，根据这个 PDG 子图可以得出变量关系对(*Set X, Set Z*)。

此外，本文方法在关联关系的覆盖范围上优于文献[12]和文献[13]。通过引入交叉互信息和路径权重分析，VCA 方法不仅识别了直接依赖关系，还考虑了间接影响路径，从而提高了关联关系的检测精度。例如，*(Lid at place, Set Z)*和*(Base at place, Set Z)*等变量对的识别，显示了本文方法在复杂工业控制系统中的鲁棒性。这种综合分析能力为优化 PLC 控制逻辑提供了更可靠的数据支持。

4.3.4 关联变量影响传播分析

为了理解具备强关联关系变量的行为变化情况，本小节使用中间人攻击（MITM）的方法[22]，对源变量*(X)*施加扰动，去观察具备与其具备强关联关系的变量*(Set X, Set Z, Bases conveyor 1, Bases conveyor 2, Stop blade 2, Grab)*的变化情况。变量的变化趋势如图 9 所示。

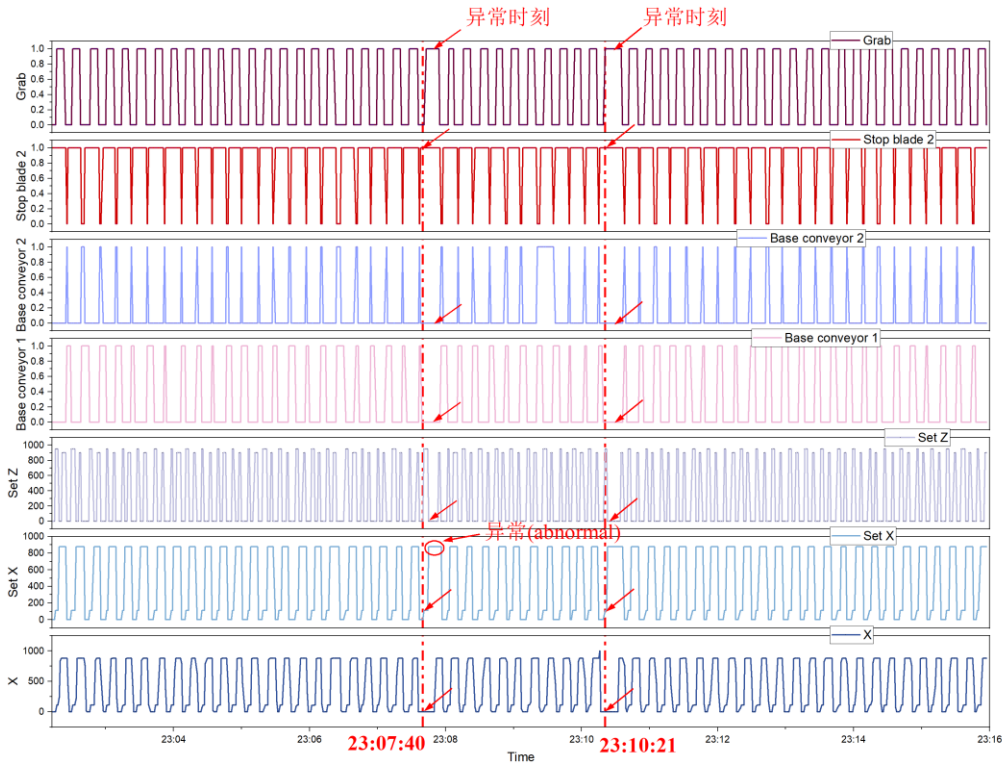

图 9 变量的时序变化情况概览

图 9 中，分别列出了变量 *X*，*Set X, Set Z, Bases conveyor 1, Bases conveyor 2, Stop blade 2, Grab* 的时序变化情况。红色虚线表示的是对变量 *X* 施加扰动干扰的时刻点，该图在“23:07:40”和“23:10:21”时刻对变量 *X* 施加了扰动。该扰动的恶意值被写进了 PLC 的输入映像区。对变量 *X* 构成强关联关系的变量，如 *Set X* 和 *Set Z* 等，分别在“23:07:40”和“23:07:42”时刻也出现了扰动的情况，即 *Set X* 在“23:07:40”时刻，从 0 跳变至 110，并明显保持在 880 的非法值的时间比其他正常运行时的长，如图 9 中红色圈中的“异常(abnormal)”部分内容。这说明了当变量 *Set X* 充当目标变量时，对源变量 *X* 构成了强关联关系。

另一方面，从 “23:07:40”到 “23:10:31”时刻的数据看，变量 *X* 的扰动（从 880 减少至 0）首先触发了 *Set X* 的快速响应（从 110 升至 880），表明 *Set X* 对 *X* 的扰动响应几乎无时延。随后变量 *Set Z* 在 23:07:42 从 0 增加到 950，表明 *Set Z* 可能受到 *Set X* 的间接影响，存在约 2 秒的时延。这一路径（*X* →*Set X* →*Set Z*）与系统中预期的依赖关系一致，反映了扰动通过中间变量逐步传播的效应。此外，*Bases conveyor 1* 在整个扰动过程中保持值为 0，未受显著影响，表明其可能仅在特定条件下（如业务过程切换）与 *X* 产生强关联，而非持续响应。扰动传播还表现出多路径效应，例如 *Grab* 在“23:07:46”和 “23:10:25”时刻的两次响应（一直保持为 1），可能受到 *X* 和其他中间变量（如 *Item detected*）的叠加影响。这不仅验证了变量间的关联强度差异，还为异常检测提供了依据。如果 *Set X* 未及时响应 *X* 的扰动，或 *Grab* 的多次响应，可能提示系统存在故障或潜在攻击。

在变量影响的时延方面(公式(4))，对源变量 *X* 施加扰动后，各目标变量的时延值如图 10 所示。

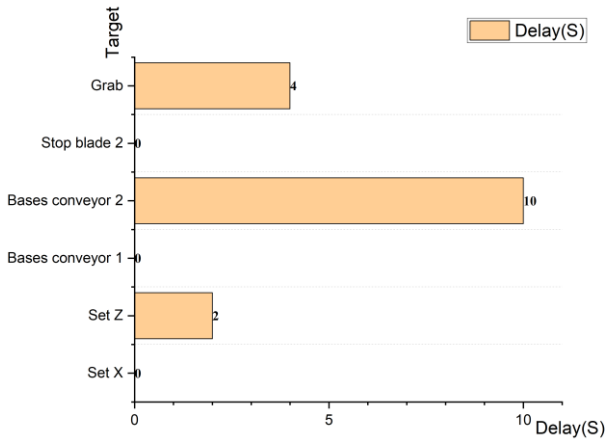

图 10 各目标变量的时延值

由图 10 可知，当源变量为 *X* 时，与 *X* 具备强关联关联的目标变量的时延值各不相同，最低的时延值为 0，如目标变量 *Set X, Base conveyor 1* 等；最高值为 10，如 *Base conveyor 2* 等。这说明了当 *X* 出现扰动时，*Set X* 和 *Base conveyor 1* 等目标变量会马上受到扰动；而 *Base conveyor 2* 会在 10 秒后受到了干扰。这可能与组装线系统业务过程的设计相关。时延为 0 的目标变量，通常是直接依赖于 *X* 的控制逻辑中的关键响应点，其状态变化由 *X*的扰动直接触发，无需经过复杂的中间处理或物理延迟。例如，*Set X* 可能作为 *X*（位置传感器）的直接控制输出，用于实时调整执行器的设定点，因此能够瞬时响应 *X* 的变化；类似地，*Bases conveyor 1* 和 *Stop blade 2*，可能位于与 *X* 直接关联的控制路径上，负责快速执行如输送带启停或阻挡动作等任务，无需等待其他变量的状态更新或机械动作的完成。

而时延值不为 0 的目标变量，如 *Grab* 和 *Set Z* 等，其时延值较高是由于它们在组装线系统中其他的过程中，需要经过额外的中间处理、状态更新或机械动作的完成。例如，*Grab* 涉及机械臂的运动控制，其响应时间受到运动轨迹规划、执行器速度或传感器反馈延迟的限制；*Set Z* 需要等待其他变量（如 *Z*，*Lid at place*）的状态确认后才能触发动作。此外，*Base conveyor 2* 的时延值为 10 秒，与其在生产线中的物理位置或功能设计有关。位于业务过程

下游(表 2 中序号为 6 的“传输产品”的业务过程)，需等待表 2 中的“组装产品”业务过程完成。而“组装产品”的业务过程又依赖于前面的业务过程完成。因此，在变量 *X* 受到干扰时，其扰动传播到 *Base conveyor 2* 需要一些时间。

5 VCA 的应用探讨

VCA 识别得到的 PLC 变量关联关系可以应用到 PLC 蜜点当中。因为 PLC 蜜点作为一种安全防御工具，可以被用来吸引、误导并分析攻击者的行为，需要模拟真实的 ICS 环境[23]。PLC 蜜点需要考虑动态响应的内容，即当攻击者对蜜点进行多次的请求时，按照 PLC 变量的关联关系更新变量的状态。另一方面，在蜜点中故意暴露某些变量的关联关系，引导攻击者沿着预设路径行动。以上均可以从 ICS 的功能域方面为设计 PLC 蜜点提供 ICS 业务场景知识。例如，在组装线系统中，运用本文的 VCA 方法可以识别出多个变量对变量 *X* 构成强关联关系，如 *Set X, Set Z, Bases conveyor 1, Bases conveyor 2, Stop blade 2, Grab* 等。运用这种强关联关系，可以指导 PLC 蜜点依次生成下面类型的 modbus 协议[24]响应报文：

(1) 模拟传感器数据变化的报文。通过 Modbus 写入指令（功能码 0x04）向 PLC 的输入寄存器（如地址 100）写入模拟的传感器数据（如变量*X*的值），诱导攻击者观察蜜点对数据变化的响应，误以为发现了关键的控制点。

(2) 触发执行器动作的报文。发送 Modbus 读取指令（功能码 0x06）从 PLC 的输出寄存器（如地址 100）读取执行器变量（如 *Set X*）的状态，引导攻击者分析执行器与传感器变量的关联关系，增加其对蜜点的兴趣。

(3) 模拟控制逻辑执行的报文。通过 Modbus 写入指令向 PLC 的内部寄存器（modbus 数据地址 803，功能码 0x05）写入控制信号，触发与变量 *X* 相关的控制逻辑（如启动 *Base conveyor 1*），诱导攻击者尝试篡改控制逻辑，暴露其攻击意图和手法。

(4) 暴露伪造的弱点报文。在 Modbus 响应中故意包含异常的变量值或状态（如 *Grab* 的错误状态），模拟系统中的潜在漏洞，吸引攻击者深入探索和利用，延长其在蜜点中的停留时间。

这些类型的报文可以诱导攻击者多次对该 PLC 蜜点扫描和分析。通过模拟真实的传感器-执行器关联关系和控制逻辑，蜜点的行为与真实 PLC 高度一致，增强了仿真真实性；同时，暴露伪造的弱点和关联关系能够吸引攻击者投入更多资源和时间，显著提升诱捕效果。因此，将 PLC 变量的关联关系融入蜜点设计，可以显著提升其仿真真实性与诱捕效果。

另一方面，本文 VCA 方法通过互信息量化了 PLC 变量间的时序相关性，为进一步挖掘变量变化趋势同步提供了基础。这些特性可能被用来进一步挖掘变量的变化趋势同步特性。例如，VCA 通过互信息识别变量间的时序依赖后，可以筛选高互信息变量对，构建时序关联网络，并提取变量时间序列，量化同步时延和强度，最终优化趋势预测模型，检测异常同步。识别出来的同步特性可应用于故障预测（识别异常趋势）和实时监控（动态校准控制）。这个过程从关联关系挖掘趋势同步规律，增强工业控制系统的诊断与优化能力，保障工业控制系统的网络安全。

6 相关工作

当前已有大量研究致力于探索 PLC 变量之间的关联关系，这些工作主要可分为两大类：基于 SCADA 日志的因果分析和基于 PLC 代码的分析方法。基于 SCADA 日志的因果分析方法主要利用机器学习或深度学习技术，对 SCADA 系统采集的日志数据进行因果关系挖掘，以揭示 PLC 变量间的关联性。这些方法通常依赖于历史数据的统计特性，能够从海量时序数据中自动提取变量间的潜在关

系，具有较高的自动化程度和适应性。例如，文献[25]采用 K2 算法评估全局报警关联结构，分析物理层中报警变量的关联关系。该方法分析出的关联结构能够较好地贴合工业流程，适用于工业环境中报警事件的自动化分析。文献[12] 提出了一种利用传递熵和贝叶斯网络的 K2 算法识别工业报警变量之间因果关系的方法。该方法简化了报警变量间的关联关系建模，进一步提高了因果分析的效率和精度。文献[13]提出了一种时空因果有向图（STCG），用于描述过程变量间的因果依赖关系。该方法通过结合知识-数据引导的强化学习来识别最优的因果结构，同时考虑了空间和时间因素以提高模型的预测可靠性和物理可解释性。文献[26]提出了一种基于 CGTST 的方法，该方法在识别工业过程中变量间的因果关系时展现了优越性能，能够有效消除虚假关联并处理噪声数据。特别地，此方法成功应用于 CCR 过程，证明了其在实际大规模化工过程中进行精确因果关系推断的有效性。文献[27]提出了一种基于生成对抗网络和变分自编码器融合的方法，用于精准分析工业系统中变量的因果关系，通过将变量映射到潜在空间，有效克服传统方法在处理复杂关系时的不足，提升因果关系判断的准确性。文献[28]提出了一种基于改进的时间卷积和多头自注意力机制（MTCMS）的框架，通过对时间卷积网络进行特征重构和跳跃连接，增强其特征提取能力，同时改进多头自注意力机制进行可量化的因果推断，从而实现对多元时间序列的因果发现和根本原因诊断。尽管这些方法能够在一定程度上识别变量关联关系，但其主要局限在于未充分考虑 PLC 控制逻辑代码的影响，仅依赖日志数据的统计特性，难以全面反映变量间的真实依赖关系。此外，这些研究工作主要集中在传感器变量和传感器变量之间的关联关系分析。

另一类研究聚焦于 PLC 代码分析，通过构建代码依赖图并结合静态分析方法[29]，识别 PLC 变量间的相互影响。例如，文献[8] [9]和[11]通过构造程序依赖图，并在依赖图上分析变量的可达路径，挖掘 PLC 变量间的关联关系。文献[10]则通过建立代码与数据间的变量映射，揭示变量依赖关系。这些方法可以准确地挖掘出一些 PLC 变量之间的因果关系，在 ICS 攻击或者防御上都取得了一定的应用。文献[30]提出通过对 PLC 和机器人代码进行静态程序分析，提取事件因果关系，生成定时事件因果图（TECG）来描述 PLC 变量间的因果依赖 。该方法考虑控制流、常量和事件持续时间等因素，有效降低事件序列搜索空间。文献[31]提出了 PLC-PROV 系统，该系统通过追踪 PLC 系统中传感器和执行器变量，收集带有时间戳的数据形成系统执行轨迹，进而生成数据溯源图来呈现 PLC 变量间的因果关系。然而，这些方法的局限性在于仅利用代码的静态特征，未考虑代码运行时的动态行为，导致分析结果可能缺乏完整性与动态适应性。此外，静态分析难以反映运行时外部输入或环境变化对变量关系的影响，例如传感器数据波动或网络延迟导致的动态行为变化。

综上所述，本文提出的基于依赖路径分析的 PLC 变量关联关系分析方法，融合了代码的静态特征与运行时的动态特征。通过结合程序依赖图与 SCADA 日志数据，采用交叉互信息和路径权重分析，本方法不仅准确识别了变量间的直接依赖，还捕捉了间接影响关系。实验结果表明，该方法能够全面且可靠地揭示 PLC 变量间的实际因果关系，相较于现有研究具有明显优势。此外，本文工作可以为信息域设计 PLC 蜜点时提供功能域知识，从而提高蜜点的仿真真实性，增强对攻击者的诱捕效果[32][33]。

7 结论

本文提出了一种基于依赖路径分析的 PLC 变量关联关系分析方法（VCA），用以解决现有研究中静态分析依赖图和 SCADA 日志因果分析的局限性。VCA 首先将 PLC 代码解析为 AST 并构造程序依赖图，随后利用 SCADA 日志的时序数据，基于

互信息为依赖边赋予权重，最后通过加权邻接矩阵和路径权重和计算，全面挖掘变量间的直接与间接关联关系。实验表明，本文 VCA 方法在组装线工控系统上能够全面且可靠地识别出 PLC 变量之间的关联关系，相较于现有研究具有明显优势。此外，本文工作可以为设计PLC蜜点时提供ICS功能域知识，从而提高这类蜜点的仿真真实性，增强对攻击者的诱捕效果。下一步将继续优化 PLC 变量关联关系分析方法，并将挖掘得到的关联关系应用到 PLC 蜜点的设计中。

参 考 文 献

[1] AN Y, QIN F, CHEN B, et al. OntoPLC: semantic model of PLC programs for code exchange and software reuse . IEEE Transactions on Industrial Informatics, 2021, 17(3: 1702-1711

[2] SUN Y, WANG H, TIAN Z, et al. Development of security protection technologies for Industrial Control System. Strategic Study of CAE, 2023, 25(06): 126-136. (in Chinese)
(孙彦斌, 汪弘毅, 田志宏, et al. 工业控制系统安全防护技术发展研究. 中国工程科学, 2023, 25(06): 126-136.)

[3] Wang Y, REN L, Feng D, et al. Security enhancement for new-generation PLC of the industrial internet integrating sensing, computing, control and intelligence: trends and perspectives. Chinese Journal of Computers, 2025, 48(3): 738-62. (in Chinese)
(王雅哲, 任磊, 冯登国, et al. 面向感算控智一体化融合的工业互联网新型PLC安全增强:趋势与展望. 计算机学报, 2025, 48(3): 738-62.)

[4] YANG Z, HE L, YU H, et al. Reverse Engineering Physical Semantics of PLC Program Variables Using Control Invariants //Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems. New York, USA, 2022: 548-62.

[5] LIU J, LIN X, CHEN X, et al. ShadowPLCs: A Novel Scheme for Remote Detection of Industrial Process Control Attacks . IEEE Transactions on Dependable and Secure Computing, 2022,19(3): 2054-2069

[6] SARKAR E, BENKRAOUDA H, MANIATAKOS M. I came, I saw, I hacked: Automated Generation of Process-independent Attacks for Industrial Control Systems//Proceedings of the 15th ACM Asia Conference on Computer and Communications Security. New York, USA, 744–758

[7] DI PINTO A, DRAGONI Y, CARCANO A. TRITON: The first ICS cyber attack on safety instrument systems//Proceedings of Black Hat. San Francisco, USA, 2018: 1-26

[8] SANG C, WU J, LI J, et al. From Control Application to Control Logic: PLC Decompile Framework for Industrial Control System. IEEE Transactions on Information Forensics and Security, 2024,19: 8685-8700.

[9] Abbas S G, Ozmen M O, Alsaheel A, et al. {SAIN}: Improving {ICS} Attack Detection Sensitivity via {State-Aware} Invariants//Proceedings of the 33rd USENIX Security Symposium (USENIX Security 24). Philadelphia, USA, 2024: 6597-6613.

[10] Yang Z, He L, Cheng P, et al. {PLC-Sleuth}: Detecting and localizing {PLC} intrusions using control invariants// Proceedings of the 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020). San Sebastian,USA, 2020: 333-348.

[11] M. Ike, K. Phan, K. Sadoski, R. Valme and W. Lee. Scaphy: Detecting Modern ICS Attacks by Correlating Behaviors in SCADA and PHYsical// Proceedings of the 2023 IEEE Symposium on Security and Privacy (SP), San Francisco, USA, 2023, 20-37

[12] DE ABREU R S, NUNES Y T, GUEDES L A, et al. A method for detecting causal relationships between industrial alarm variables using Transfer Entropy and K2 algorithm . Journal of Process Control, 2021, 106:142-154

[13] ZHANG X, SONG C, ZHAO J, et al. Spatial-temporal causality modeling for industrial processes with a knowledge-data guided reinforcement learning . IEEE Transactions on Industrial Informatics, 2024,20(4):5634-5646

[14] EPHREM RYAN A, MOHAMMAD OMAR A. A review on the applications of programmable logic controllers (PLCs) . Renewable and Sustainable Energy Reviews, 2016, 60: 1185-1205.

[15] TIEGELKAMP M, JOHN K-H. IEC 61131-3: Programming industrial automation systems. Berlin, Germany: Springer, 2010. 166: , VI390

[16] Tychalas D, Benkraouda H, Maniatakos M. {ICSFuzz}: Manipulating {I/Os} and repurposing binary code to enable instrumented fuzzing in {ICS} control applications// Proceedings of the 30th USENIX Security Symposium (USENIX Security 21). Virtual, 2021: 2847-2862.

[17] YAO W, JIANG Y, YANG Y. The Metric for Automatic Code Generation Based on Dynamic Abstract Syntax Tree. International Journal of Digital Crime and Forensics (IJDCF), 2023, 15(1): 1-20.

[18] Ford B. Parsing expression grammars: a recognition-based syntactic foundation//Proceedings of the 31st ACM SIGPLAN-SIGACT symposium on Principles of programming languages. New York, USA, 2004: 111-122.

[19] Narciso F, Rios-Bolivar A, Hidrobo F, et al. A syntactic specification for the programming languages of the IEC 61131-3 standard //Proceedings of the 9th WSEAS international conference on computational intelligence, man-machine systems and cybernetics. Stevens Point, USA, 2010: 171-176.

[20] ALVES T, MORRIS T. OpenPLC: An IEC 61,131–3 compliant open source industrial controller for cyber security research. Computers &

Security, 2018, 78: 364-379.

[21] PHILIPPOT A, RIERA B, KOZA M, et al. HOME I/O and FACTORY I/O: 2 Pieces of innovative PO simulation software for automation education//Proceedings of the 2017 27th EAEEIE Annual Conference (EAEEIE), Grenoble, France, 2017, 1-6

[22] SALEHI M, BAYAT-SARMADI S. PLCDefender: Improving remote attestation techniques for PLCs using physical model. IEEE Internet of Things Journal, 2021,8(9):7372-7379

[23] López-Morales E, Rubio-Medrano C, Doupé A, et al. Honeyplc: A next-generation honeypot for industrial control systems//Procee dings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. New York, USA, 2020: 279-291.

[24] Fovino I N, Carcano A, Masera M, et al. Design and implementation of a secure modbus protocol//Critical Infrastructure Protection III: Third Annual IFIP WG 11.10 International Conference on Critical Infrastructure Protection, Hanover, New Hampshire, USA, 2009: 83-96.

[25] WANG Y Z, ZHANG Y F, XU Y F, et al. Physical layer alarm correlation algorithm for industrial control system. Chinese Journal of Network and Information Security, 2025, 11(1): 178-188. (in Chinese)
(王英州, 张耀方, 徐有方, et al. 工业控制系统物理层报警量关联算法. 网络与信息安全学报, 2025, 11(01): 178-88.)

[26] BI X, WU D, XIE D, et al. Large-scale chemical process causal discovery from big data with transformer-based deep learning. Process Safety and Environmental Protection, 2023, 173: 163-177.

[27] SHIRSHAHI A, MOSHIRI B, ALIYARI-SHOOREHDELI M. Identification of propagation path and root cause of faults based on generative adversarial networks in industrial systems. Process Safety and Environmental Protection, 2024, 187: 1606-1617.

[28] ZHOU Y, XU K, HE F. Root cause diagnosis in multivariate time series based on modified temporal convolution and multi-head self-attention . Journal of Process Control, 2022, 117: 14-25.

[29] Li Yonggang, Zhong Yeqing, Zheng Yijian, el al. A control flow protection method based on code extraction. Chinese Journal of Computers, 2024, 47(6): 1372-92. (in Chinses)
(李勇钢, 钟叶青, 郑伊健, et al. 一种基于风险代码抽取的控制流保护方法. 计算机学报, 2024, 47(6): 1372-92.)

[30] Zhang M, Chen C Y, Kao B C, et al. Towards automated safety vetting of PLC code in real-world plants//Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP). IEEE, San Francisco, USA, 2019: 522-538.

[31] FAROOQ A A, MARQUARD J, GEORGE K, et al. Detecting Safety and Security Faults in PLC Systems with Data Provenance//Proceedings of the 2019 IEEE International Symposium on Technologies for Homeland Security (HST), Woburn, USA, 2019, pp. 1-6.

[32] Wang Rui, Yang Changjiang, Deng Xiangdong, Liu Yuan, Tian Zhihong. Development of Deception Defense Technology and Exploration of Its Large Language Model Applications. Journal of Computer Research and Development, 2024, 61(5): 1230-1249. DOI: 10.7544/issn1000-1239.202330961 (in Chinese)
(王瑞, 阳长江, 邓向东, 刘园, 田志宏. 欺骗防御技术发展及其大语言模型应用探索. 计算机研究与发展, 2024, 61(5): 1230-1249. DOI: 10.7544/issn1000-1239.202330961)

[33] Tian Zhihong, Fang Binxing, Liao Qing, et al. Cybersecurity as-surance system in the new era and development suggestions thereof: From self-defense to guard. Strategic Study of CAE, 2023, 25(6): 96−105. (in Chinese)
(田志宏，方滨兴，廖清，等. 从自卫到护卫：新时期网络安全保障体系构建与发展建议. 中国工程科学，2023，25（6）：96−105)

YAO Wen-Jun, PH.D candidate. His research is ICS security and program analysis.

FANG Bin-Xing, PH.D, professor, His research is computer network, information security.

WU Guo-Dong, Master candidate, His research is reverse engineering, ICS security.

SUN Yan-Bin, PH.D, professor, His research are network security, ICS security

TIAN Zhi-Hong，PH.D, professor, network attack and defense confrontation, APT detection and tracing, ICS security.