申请上海交通大学硕士学位论文

上海交通大学学位论文

论文作者 肖　敏

学　　号 1140329109

导　　师 邬晶副教授

专　　业 控制工程专业

答辩日期 2017 年 2 月 15 日

Submitted in total fulfillment of the requirements for the degree of Master in Control Engineering

过程控制系统的入侵检测和防御

Min Xiao

Advisor

A.P. Jing Wu

Depart of Automation, School of Electronic, Information and Electrical Engineering

Shanghai Jiao Tong University

Shanghai, P.R.China

Feb. 15th, 2017

上海交通大学
学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：_____

日　期：_____年_____月_____日

上海交通大学
学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权上海交通大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于

保　密 □，在 _____ 年解密后适用本授权书。

不保密 □。

(请在以上方框内打 ✓)

学位论文作者签名：_____　　指导教师签名：_____

日　期：_____年 _____月 _____日　　日　期：_____年 _____月 _____日

上海交通大学学位论文

摘 要

过程控制系统（PCS, Process Control Systems）的网络物理安全问题正在变得日趋严重，不管是从网络信息安全还是从物理数据完整性的角度，PCS 所面临的安全风险和入侵威胁都在不断增加，而且伴随着工业安全事故的数量逐年增加，相应带来的破坏和灾难也越发严重。所以，PCS 的网络信息安全和数据安全成为越来越需要全面和深入研究的国家基础设施和关键领域。正是在这种背景下，为了检测对 PCS 系统的入侵威胁，我们需要建立安全系统和国家关键基础设施安全的综合防御系统，本文针对 PCS 网络物理安全的入侵类型、控制器本身及其与物理设备交互数据的脆弱性特点，在研究和设计 PCS 的攻击和入侵检测方法的方面主要做了以下工作：

1. 构造了基于故障检测机制错误序列注入（FSI）攻击，其可以避开现有的系统故障检测机制，通过感染和篡改 PLC 的输入信号迫使其执行误操作并破坏控制系统的关键设备。首先我们对 PLC 控制器和物理设备之间交换的信号或堆栈数据进行采集；然后利用输入和输出向量数据库，我们辨识出与故障检测的建模方法类似的无故障的离散事件模型；最后，我们搜索所有的不可被故障机制检测的虚假序列的集合，并且通过获得适当长度的恶意序列对受控制的传感器采集的输入信号进行篡改实施攻击。
2. 设计了基于异常数据的入侵检测机制，其针对控制系统中控制器接收来自本地和远程输入信号的完整性和安全性而设计，并且考虑到错误序列注入攻击序列构造特性设计了有效的 FSI 检测算法。与攻击建模类似，首先采样数据库，辨识出能够高度复现待检测系统的无故障离散事件模型；然后针对异常数据注入攻击的特点，分为两个阶段设计异常数据检测算法。第一阶段将监测的控制器输入信号与模型的预期输出残差对比判断是否数据异常；第二阶段对隐蔽的错误序列注入攻击设计额外 FSI 检测算法来避免此类型的攻击；最后对检测存在异常的数据分析判断，定位到具体攻击源并给出相应的应对措施。
3. 设计基于规范的入侵检测设计，其针对可编程控制器本身控制程序和指令的保护并使其免受恶意代码注入而设计，只有经过验证合法的程序和指令才能从操作系统或控制服务器上传到指定的可编程控制器设备中。首先将 PLC 代码（IL code）格式化整理并通过 IL2boolIL 算法转换为中间语言；然后布尔逻辑指令表代码通过模板实例化（Template Instantiate）过程被迭代地执行转化，将通用模板代码实例化为验证工具 NuSMV 输入程序；最后我们通过将得到的形式化代码模型（NuSMV code）输入到验证工具 NuSMV 逐个地检查我们的安全规范，每个布尔规范表示有限状态机中的安全属性是否为真，如果存在任何可达的路径其属性为假，则会给出相应的反例并证明存在安全违规不能上传到 PLC 设备中。

关键词： 过程控制系统　网络物理安全　入侵检测　错误序列注入　FSI 检测　模型检测

过程控制系统的入侵检测和防御

ABSTRACT

The network physical security problem of PCS (Process Control Systems) is becoming more and more serious. The security risks and intrusion threats of PCS are increasing from the point of network information security and physical data integrity. And with the number of industrial safety accidents increasing, the corresponding damage and disaster is also more serious. Therefore, PCS network information security and data security needs more and more comprehensive and in-depth study as the national infrastructure and key areas. Based on such background, in order to detect intrusion threats to PCS system and establish the integrated defense system of security system and key national infrastructure security, this paper aims at the intrusion type of PCS network physical security, the controller itself and its interaction with physical equipment Data vulnerability, and we do the following work judged on research and design of the PCS-oriented attack and intrusion detection methods:

1. Constructed false sequence injection (FSI) attacks based on the fault detection mechanism, which can avoid the existing system fault detection mechanism, by infecting and tampering the input signal of the PLC to force it to perform mis-operation and destroy the key equipment of the control system. First, we collect the signals or stack data exchanged between the PLC controller and the physical device. Then we use the input and output vector databases to identify a fault-free discrete event model similar to the fault detection modeling method. Finally, Searching all the false sequences which can not be detected by the fault mechanism, and obtaining the malicious sequence of the appropriate length to inject the attack on the input signal collected by the controlled sensor.
2. Designed the intrusion detection mechanism based on anomaly data, which is designed for the controller in the control system to receive the integrity and security of the local and remote input signals, and designed the effective FSI detection algorithm in consideration of the construction of the FSI attack sequence. Similar to the attack modeling, we first sampled the database and identified the fault-free discrete event model which can highly reproduce the system to be detected. Then, we divide the abnormal data into two phases to design the anomaly data detection algorithm. In the first stage, the input signal of the monitored controller is compared with the predicted output residual of the model to judge whether the data is abnormal. The second stage is to design the additional FSI detection algorithm to avoid the attack. In the end, we analysis the data , locate the specific attack source and give the corresponding response measures.
3. Design the intrusion detection based on safety specifications, which is designed for the the control programs and instructions of programmable controller to protect them from malicious code injection.Only validated programs and instructions can be uploaded from the operating system or the con-

trol server To the specified programmable controller device. First, the PLC code (IL code) formatted by IL2boolIL algorithm into intermediate language; and then Boolean logic instruction code is iterative implementation of the transformation that generates the generic template code instantiation for the verification tool NuSMV input programs through the template instantiation (Template Instantiate) process; Finally, we will check our safety standards be through the formal code model (NuSMV code) input to the verification tool NuSMV one by one. Each Boolean specification denotes that whether security attributes in the finite state machine are true, if there is any attribute of the reachable path is FALSE, the corresponding counter-example is given and can not be uploaded to the PLC.

KEY WORDS: process control system network physical security intrusion detection false sequence injection FSI detection model detection

目 录

插图索引

表格索引

算法索引

主要符号对照表

$\mathbf{BEH}^m_{Ident}$	特征值为 m 的系统辨识模型
$\mathbf{BEH}^m_{Obs}$	特征值为 m 的观测信号模型
Σ	观察到的输入/输出（I/O）序列
L^q_{Obs}	序列长度为 q 的观测向量集合
X	NDAAO 有限状态集
Ω	NDAAO 有限输出变量集
f_{nd}	NDAAO 非确定性转移函数
λ	NDAAO 状态对应的输出函数
$W^n_{x_i}$	初始状态为 x_i 的 NDAAO 生成长度为 n 词语集合
$W^n(NDAAO)$	词语组成长度为 n 词组集合
$\varepsilon(j)$	NDAAO 中相邻输出向量 $\omega(j)$ 和 $\omega(j+1)$ 的变化量
$S^n_{x_i}$	从 x_i 开始的长度为 n 的错误序列
A^k	辨识参数 k 生成的错误序列集合
C_s	结构复杂度指标
C^n_A	攻击脆弱度指数
$\mathcal{T}_{\mathcal{IL}}$	IL 程序网络
$\mathcal{T}_{boolIL}$	boolIL 程序网络

第一章 绪论

1.1 研究背景及其意义

过程控制系统（PCS, Process Control Systems），也称为工业控制系统（PCS，Industrial Control Systems），是一系列由监控和数据采集（SCADA）、可编程逻辑控制器（PLC）或分布式控制系统（DCS）等设备组成的工业生产过程的控制系统，并且可以收集和传输在制造过程期间获得的数据。PCS 可以是相对简单的系统，仅仅包含具有接收输入的传感器（通常称为主传感器），处理输入的控制器和处理输出的接收器。

PCS 系统在全世界范围内不间断地监控和运行于关键性的工业基础设施领域中。PCS 系统广泛应用并服务于电能产生和交付、石油天然气精炼和管道、水分配和处理、化学加工和生产、制药、食品饮料生产、铁路运输和空中交通管制以及离散制造 [1–3]。相比更贴近家庭的“智能电网”设备正在被集成到能量输送 PCS 系统中。这些新设备直接控制电表，允许电能流入我们的家庭并实时监测个人家庭的电能消耗。在医学上，PCS 也运用于医院系统和常用的高科技医疗设备。

一方面，随着 PCS 系统越来越多的接入外部网络并且伴随着网络威胁的持续增加，PCS 系统越来越容易受到能够破坏硬件的软件入侵的攻击, 尤其是通用的计算和网络连接给关键基础设施带来全球恶意分子的电子攻击。控制系统已经被证明容易受到来自传统计算机病毒 [4, 5]、远程攻击 [6]、内部攻击 [7] 和有针对性的策略攻击 [8] 的侵入。涉及的关键攻击目标主要包括核电和改进材料 [5, 9]、运输 [6]、电力输送 [10]、制造 [4]、楼宇自动化 [7]。在一项对工业控制系统十年渗透测试结果的研究中，超过 50%的报告表明控制系统的攻击已导致高于 100 万美元的损失 [11]。在同一研究中还发现，41%的报告显示攻击可以导致“生产损失”，29%报告称网络攻击可以使 PCS 系统“丧失了查看或控制系统的能力”。在电力行业安全专家进行的调查中，发现能源供应商对他们自己的基础设施的安全性存在普遍的误解，例如他们认为控制系统与公共网络隔离，而且使用模糊的协议防止外部滥用的攻击 [12]。此外安全研究人员还发现了针对控制系统的新型攻击。最近，研究人员发现智能电表中的漏洞允许窃取能量和拒绝供电 [13, 14]，并且在电网的基本状态估计功能中所发现的缺陷允许对整个传输和分配系统进行基于仪表的攻击 [15]。

另一方面，从数据的角度来看，分布式 PCS 系统包括传感器数据采集，处理和控制命令。从现场传感器到端点功率控制网络设备（例如 PLC）的信息路径实现了诸如状态估计和设备控制的工业系统应用。由于许多原因包括错误配置、传感器故障、通信故障或蓄意的虚假数据注入攻击都会影响甚至破坏信息路径内的数据完整性。实际上，PLC 是在以控制自动化为目的的工业控制系统中广泛使用的多输入和多输出的微控制器设备。在系统的拓扑结构看来，噪声数据持续存在于系统中，但是由于存在用于检测和处理这样的数据的机制，才使得系统保持高水平的可靠性。然而最近的研究 [16, 17] 表明，恶意协调的虚假数据注入攻击可能能够绕过传统的检测噪声数据状态估计机制，所以这种攻击可能影响 PCS 系统状态估计应用程序进而操纵计算系统状态估计行为 [18–20]。在系统的物理控制器看来，存在针对网络物理平台的控制器感知虚假数据注入攻击。该攻击需要关于变电站配置的本地有限的高级信息，以及对向控制 PLC 设备反馈数据的变电站内的几个受感染的传感器进

行控制。攻击考虑了由 PLC 运行的控制器算法，以通过操纵其输入值来生成恶意控制命令。值得注意的是，故障存在于系统各种控制单元中，故障检测和诊断机制则用于检测和处理控制系统中的故障问题使得各智能控制器能够以高可靠性的状态工作。然而本文构造了一种错误序列注入攻击能够避开检测故障机制，从而通过影响 PCS 系统中控制器的执行次序来操纵可编程控制器的逻辑行为，最终达到破坏受感染系统的目的。

从过去十年中受到的工业攻击事例可以看出，现实世界的控制系统及其组件缺乏实用的安全措施。

2007 年，爱达荷国家实验室进行了极光攻击，以证明网络攻击如何破坏电网的物理组件 [21]。攻击者获得了能够访问柴油发电机的控制网络，然后运行恶意计算机程序以快速打开和关闭发电机的断路器和电网的其余部分的相位参数，从而导致柴油发电机的爆炸。

2008 年，土耳其的一条管道遭到一个强大的爆炸袭击，在含水层以上的区域溢出了 30000 桶石油。此外，它花费英国石油每天 500 万美元的过境关税。攻击者通过利用无线摄像机通信软件的漏洞进入系统，然后深入到内部网络篡改了用于向控制室报告故障和泄漏的单元，并且控制阀站处的 PLC 以增加管道中的压力，从而导致爆炸。

2010 年，Stuxnet 计算机蠕虫感染在伊朗的 14 个工业现场的 PLC，包括铀浓缩厂 [22, 23]。它通过受感染的 USB 闪存驱动器侵入到目标系统，然后 Stuxnet 通过感染可移动驱动器，在网络共享资源中复制本身并利用未修补的漏洞在网络中隐藏地传播，指示受感染的计算机连接到外部命令和控制服务器。最后中央服务器对 PLC 进行重新编程，以修改离心机的操作，以便由受损的 PLC 将它们分开 [24]。

在 2015 年，两个黑客展示了一个车辆的遥控器 [25]。零日漏洞使黑客无线控制车辆。车辆娱乐系统中的软件漏洞允许黑客对其进行远程控制，包括仪表板功能，转向，制动和传输，从而实现恶意动作，例如控制空调和音频，禁用发动机和制动器，以及控制车轮 [26]。据权威统计，截止 2015 年底，全球发生 800 余起针对工业控制系统的攻击事件，尤其是在最近几年存在越演越烈的趋势，具体如图1–1所示。

综上分析可以看出，不管是从网络信息安全还是从数据完整性的角度，PCS 系统所面临的安全风险和入侵威胁都在不断增加，而且伴随着工业安全事故的数量逐年增加，相应带来的破坏和灾难也越发严重。所以，PCS 系统的网络信息安全和数据安全已经成为越来越需要全面和深入研究的国家基础设施和关键领域。正是在这种背景下，为了检测对 PCS 系统的入侵威胁，建立安全系统和关键国家基础设施安全的综合防御系统就有了现实的依据，本文重点谈久了基于 PCS 系统的入侵检测这样一个重要课题。

本文首先提出了对 PLC 的错误序列注入攻击，通过监视并控制 PLC 和实际物理设备之间的部分受感染传感器和足够的输出信号序列构造特定破坏序列注入到测量输入信号中。值得注意的是，在现有故障检测的条件下，我们利用容错率构造的攻击可以很好的绕开故障检测和诊断机制。根据控制系统不同层面的攻击特点，本文设计了一个双层检测和防御结构，以更好地保护过程控制系统。第一层主要是基于规范的程序攻击检测来应对网络侵入并攻击控制网络的情况，只有被检测并指示安全的程序和指令才能下载到可编程控制器设备中。基于第一层检测机制，我们可以保证运行在控制系统的程序的完备性和有效性，进而可以对控制器采集信号数据库，以执行第二层基于异常的数据完整性攻击检测。值得注意的是，我们创造性地提出了防御新型的控制器错误数据注入攻击尤其

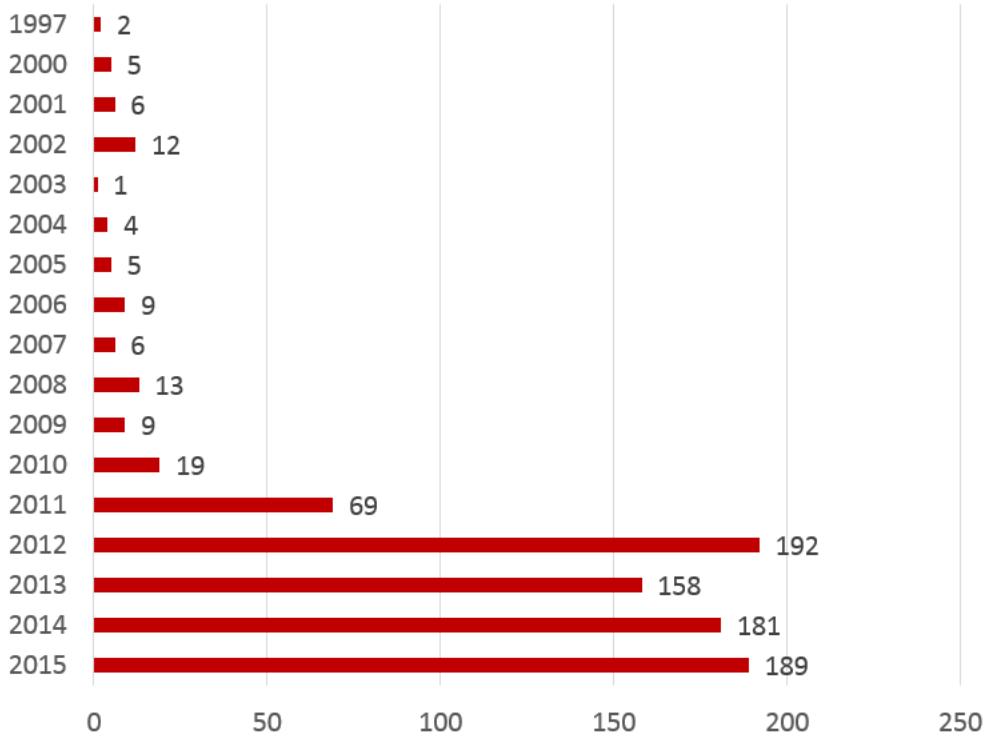


图 1–1 1997-2015 全球工业系统攻击事件.

是错误序列注入攻击的检测和诊断方法，这是传统异常检测机制无法完成的工作。通过这两层检测，我们将为 PCS 系统构建一个完整的安全系统。

1.2 过程控制系统与传统 IT 网络安全区别

在过去，PCS 使用专有硬件和软件，他们之间互连主要集中在总线通信。在 20 世纪 90 年代中期，人们将以太网和 Microsoft Windows 引入 PCS，随后开发 OPC 接口，大大简化了本地与远程以及分布式通信问题，为此 PCS 也付出了暴露于以前仅为 IT 系统所知的安全威胁为代价。

此外，随着对工业系统的攻击在过去几年中迅速增加，企业的信息安全部门通常负责整个工厂的网络安全，包括 PCS。不幸的是，并不是所有的 IT 安全解决方案都适合 PCS，因为 PCS 和 IT 系统之间存在根本的区别。此外，工业生产通常具有多个生产过程和 PCS，并且一些工业生产过程自然比其他生产过程更关键。因此，在工厂中的各种 PCS 之间处理不同的安全性并不罕见。

虽然 IT 安全的良好剂量对工业控制系统安全至关重要，但是成功地保护控制系统需要额外的机制。工业控制系统的独特之处在于它需要将安全措施与大多数 IT 系统区分开来，其中一些因素包括控制系统放置在车间，而不是数据控制中心; 它们存在置于危险环境或接近危险环境的风险; 而且设备在平台上的平均寿命是在几十年而不是几年来衡量的也是不争的事实。

参考 Belden 工业以太网基础设施设计研讨会的信息，IT 和 PCS 安全解决方案之间的差异有如下几个方面:

- 性能要求
- 可靠性要求
- 操作系统和应用程序
- 风险管理目标
- 安全架构
- 安全目标

安全目标是两者之间的本质区别。例如，IT 安全的第一目标集中于隐私，即保护数据; 而 PCS 安全的第一目标是基于安全性，即保护该过程。

根据 PCS 本身特点，我们队其安全要求的问题又细分为三大类。这些问题是:

软目标：根据 Belden，控制网络充满了所谓的“软”目标，即设备易受到网络接口干扰。许多工厂中的 PC 运行数周或数月，没有任何安全更新，有些甚至没有任何防病毒工具。此外，这些网络中的许多控制器是在网络安全不受关注的时代设计的，结果这些设备中的许多可能由于畸形的网络流量或甚至大量的正常流量而中断。

多路径：许多控制网络具有多种传输途径，网络安全威胁可以通过这些途径进入工厂。这些路径通常绕过工厂中的现有安全措施，有些甚至不出现在网络图上。例如，携带进出设施的笔记本电脑，或从一个 PC 移动到另一个 PC 的 USB 密钥。这些可以很容易地将恶意软件带入工厂，并迅速从一个系统传播到另一个系统。

平面网络：许多 PCS 网络仍然被实现为不相关子系统之间没有隔离的大且“平坦”网络。这意味着如果在设备中一部分出现问题，它可以非常快地传播到其他不相关的子系统，甚至到远程工厂站点。

根据参考文献 [27], 对于 PCS 网络和 IT 网络的不同总结如下表1–1:

1.3 国内外研究现状

1.3.1 过程控制系统现存的攻击威胁

传统过程控制系统受到的攻击计算机和网络威胁，例如内存侵入，目的是损害其一个或多个组件，以获得对系统行为的控制或者访问与进程相关的敏感数据。这里我们回顾近年来三类 PCS 漏洞的研究，第一类总体上研究 PCS 系统及其运营商的安全现状，第二类考虑 PLC 中的安全漏洞，第三类考虑传感器中的漏洞，在这种情况更侧重于智能电表，因为其是智能电网基础设施的重要组成部分。

1. PCS 总体安全现状：现有的研究 [11] 表明目前的 PCS 安全防护是有很大改进空间的。首先，因为 PCS 一直以来与因特网隔离并采用专有系统，所以经常被认为有很高的安全性。然而，使用商用操作系统（例如，Microsoft Windows 操作系统）和开放的标准网络协议使 PCS 不仅对于恶意攻击而且对因特网恶意软件的侵入感染都是开放的。例如，发生在俄亥俄州核发电站大面积破坏事件便是由于一些发电设施受到蠕虫病毒感染引起的。在 10 年内对 100 多个真实世界 PCS 进行渗透测试研究表明，在大多数情况下，这些 PCS 至少比标准补丁更新周期晚了一年。在某些情况下，将 PCS 与外部网络之间的 DMZ 隔离区域如果在几年内都没有更新的话，DoS 攻击会变得很微弱。例如，在他们对网络连接的 PLC 的评估中，发现具有 6

表 1–1 PCS 与 IT 系统网络的区别

分类	PCS 系统	IT 系统
优先级	SRA(安全性,可靠性,可用性)是满足生产要求的最重要的特性。此外,发送到 PLC 的参数的完整性也是重要的。	CIA（机密性，完整性，可用性）是 IT 中的优先考虑的安全因素。
意识差异	PCS 工程师通常不能正确地解释恶意攻击者的能力。他们经常假设任何规范不允许的事情是不可能发生的。	IT 系统工程师认为如果攻击者能够获得对设备的物理访问，则很难保护它。
风险管理	安全是主要关注的问题，即使安全问题可能会使安全受到威胁。	安全开始成为设计过程的组成部分。
安全架构优先级	终端设备（如 PLC）需要防止恶意攻击者。	数据资料需要保护。
寿命	10-20 年。IT 和 PCS 的融合可能会改变这个生命周期。	3-5 年。
实时要求	自动化设备具有严格的实时要求，因此需要满足最后期限。	通常没有严格的实时要求。
物理交互	与设备，工作人员和生产过程的重要物理交互。	与环境的少量物理交互。
资源约束	终端设备(PLC)具有有限的处理能力。然而，近年来的设备的处理能力显着增加。	IT 系统通常具有重要和高效的处理能力。
补丁管理	更新补丁很困难，因为可用性要求需要开发定制解决方案。	补丁管理现在是一个标准过程。静默更新在后台执行安装。
供应商设备支持	通常需要原始供应商的支持。	支持来自各种来源的软件供应商。
有限的物理访问组件	分布式系统使得对设备的访问困难且昂贵。	可以访问大多数 IT 系统。

字节分组的洪水攻击足以使 PLC 变得不可操作，导致所有状态丢失并迫使其重新启动。提高 ICS 安全性的另一个障碍是以下三个常见的错误 [28]：

- 认为安全性可以通过隔离实现;
- 盲目部署安全技术提高安全性，例如防火墙、加密和防病毒软件的应用经常使系统操作者产生虚假的安全感;
- 认为标准合规性可以保证系统安全，事实上即使北美能源可靠性公司提出的网络基础设施保护标准也被批评为提供虚假的安全感 [29]。

2. 对 PLC 的攻击：PLC 监视和操纵物理系统的状态，最常用的西门子 PLC 也被证明有漏洞。由于缺乏适当的会话饱满度，这些 PLC 使用的 ISO-TSAP 协议可以实现重放攻击 [30]，也可以绕过 PLC 认证上传有效载荷，并在 PLC 上执行任意命令。有研究表明用于监狱设施的西门子 PLC 也存在被操纵监狱门的可能性 [31]。
3. 对传感器的攻击：PCS 的另一个关键组成单元是收集数据并将其传送回控制单元的传感器。智能电表作为演进智能电网的广泛部署的要素具有与传统模拟电表相同的形状因子 [32] 和多个增强特征：使用时间定价 [33]、自动读表、电力质量监视和远程电源断开。现实世界智能计量系统的安全评估通常被认为是通过篡改测量值引发类似能源盗窃等一系列安全事故 [34]。存在研究表明受测系统在被实施重放攻击时允许在在仪表的内存以及飞行中进行不可检测的篡改测量值操作。后续研究检查了来自多个供应商的计量表时发现允许针对任意仪表的 DOS（拒绝服务）攻击的漏洞，而且可以对远程断开开关进行完全控制使得能够有目标地断开对客户的服务 [35]。

除了上述总结的传统安全威胁，这里我们着重考察 PCS 攻击的两个新方向。其中第一类构造了针对对手可能不具有完全访问权的 PCS 的程序篡改攻击。第二类攻击操纵传感器输入以误导控制系统尤其是可编程控制器执行动作。

1. 程序篡改攻击：一种类型的攻击旨在收集有关受害者 PCS 的情报。例如，Duqu 蠕虫病毒主要收集关于受害者系统的信息 [36]，然后将其转发到命令和控制服务器。对 PCS 的另一种类型的攻击旨在影响受害者系统的物理行为，这种攻击的最着名的例子是 Stuxnet 蠕虫，其攻击手段主要是操纵用于铀浓缩的一组离心机的参数。这样的攻击有两个阶段：感染和上传恶意程序。传统来说一旦攻击者破坏了信息系统，上传预先构建的恶意程序是不难的。这是因为攻击者通常有一个被攻击的软件的副本。然而对于 PCS 则不一定是这种情况，根据攻击者实施的攻击类型，恶意程序的构造可能容易出错或几乎不可能。恶意程序是无区别的或有针对性的，不加区分的恶意程序执行随机攻击，在受害者 PCS 的机制内引起恶意行为。有几种方式恶意软件可以在获得对一个或多个受害者 PCS PLC 的访问时自动构造不加区分的有效载荷 [37]。这里的假设是，如果恶意软件能够写入 PLC 代码区域，则它还必须能够从 PLC 代码区域进行读取。由于具有读取 PLC 代码的能力，首先恶意软件推断出被称为互锁的基本安全属性 [38]，并生成一个恶意程序，其可以违反尽可能多的安全属性。其次恶意软件识别系统中的主定时环路。考虑交通灯的示例，其中主循环确保每种颜色的光在特定时间段内依次有效。然后，恶意软件可以构造违反定时循环的恶意程序，例如通过允许某些光重叠。最后在总线枚举技术中，恶意软件使用标准化标识符（如现场总线 ID）来找到受害系统中的特定设备 [39]。

2. 错误数据注入（FDI）攻击：在 FDI 攻击中，攻击者选择一组注入到一个或多个控制器的传感器，然后向这些传感器提供精心制作的恶意值，从而使控制器获得预期的结果。例如，如果所提供的恶意值告诉控制器温度变得太低，则它将令加热元件不断增加温度，即使实际温度很好也会导致未检测到的设备过热。最早的 FDI 攻击目标是电力系统状态估计 [40]。状态估计是分布式控制系统中的重要步骤，其中基于多个可观测量来估计实际物理状态。电力系统状态估计确定电力负载如何分布在输电网络中的各种高压线路和变电站上。人为干扰相量测量单元（PMU）的子集可导致不正确的状态估计 [40]。报道了基于卡尔曼滤波的状态估计的 FDI 攻击。卡尔曼滤波器是比线性和直流系统模型更一般的状态估计形式，基于卡尔曼滤波器的状态估计器对 FDI 攻击的敏感性取决于设计系统的固有属性 [40]。如果基础状态转移矩阵包含不稳定的特征值，则系统只保证通过 FDI 攻击是可控的。这不仅对攻击具有重要的意义，而且对于针对 FDI 攻击的防御具有重要的意义，因为缺乏不稳定特征值的系统可能不会被完全攻击。从网络物理平台的角度来看，Mclaughlin 等人 [41] 通过控制器的行为模型，提出了对 PLC 的 FDI 攻击，以搜索最优输入向量来破坏控制系统。Pang 等人 [42] 提出隐秘虚假数据攻击，这可彻底破坏输出跟踪控制系统的正常运行。然而，以上两者都忽略了当前成熟的故障检测方案 [43–45]，而且它们也只是被广泛应用于智能控制器如 PLC。

1.3.2 过程控制系统中入侵检测研究现状

针对网络或恶意软件攻击的防御最早是由 Abadi 等人提出的强制控制流完整性（CFI）检测方法 [46]，这种防止代码重用的技术保证应用程序只根据预定的控制流图（CFG）执行并通过代码注入和面向返回的编程导致 CFG 的偏差来检测恶意程序攻击。后来 McLaughlin 等人提出可信安全验证器（TSV）[47] 对下载到可编程控制器的程序进行静态逻辑检测，不允许控制器执行具有存在威胁的控制器代码。在他的另一篇论文中采用 C^2 架构 [48] 来动态监测处在运行中的控制器，同时为顺序和混合控制系统提供动态参考监视器。与 TSV 一样，C^2 强制执行由工程师提供的安全规范，C^2 中的实施在运行时由位于 PLC 和 PCS 硬件设备之间的外部模块完成。Samam 等人提出了可以与复杂的控制理论模型一起工作的逆向工程方法来检测智能控制器的恶意代码 [49]。然而，他们在对控制程序进行规准化建模的时候仅仅考虑了基本指令的实现，对于复杂的控制系统无法完全地建模，从而在入侵检测时导致较高的漏报率和误检率。

从传感器和执行器存在的异常数据注入或篡改攻击的角度看，Sandberg 等人首先针对电网系统中常见的坏数据注入攻击从控制系统给定的状态估计量计算出两个安全指数。第一指数测量状态估计器的坏数据检测器如何有效地处理攻击，其中攻击者被限制到仅可以篡改极少量的测量 [50]。第二个指标用来测量坏数据检测器能够处理攻击的程度，其中攻击者只能够对测量量值进行微小的改变。Wang 等人提出了一种新的基于关系图的检测方法，无论受影响的数据是否在 SCADA 系统中的有效或正常范围内都可以防止假设数据注入攻击 [51]。Thorsley 等人证明了在特定条件下管理程序能够检测由篡改数据引起的威胁状态，以阻止异常跟踪的执行 [52]。Esmalifalak 等人提出了人工冗余操作可以减轻恶意注入数据的影响，并且这些方法已经证明对于保护较小规模的系统变量的完整性是有效的 [53]。值得注意的是，他们仅仅给出一些可用的方法来处理传统的并且蛮力错误注入的数据，而没有给特定恶意构造的数据的检测算法和机制，例如我们之前构造的可编程控制器的伪序列注入攻击。此外，从整个完整的控制系统考虑，它们忽略了用于可编程物理控制器接收到的程序

和指令的完整性，这直接影响验证或检测模型的正确性。

基于以上方法存在的不足，本文在研究了基于顺序控制系统的错误序列注入攻击算法设计了基于异常的数据完整性的攻击检测方法，并且配合基于规范的程序攻击检测方法，能够有效地对过程控制场景中的关键指令程序和信号数据进行检测，这些关键数据能直接反映 PCS 的运行状况。基于异常的数据完整性的攻击检测方法对传统的错数据攻击检测算法进行了改进，可以有效提高检测精度和降低误报率。

1.4 论文组织结构

本课题主要研究入侵检测方法检测对 PCS 系统的攻击，在充分分析过程控制系统受到的攻击和异常检测算法特点的基础上，设计改进的基于规范的程序攻击检测方法和异常的数据完整性的攻击检测方法，实现基于过程控制模型的入侵检测。本文的主要工作如下：

1. 本文论述了过程控制系统与传统 IT 网络之间的信息安全差异，分析了入侵检测的相关概念，分类和技术，并阐述了过程控制网络安全中异常检测的重要作用。
2. 提出了错误序列注入 (FSI) 攻击，可以通过感染和篡改控制器的输入信号迫使其执行误操作和破坏控制系统的关键设备，不需要渗透到防御坚固的控制网络并执行恶意代码上传的情况下便可以对系统实施有目的性攻击。与传统注入攻击不同，我们构造的攻击可以避开现有的系统故障检测机制，并利用其容错率漏洞来构造基于离散时间模型的 FSI 攻击。
3. 针对提出的 FSI 攻击以及类似隐蔽和避开故障检测的攻击，设计完整的基于异常数据入侵检测机制，并进行了基于温度控制系统的入侵检测实验，实验证明不管对设备故障还是人为恶意数据注入攻击，我们设计的机制都能有效地达到检测效果。
4. 针对可编程控制器本身控制程序和指令容易受到类似 Stuxnet 病毒等恶意代码注入，我们提出了基于规范的的入侵检测来应对控制器的恶意代码注入攻击，只有经过验证合法的程序和指令才能操作系统或控制服务器上传到指定的可编程控制器设备中。

相应的，本文分为五章，各章节的主要内容组织如下：

第一组绪论，我们从总体上介绍过程控制系统的网络安全研究的背景和重要意义，比较了过程控制系统与传统 IT 网络安全区别，并分析了国内外 PCS 现存的攻击威胁和入侵检测现状，然后我们提出本文的入侵检测设计方案。本章所介绍的过程控制系统的网络物理安全及安全防御研究现状是面向过程控制系统的入侵检测研究提出的前提和基础。

第二章基于故障检测机制错误序列注入（FSI）攻击构造。首先通过辨识出与故障检测的建模方法类似的无故障的离散事件模型，搜索所有的不可被故障机制检测的虚假序列的集合对受控制的传感器采集的输入信号进行篡改实施攻击。其可以避开现有的系统故障检测机制，通过感染和篡改 PLC 的输入信号迫使其执行误操作并破坏控制系统的关键设备。本章为第三章的异常数据检测提供攻击库和检测依据。

第三章我们设计基于异常数据的入侵检测机制，主要针对控制系统中控制器接收来自本地和远程输入信号的完整性和安全性进行防护，并且我们考虑到错误序列注入攻击序列构造特性设计有效的 FSI 检测算法。

第四章设计基于规范的入侵检测设计，其针对可编程控制器本身控制程序和指令的保护并使其免受恶意代码注入而设计，只有经过验证合法的程序和指令才能操作系统或控制服务器上传到指定

的可编程控制器设备中。我们通过从 PLC 程序转化得到的形式化代码模型（NuSMV code）输入到验证工具 NuSMV 逐个地检查我们的安全规范，每个布尔规范表示有限状态机中的安全属性是否为真，如果存在任何可达的路径其属性为假，则会给出相应的反例并证明存在安全违规不能上传到 PLC 设备中。本章检测对象是控制器本身包括控制程序和指令本身，也是对上一章仅对控制器输入和输出信号检测进行补充，有效地保证其检测的准确性和精度。

第五章总结和展望，对全文进行总结，对 PCS 的攻击和入侵检测研究作了进一步的规划和展望。

第二章 过程控制系统的错误序列注入攻击

2.1 引言

过程控制系统作为国家关键基础设施的基本组成部分已经广泛应用于工业和物联网系统中。正因为它们在现代工业社会中的关键作用，使得它们成为怀有恶意目的的攻击者的感染和侵袭的目标。传统的安全保护主要是通过多层网络防火墙和成熟的的病毒防御软件阻止网络攻击入侵工业网络。然而考虑到网络和复杂的硬件系统实施，这些方法不能完全保护网络和硬件平台免受不断变异的威胁侵入，例如 Stuxnet 病毒侵入人机界面（HMI）服务器并上传恶意代码到可编程逻辑控制器 PLC 使其损坏制造核原料的离心机。又因为 HMI 服务器通常是被安放在受良好保护的工业控制网络中，所以对其实施攻击是非常困难的。

本章我们提出的错误序列注入 (FSI) 攻击，可以通过感染和篡改 PLC 的输入信号迫使其执行误操作和破坏控制系统的关键设备。因为 PLC 的输入仅仅来自假设被 FSI 攻击注入的远程传感器，所以在不需要渗透到防御坚固的控制网络并执行恶意代码上传的情况下便可以对系统实施有目的性攻击。值得注意的是我们构造的攻击可以避开现有的系统故障检测机制，并利用其容错率漏洞来构造基于离散时间模型的 FSI 攻击。

2.2 攻击模型概述

考虑到图2–1描述的威胁模型，攻击者只需要侵入并控制遍布在远程的信号采集传感器。从过去的假数据注入攻击研究 [5]，[9]，[19]，[20]，[30] 中我们知道大部分工业控制系统包括电力系统的远程传感器如相量测量单元在实践中受到的保护较少并且分布在全国各地，因此与控制网络内的服务器相比更容易访问和侵入。我们假设攻击者知道高级基础设施配置，即连接到 PLC 传感器和执行器的输入输出的变量映射关系，这样攻击者不必渗透到控制网络中 FSI 攻击仍然可以成功。即使 HMI 服务器没有泄密而且攻击者也没有获得在 PLC 设备上上传恶意控制器程序的权限，我们构造的错误序列攻击只需要注入并控制向 PLC 发送测量信号的传感器便可以对控制系统造成不可逆的破坏。

然而困难是如何在现有控制系统中部署成熟故障检测的情况下构建攻击使 PLC 执行误动作。FSI 攻击的主要任务之一是分析并且绕开故障检测机制。通常来看用辨识精度不高的系统模型来检测随机性的故障能够得到很高的准确度，我们通过这一特性反向构造错误序列并且注入到受控制的远程传感器则可以达到使故障检测失效的目的。图2–2显示了攻击是如何构造的。我们假设攻击者可以访问在 PLC 控制器和物理设备之间交换的信号或堆栈数据。然后利用输入和输出向量数据库，我们辨识出与故障检测的建模方法类似的无故障的离散事件模型。最后，我们搜索所有的不可被故障机制检测的虚假序列的集合，并且获得适当长度的恶意序列对受控制的传感器采集的输入信号进行篡改。

2.3 系统建模和攻击构造

我们需要一个形式化的描述来特征值为 m 的系统辨识模型 $\mathbf{BEH}^m_{Ident}$ 和观测信号模型 $\mathbf{BEH}^m_{Obs}$，从形式化模型定量生成特征值大于 m 特征向量。辨识的目标是使得在给定便是参数 k 的条件下特征

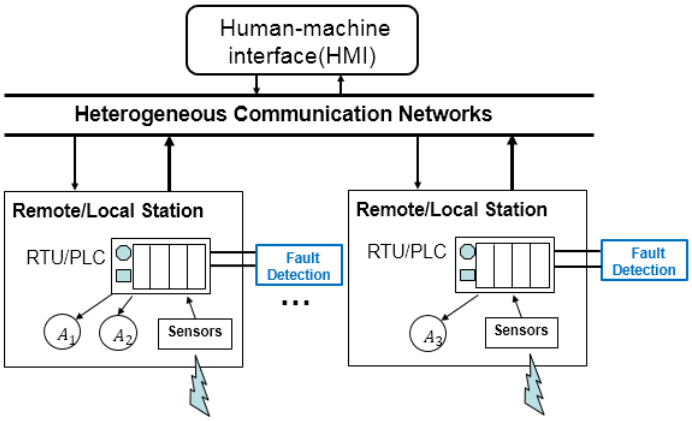


图 2–1 FSI 攻击威胁模型

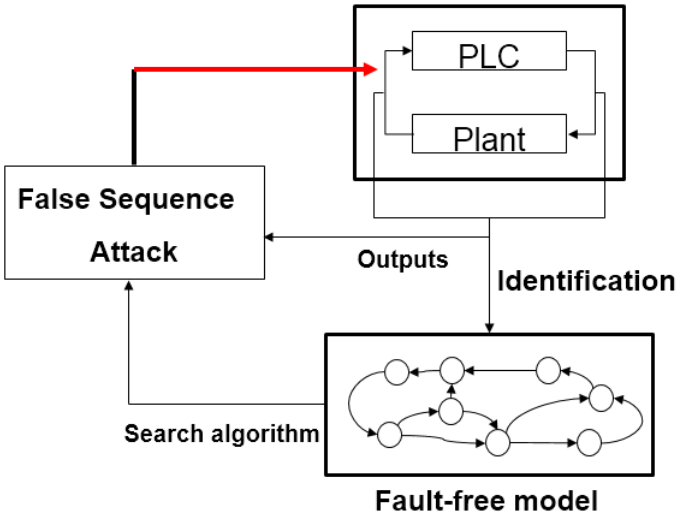


图 2–2 FSI 攻击构造示意图

值超出 k 的特征向量数目达到最小，理想情况是 $\mathbf{BEH}^m_{Ident}$ 与 $\mathbf{BEH}^m_{Obs}$ 相等。最后我们在辨识得到的系统模型和观测信号模型基础上搜索 FSI 攻击序列。

2.3.1 信号采集和观测特性定义

我们通过在 PLC 控制器获取信号之后以固定时间间隔读取信号来采样数据 [43]。图2–3显示了从 PLC 采样 I/O 向量序列的最常用的方法，通过 OPC 通信模式收集在服务端接收到的 I/O 数据形成标识数据库。

在实现数据收集之后，我们需要定义观察到的输入/输出（I/O）序列、采样数据的语义和词义。首先我们引入以下定义，

定义 1: r 个输入和 s 个输出的采样数据的观测 I/O 序列集合定义为:

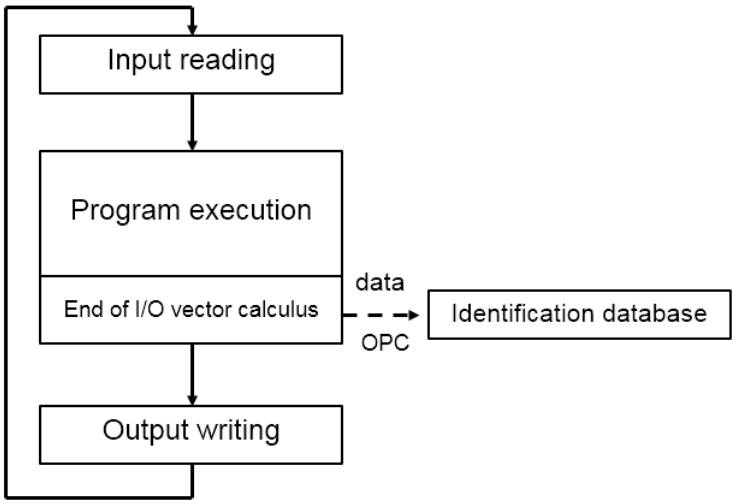


图 2–3 对 PLC 采样数据示意图

$$\Sigma = (\gamma_1, ..., \gamma_p) \tag{2–1}$$

这里 $\gamma_i = (u_i(1), u_i(2), ..., u_i(|u_i|)), u_i(j)$ 是向量 u 第 i 个分量的第 j 个输出，其中 $u = (I_1, ..., I_r, O_1, ..., O_s) = (IO_1, ...IO_m)$ 且 $m = r + s$。

我们假设对于 $\forall$ 两个连续的 I/O 向量 $u(t) \neq u(t+1)$ 成立当且仅当至少一个 I/O 向量分量改变且 I/O 向量是新生成的。

定义 2: 观测到的采样数据词义集合表示和语义集合表示: 观测的词义可以用 I/O 序列长度为 q 的观测向量集合描述

$$L^q_{Obs} = \bigcup_{\gamma_i \in \Sigma} \Big(\bigcup_{t=1}^{|\gamma_i|-q+1} (u_i(t), u_i(t+1), ..., u_i(t+q-1)) \Big)$$

有了词义的形式化表示后，我们定义长度为 n 的采样数据的语义的表示:

$$\mathbf{BHE}^n_{Obs} = \bigcup_{i=1}^{n} L^i_{Obs} \tag{2–2}$$

2.3.2 模型辨识

2.3.2.1 模型选择

模型辨识的目的是确保辨识的任意长度为 m 的语义表示 $\mathbf{BEH}^m_{Ident}$ 等于观测到的任意长度为 m 的语义表示 $\mathbf{BEH}^m_{Obs}$，其中 m 可以是任何正整数。简而言之，所辨识的模型在精度足够高的情况下能够完美复现基于 PLC 的过程控制系统。考虑到过程控制系统通常是可编程的控制器和物理设备的耦合系统，可编程的控制器可以被认为是确定性的，而物理设备通常被认为是非确定性的。因此，物理设备和控制器的耦合系统一定是非确定性的。因此，我们提出了适合于辨识过程控制系统的非确定性自发输出自动机（NDAAO）[45]。

定义 3: NDAAO 是由 5 元组函数表示:

$$NDAAO = (X, \Omega, f_{nd}, \lambda, x_0)$$

with

- $X = x_0, ..., x_{|x|-1}$ 是有限状态集
- $\Omega = \omega_1, ..., \omega_{|\omega|}$ 是有限输出变量集
- $f_{nd} : X \to 2^X$ 是非确定性转移函数
- $\lambda : X \to \Omega$ 是状态对应的输出函数
- x_0 是初始状态

NDAAO 可以由图 $G = (V, E)$ 的形式表示，图 G 的顶点集是 NDAAO 所有的状态集，有向边集是由非确定性转移函数 f_{nd} 组成，即

$$E(G) = \{(x_i, x_j) \in X \times X : x_j \in f_{nd}(x_i)\}$$

每个节点对应一个状态并包含状态对应的输出，图 2–4通过一个简单的例子来展示图形化的 NDAAO 模型。

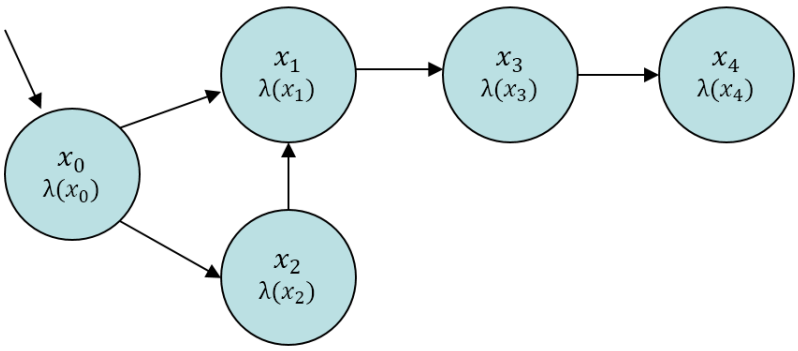


图 2–4 NDAAO 图形表示

定义 4: NDAAO 的词义和语义表示: 由初始状态为 x_i 的 NDAAO 生成的长度为 n 词语集合表示为

$$W_{x_i}^n = \left\{ \begin{array}{l} w \in \Omega^n \mid w = \big(\lambda(x(1), ..., \lambda(x(n)))\big) : \\ [\exists \big(x(1), ..., x(n)\big) : x(1) = x_i \in X, \; and \\ \forall \, 1 \le t \le n-1, x(t+1) \in f_{nd}(x(t))] \end{array} \right\} \tag{2–3}$$

然后由词语组成的长度为 n 词组集合表示

$$W^n(NDAAO) = \bigcup_{x_i \in X} W_{x_i}^n \tag{2–4}$$

有了词组集合，我们可以获得 NDAAO 的长度为 n 语义集合表示

$$\mathbf{BEH}_{Ident}^n = \bigcup_{p=1}^{n} W^p(NDAAO) \tag{2–5}$$

定义 5: 辨识事件向量 $\varepsilon(j)$ 是 NDAAO 中相邻两个辨识输出向量 $\omega(j)$ 和 $\omega(j+1)$ 的变化量，用公式表示为 $\varepsilon = \omega(j+1) - \omega(j)$. 输入事件向量 $I(\varepsilon(j))$ 是相邻两个辨识输入向量 $I(j)$ 和 $I(j+1)$ 的变化量，同理输出事件向量 $O(\varepsilon(j))$ 也有相似的定义. 具体的公式表示为:

$$\varepsilon(j) = \bigcup_{l=1}^{m} \begin{cases} I_l\_1 \ or \ O_l\_1, \ if \ I_l(j+1) - I_l(j) = 1 \\ I_l\_0 \ or \ O_l\_0, \ if \ I_l(j+1) - I_l(j) = -1 \\ \epsilon, \ if \ I_l(j+1) - I_l(j) = 0 \end{cases} \tag{2–6}$$

考虑包含两个输入和一个输出的 I/O 向量序列 γ，我们得到

$$\gamma = (A, B, C) = (\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix})$$

该序列可以简化表示成 $A \xrightarrow{I_1\_1} B \xrightarrow{I_2\_0, O_1\_1} C \cong A \xrightarrow{I_1\_1, I_2\_0, O_1\_1} C$

2.3.2.2 辨识算法

本节我们提出了一个辨识算法用来生成上一节中所描述的 NDAAO 模型。我们定义辨识参数 k 用于确定用于生成新状态的 I/O 向量序列的长度。如果生成的 NDAAO 模型的长度为 k 的语义集合正好与观测时采样数据产生的长度为 k 的语义集合一致的话，这样的 NDAAO 模型被称为 k 完全可观测的。

NDAAO 的构造主要分为三个步骤。首先，我们将观察到的序列转换为一组长度为 k 的词组序列并在每组的第一个状态前创建一组长度为 $k-1$ 伪状态以确保与其他词语一致。算法 1 的第一部分表示了观测序列的转换过程。然后我们开始执行 NDAAO 辨识操作，将 NDAAO 的状态集与长度为 k 的词组集相关联，并且转移函数由长度为 $k+1$ 的词语表示。算法2–1的第二部分表示了 NDAAO 的辨识过程 。最后，我们合并等效状态来对状态空间进行降维，对于任意两个不同的状态 x_i 和 x_j，如果它们相关联的输出相同并且它们具有相同的后继状态集，则我们可以将这两个状态合并到一个状态。我们通过从 NDAAO 的状态集和转移函数绘制状态转移图来使模型可视化。具体过程如算法2–2所示。

例子 1. 考虑从无故障例子系统中采样三组向量序列: $\gamma_1 = (A, B, C, D, E, A)$, $\gamma_2 = (A, B, D, C, D, E, A)$, $\gamma_3 = (A, D, B, C, D, F, E, A)$。大写字母代表不同的 I/O 向量。这里我们选取辨识参数 $k = 2$，经过序列转换后我们得到

$$\gamma_1^{k=2} = (AA, AB, BC, CD, DE, EA)$$
$$\gamma_2^{k=2} = (AA, AB, BD, DC, CD, DE, EA)$$
$$\gamma_3^{k=2} = (AA, AD, DB, BC, CD, DF, FE, EA)$$

算法 2–1 NDAAO 构造算法

输入: 观测的 I/O 序列（采样数据）Σ 和辨识参数 k

输出: NDAAO 模型和状态转移图 $G=(V,E)$;

1: // **第一部分: 观测序列的转换过程**
2: **for** each $\gamma_i \in \Sigma$ **do**
3: **if** $u_i(1) \neq u_i(|\gamma_i|)$ **then**
4: 从 Σ 删除 γ_i; Return;
5: **else**
6: $$\alpha_i(t)=\begin{cases} u_i(1), & for\ \ 1 \le t \le k-1 \\ u_i(t-k+1), & for\ \ k \le t \le k+|\gamma_i|-1 \end{cases}$$
7: **for** $m=1\ \ to\ \ |\gamma_i|$ **do**
8: $w_i(m)=(\alpha_i(m),...,\alpha_i(m+k-1))$;
9: **end for**
10: $\gamma_i^k=(w_i(1),...,w_i(|\gamma_i|))$;
11: **end if**
12: **end for**
13: $\Sigma^k=\bigcup_{i=1}^{|\Sigma|}\gamma_i^k$;
14:
15: **第二部分:NDAAO 的辨识过程** 初始化状态集 $X=\emptyset$, 转移函数 $f_{nd}(x_0)=\emptyset$, 输出 $\Omega=\emptyset$, 初值状态 $x_0=\Sigma^k[0][0]$, 节点集 $N=\emptyset$ 和边集 $E=\emptyset$;
16: **for all** ξ such that $\xi \in \Sigma^k$ **do**
17: **for all** η such that $\eta \in \xi$ **do**
18: $X \leftarrow X \cup \xi$; $\Omega \leftarrow\ \Omega \cup \eta(|\eta|)$;
19: **end for**
20: **end for**
21: **for all** δ such that $\delta \in \Sigma^{k+1}$ **do**
22: **for all** ψ such that $\psi \in \delta$ **do**
23: $x \leftarrow \psi[1-k]$;$f_{nd}(x)=\psi[k-|\psi|]$;
24: **end for**
25: **end for**

和

$$\begin{aligned}\gamma_1^{k=3}&=(AAA,AAB,ABC,BCD,CDE,DEA)\\ \gamma_2^{k=3}&=(AAA,AAB,ABD,BDC,DCD,CDE,DEA)\\ \gamma_3^{k=3}&=(AAA,AAD,ADB,DBC,BCD,CDF,\\ &\qquad DFE,FEA)\end{aligned}$$

在获得 $\Sigma^{k=2}$ 和 $\Sigma^{k=3}$ 之后，我们从 $\Sigma^{k=2}$ 得到状态集并且通过算法2–1的第二部分从 $\Sigma^{k=3}$ 得到

算法 2–2 状态空间降维和图形化表示

1: **for all** x_i, x_j such that $x_i, x_j \in X and i \neq j$ **do**
2: **if** $\lambda(x_i) = \lambda(x_j)$ 和 $f_{nd}(x_i) = f_{nd}(x_j)$ **then**
3: 合并 x_i 和 x_j, 从 X 中删除 $x_i(x_j)$ 并且用 $f_{nd}(x_{pre}) = x_j(x_i)$ 替换 $f_{nd}(x) = x_i(x_j)$, 这里 x_{pre} 是 $x_i(x_j)$ 的前继;
4: **end if**
5: **end for**
6: $V \leftarrow V \cup X$;
7: $E \leftarrow E \cup (x, f_{nd}(x))$;
8: 对 $G(V, E)$ 画图;

转移函数。相应的例子示意如图 2–5所示。

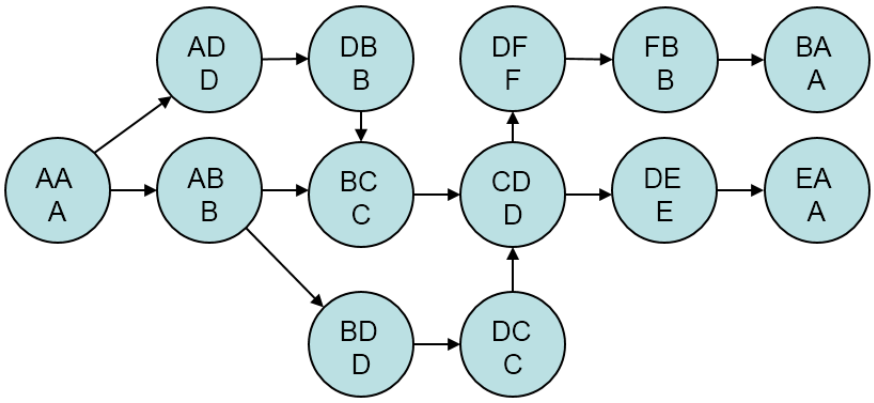


图 2–5 经过两步辨识算法得到的 NDAAAO 模型图示

初始辨识的模型相比真实系统存在冗余的状态集和边集, 因此需要根据算法2–1的第三部分来减少状态空间以简化初始模型。在例 1 中如果我们将状态 DB 与状态 AB、状态 BA 与状态 EA、状态 BC 与状态 DC 均合并，并且对任意 $x_i \in X$ 用 x_i 替换长度为 k 的向量词语。我们得到的简化 NDAAO 模型如图 2–6所示。

2.3.3 FSI 攻击的构造

在我们辨识出无故障的 NDAAO 之后，我们可以利用故障检测的原理构造不可检测的错误序列。我们知道，故障检测是根据当前观测的 I/O 向量与辨识的 NDAAO 的预期输出是否一致来确定是否存在故障的。如果结果为真，则认为当前观测向量安全，否则给出警报可能存在故障。在我们的攻击方法中，我们构造了错误序列注入攻击，其不仅满足序列输出来自辨识的模型以确保不能被故障机制检测到，而且还通过选择具有恶意执行逻辑的适当长度的错误序列注入到控制器的输入信号中

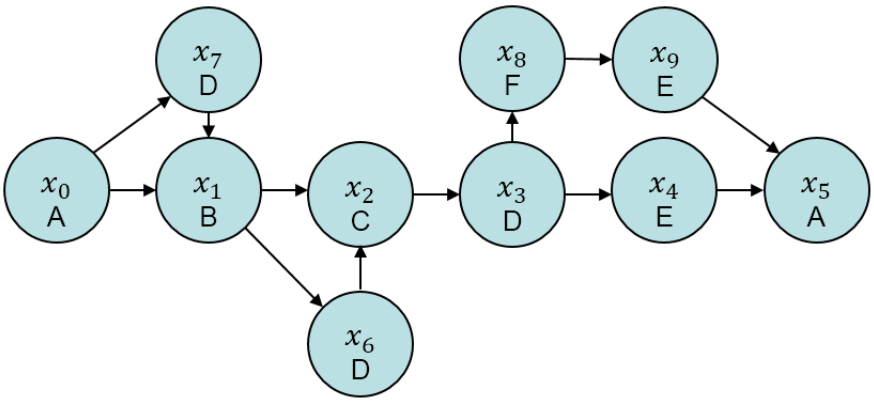


图 2–6 最终简化的 NDAAO 模型图示

来破坏系统。从 x_i 开始的长度为 n 的错误序列的具体形式描述如下定义，

$$S_{x_i}^n = \left\{ \begin{array}{l} s \in \Omega^n \mid s = \big(\lambda(x(1), ..., \lambda(x(n)))\big) \wedge \\ (s \notin L_{Obs}^n) : [\exists \big(x(1), ..., x(n)\big) : x(1) \\ = x_i \in X, \ and \ \forall \ 1 \leq t \leq n-1, \\ x(t+1) \in f_{nd}(x(t))] \end{array} \right\} \tag{2–7}$$

上述定义的 $S_{x_i}^n$ 从辨识的 NDAAO 模型中截取长度为 n 的序列并且不同于观测到的 I/O 向量集的任何长度为 n 的序列。因此，定义的错误序列当注入到控制器的输入信号时能够对控制系统造成潜在的入侵破坏。由于降维的 NDAAO 是 $k+1$ 完全可观测，故以下等式成立:

$$\forall \, m \leq k+1, \ \mathbf{BEH}_{Obs}^m = \mathbf{BEH}_{Ident}^m \tag{2–8}$$

根据上述理论，只有当 $S_{x_i}^n$ 中长度不小于 $k-2$ 的序列才可以是错误序列。我们基于辨识参数 k 获得从 NDAAO 生成的所有错误序列集合为

$$A^k = \bigcup_{x_i \in X} \big(\bigcup_{n=k+2}^{max(|\gamma_d|)} (S_{x_i}^n) \big), \ \gamma_d \in \Sigma \tag{2–9}$$

在我们得到错误序列的定义之后，下一步是提出搜索算法以获得所有不可被故障机制检测的错误序列集合。从算法2–3可以看出使用 FSI 递归遍历算法的递归公式可以获得长度不小于 $k+2$ 的错误序列，这些序列来自于 NDAAO 模型选择且不存在于开始状态为 x_{init} 的观测 I/O 向量的序列的任意子序列。通过获取 FSI 递归遍历算法的输出，我们合并所有 X 中始于状态 x_{init} 的 $S_{x_{init}}$ 的序列集合 A^k，具体表达式如下，

$$A^k = \bigcup_{x_{init} \in X} \big(S_{x_{init}} \big) \tag{2–10}$$

通过算法2–3，获得的示例 1 的错误序列的集合是

$$(A, D, B, D, ...), (..., D, C, D, F, ...), (A, B, C, D, F, ...)$$

算法 2–3 FSI 递归遍历算法

输入: 辨识的 NDAAO 模型, 观测的 I/O 序列 Σ, 辨识参数 k 和初始状态 x_{init}

输出: 错误序列集合 $S_{x_{init}}$;

for each $x \in x_{init}$ **do**
  $x_{init} = f_{nd}(x)$
  seq.append(x)
  $IncSearching(NDAAO, \Sigma, k, x_{init})$
  if $|seq| \geq (k+2)$ and $seq \notin substring(\gamma)\ for\ \forall\gamma \in \Sigma$ **then**
    $S_{x_{init}}$.append(seq)
  end if
  seq.pop()
end for
Return $S_{x_{init}}$

其中字母之前或之后的省略号可以是字母的任何前导或后继。

因为攻击者可能对来自控制系统的传感器的控制有限，我们需要确定在从上述方法产生的错误序列的两个连续向量之间变化的 I/O 向量分量。

2.3.4 可行性和性能指标

本文攻击的可行性主要来自两方面。首先随着辨识参数 k 的增加，NDAAO 的状态空间迅速增加并且需要采样庞大的顺序系统进程周期信号数据才能使模型收敛到稳定水平 [45]。因此，当在精度值 k 较大的条件下进行在线检测时往往需要大量的计算。我们的工作主要实现离线攻击，有足够的时间进行计算。其次，考虑到检测机制的稳定性和性能，在实际工业系统中通常一个较小的参数 k 足以满足基于故障检测的实际要求 [43]，也正是因为这两个条件，较小的参数 k 能够为我们的攻击的构造提供良好的可行性。

为了评估 NDAAO 的连通性，我们给出由一个状态产生的后继边的平均数量的信息。我们定义结构复杂度指标 C_s 如下:

$$C_s = \frac{\sum_{x_i \in X}(deg(x_i))}{|X|} \tag{2–11}$$

这里 $deg(x_i) = |f_{nd}(x_i)|$ 是状态 x_i 的度.

与复杂性指标类似，我们定义攻击脆弱度指数来衡量从已辨识的 NDAAO 模型中搜索错误序列的成功率。攻击脆弱度指数定义如下所示:

$$C_A^n = \frac{|\bigcup_{x_i \in X}(A_{x_i}^n)|}{|W_{Ident}^n|} \tag{2–12}$$

通过方程 (11) 和 (12)，我们可以获得所有状态的多重分支状态的比率和所有辨识的 I/O 序列集中错误序列的比率。

2.4 实验仿真

为了证明所提出的错误序列攻击方法的可行性和性能，我们给出了相应的案例仿真。考虑的系统是一个小型的货物分拣系统，该系统的功能是根据尺寸大小分拣包裹。系统有 11 个输入（来自物理设备的输出信号）和 5 个输出（从 PLC 输入到物理设备的信号）。图2–7显示了货物分拣系统的结构示图。

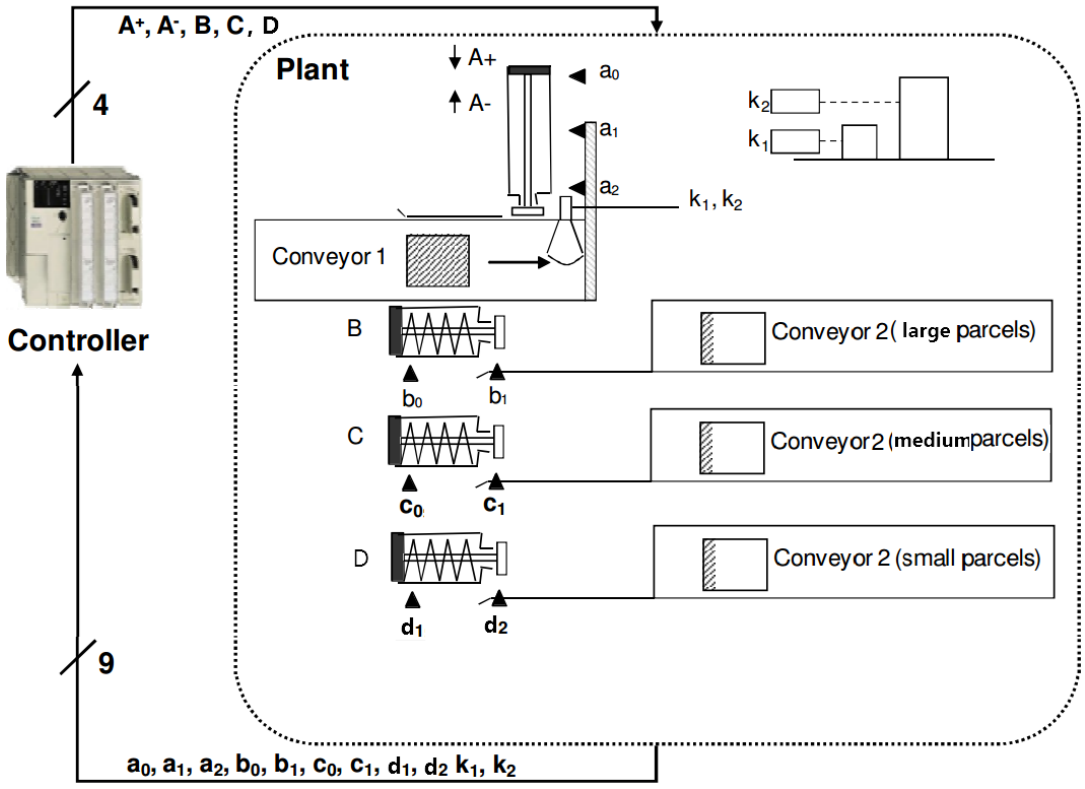


图 2–7 货物分拣系统的结构示图

辨识数据库是由 50 个采样的系统运行周期组成并且每个周期的采样时间均为货物到来和分拣时刻。系统的输入和输出变量向量为 $[A+, A-, B, C, D, k_1, k_2, a_0, a_1, a_2, b_0, b_1, c_0, c_1, d_0, d_1]$。在完成辨识过程和状态空间降维后，我们得到货物分拣系统完整的 NDAAO 模型，由于篇幅所限部分模型如图2–8所示。

通过获得的模型，我们执行了错误序列搜索过程，在简化之前我们搜索了大量不同长度的假序列，这些错误序列可以作为潜在的恶意逻辑顺序注入到受控传感器中。例如

$$A_1 = Q_{34}Q_{21}Q_{22}Q_{30}$$
$$A_2 = Q_{32}Q_{10}Q_{11}Q_{12}Q_{28}$$
$$A_3 = Q_{16}Q_{17}Q_{18}Q_{19}Q_{20}Q_{34}$$

这里 Q_i 是每个观测周期的输出向量。

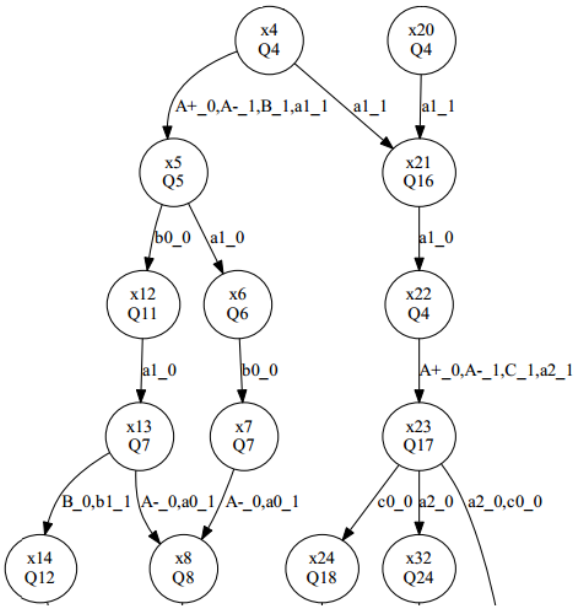


图 2–8 货物分拣系统的部分 NDAAO 模型

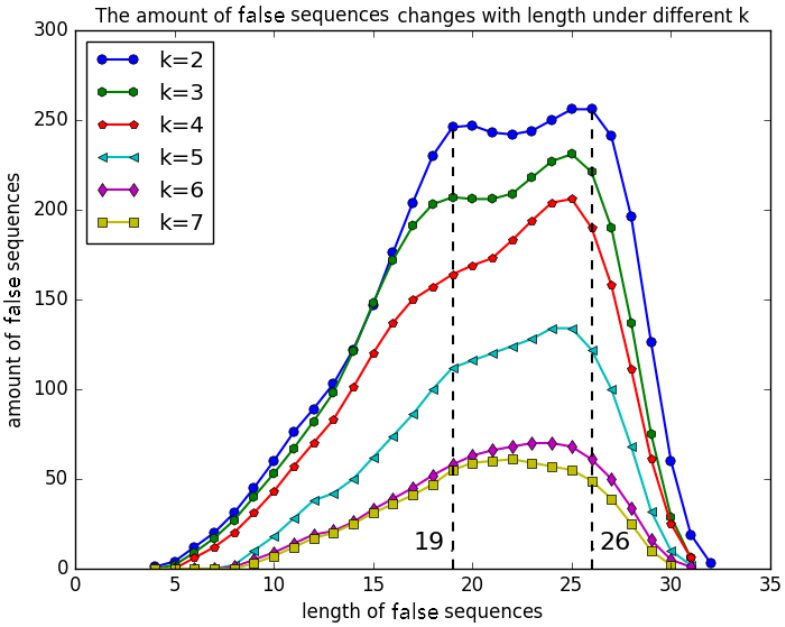


图 2–9 不同数量的错误序列数量

然而如果攻击者实施攻击，必须需要保证上述序列所有的传感器受到感染和控制，这在实际操作中是很难实现的。因此我们需要使用来自等式（6）的驱动事件向量简化处理如上所述在序列集，

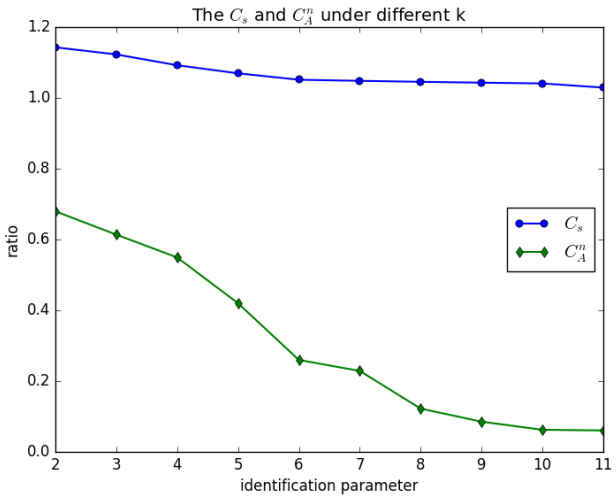


图 2–10 C_s 的 C_A^n 关系

结果如下:

$$A_1 = Q_{34} \xrightarrow{d1\_1,b0\_1,\{b0\_0,d0\_1\}} Q_{30}$$
$$A_2 = Q_{32} \xrightarrow{d1\_1,b1\_1,k2\_1,\{b1\_0,d0\_1\}} Q_{28}$$
$$A_3 = Q_{16} \xrightarrow{\{k1\_0,a0\_0,c1\_1d1\_0\},a1\_1,a1\_0,k2\_1,\{a2\_1,c1\_0\}} Q_{34}$$

这里，用逗号分隔的每个箭头的符号序列是在错误序列的每两个向量之间的单个驱动事件输入的集合。当我们注入这些错误序列到 PLC 控制器的输入信号时，在没有被故障机制所检测的情况下，分拣结束后将获得错误的分拣结果。从获得的假序列，我们发现大多数假序列集中在 19 和 26 之间的长度。所以潜在的恶意攻击可以选择这种长度的假序列。图。2–9显示了不同 k 下假序列变化的长度。考虑到良好的攻击可行性需要较大的状态度和较多的多分支状态，图2–10显示了在结构复杂度度量 C_s（多分支状态的比率）减小的情况下，攻击脆弱度指数（错误序列的占比）快速下降。因此我们可以通过在多个分支状态上添加特定的检测机制来避免这种攻击，而不是一味地提高辨识参数 k。

2.5 本章小结

在本章中，我们基于过程控制系统提出了一种错误序列注入攻击，所获得的错误序列攻击将被用作注入到与 PLC 连接的远程传感器接收的信号来对系统进行恶意逻辑攻击。我们给出了包括 FSI 递归搜索算法在内的错误序列注入攻击构建的整个实现过程。值得注意的是对多个分支状态的检测将成为对这种攻击的有效防御。实验仿真表明我们的方法是对部署故障检测的控制系统造成一定的破坏性威胁并证明我们提出的方法的有效性。

第三章 基于异常数据的入侵检测设计

3.1 引言

在上一章节中，我们讨论了过程控制系统在远程输入信号采集和传输过程中遭受的数据注入攻击并阐述如何构造较为隐蔽的错误序列注入攻击。事实上大多数控制应用（例如监控和数据采集SCADA）都需要保证严格的安全性，因为一旦由任何异常故障或恶意攻击引起的错误甚至恶意行为都会导致物理设备甚至整个系统遭受无法逆转的破害。在本章节中，我们主要针对控制系统中控制器接收的来自本地和远程输入信号的完整性和安全性设计了基于异常数据的入侵检测机制，并且考虑到 FSI 攻击序列构造特性我们设计了有效的 FSI 检测算法。

3.2 入侵检测方案概述

考虑到前一章节中描述的威胁模型，来自远程终端单元（RTU）或 PLC 的远程传感器的信号数据也面临数据注入攻击威胁，我们将入侵检测机制放在 PLC 或者 RTU 可编程控制器与物理设备连接的输入和输出两端。从传感器测量的每个数据或信号都需要经过监测并验证完整性，一旦存在未能通过检测机制的观测数据立即给出报警并定位可能受攻击的传感器位置。具体实现时，与构造攻击采用的模型辨识类似，首先收集 PLC 控制器与系统物理设备之间传输的信号和内存数据形成输入和输出向量采样数据库，辨识出能够高度复现待检测系统的无故障离散事件模型。然后针对异常数据注入攻击的特点，分为两个阶段设计异常数据检测算法。第一阶段将监测的控制器输入信号与模型的预期输出进行残差对比来判断是否数据异常；第二阶段对隐蔽的错误序列注入攻击设计额外 FSI 检测算法来避免此类型的攻击。最后对检测存在异常的数据分析判断，定位到具体攻击源并给出相应的应对措施。

3.3 系统建模和入侵检测设计

3.3.1 攻击类型

对于异常数据注入攻击，主要存在两种类型。一类是*蛮力和类故障攻击*，例如由 Mclaughlin 等人提出的拥有控制意识的 FDI 攻击 [41] 和由 Pang 等人设计的伪装假数据攻击 [42]。然而这类攻击是实现具体的任务和目标是销毁受控系统，但是没有考虑受感染系统本身的安全机制，所以很容易被成熟的故障检测机制检测到 [43, 44]。另一类是*隐蔽和避开故障检测的攻击*，这类攻击的典型代表就是我们前一章节构造的错误序列注入攻击。这种类型的攻击要求攻击者熟悉的受感染系统的物理变量与控制器连接配置信息并能够访问传感器，因此它可以避开系统故障检测。

3.3.2 检测方法

我们在图3–1中创建了一个两阶段检测的流程示意图，用于描述对上一节提到的两种类型的攻击进行检测的过程。一方面图3–1显示了结合观测到的 I/O 向量数据库和辨识的离散事件模型，我

们首先对 a）类蛮力和类故障攻击进行残差对比检测。另一方面我们需要验证当前观测的向量值并判断其映射到模型中的状态的前一状态是否存在多个后继状态，对应到状态转移图即当前对应的状态只是上一状态的某个分支，如果是的话我们针对 b）类隐蔽和避开故障检测的攻击进行特定的 FSI 检测。最后在经过两个阶段检测之后，我们将检测结果进行分析并定位以获得攻击源的具体位置。

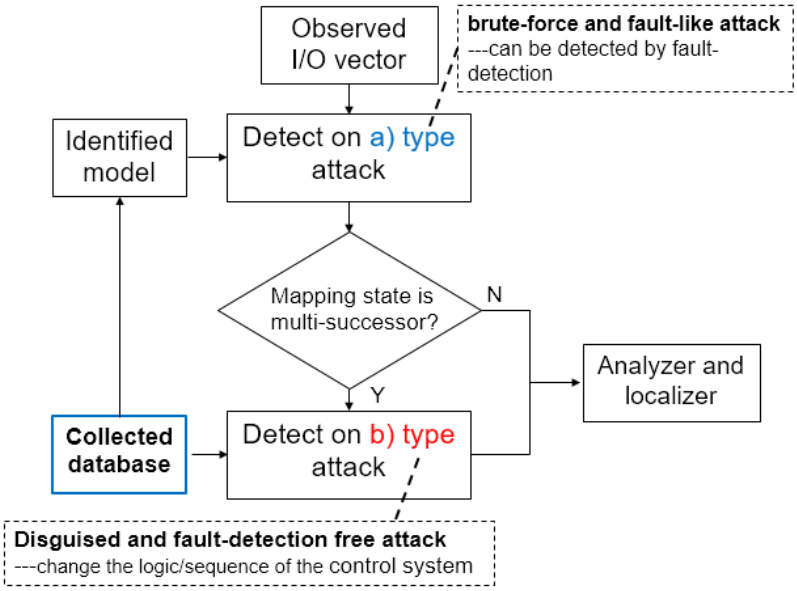


图 3–1 基于异常数据检测的流程示意图

考虑到我们的主要检测目标是能够改变控制系统的逻辑或进程的错误序列注入攻击，由于 FSI 攻击来自辨识的模型的相同输出确保其不会被故障机制直接检测，进而通过选择具有恶意执行逻辑的适当长度的错误序列注入到控制器的输入信号中来破坏系统。算法3–4显示了基于异常数据的检测过程。第一步是通过类似构造攻击模型辨识过程来获得待检测系统 NDAAO 模型，然后根据残差计算每个当前观测的 I/O 向量值与模型的预期输出是否一致。如果不一致则认为在当前模型的状态估计集为空即不存在任何可行的下个状态，此时很可能遭受蛮力和类故障攻击，否则我们需要检查先前状态估计集模 $|\hat{X}_{t-1}|$ 的大小是否大于 1，如果是的话可能存在错误序列注入攻击并进行第二阶段 FSI 攻击的检测。

由上章节中 FSI 序列的定义式（2-7）中长度为 n 的序列 s 满足的必要条件是 $s \notin L_{Obs}^{n}$，所以我们需要增加记录每一观测向量值并判断从观测周期初始向量到当前观测向量组成的观测序列 ϕ 是否存在于原始用来辨识模型的采样序列库中，如果不存在的话则可以判断系统受到 FSI 攻击。为了使检测更加有效，我们只保留每次检测时序列满足 φ 属于 γ' 任意子字符串的 γ' 并将 γ' 保存于 Σ'，主要作用是删除没有用到的验证序列集并保留有效的序列以验证观测序列 ϕ 是否属于采样序列库的子串。此外由于介于相邻两个多后继状态之间的序列是唯一的，我们仅需要判断 φ 和 γ 的多后继映射向量（包括当前向量）序列可以更快地进行验证。最后判断当前估计状态集 $|\hat{X}_t|$ 是否是否等于 1，如果不是的话我们把控制系统视为受到隐蔽和避开故障检测的攻击，否则控制系统将安全地通过检测即没有受到任何攻击。

算法 3–4 基于异常数据的检测算法

输入: 辨识的 NDAAO 模型, 新观测的 I/O 向量 $\varphi(t)$, 模型的状态估计 $|\hat{X}_{t-1}|$, 和采样序列库 Σ

输出: 是否通过检测，如果没有的话系统可能受到的攻击类型并给出受攻击的路径和位置 $locatePair$。

if $|\mathbf{X}_{t-1}| > 0$ **then**
  $\mathbf{X}_t \leftarrow \{x \in X | \exists x_{pre} \in \mathbf{X}_{t-1}\colon x \in f(x_{pre}) \wedge IO(\lambda(x_{pre}), \lambda(x)) = IO(\lambda(x_{pre}), \varphi(t))\}$
  if $|\hat{X}_t| = 0$ **then**
    $locatePair \leftarrow locatePair((\varphi_{init}, ..., \varphi(t)), IO(\lambda(x_{pre}) \cup IO(\lambda(x_{pre}), \varphi(t)))$
    return 蛮力和类故障攻击
  end if
  if $deg(\hat{X}_{t-1}) > 1$ **then**
    $\phi \leftarrow (\varphi_{init}, ..., \varphi(t))$
    for each $\gamma \in \Sigma$ **do**
      保留满足 $\varphi\gamma'$ 子串的 γ' 并且 $\Sigma' \leftarrow \gamma'$
      $\phi_{ms} \leftarrow$ 保留对应到模型中的多后继向量（包括当前向量）的 ϕ 的序列
      $\gamma_{ms} \leftarrow$ 保留对应到模型中的多后继向量（包括当前向量）的 γ 的序列
      $\phi \leftarrow \phi_{ms}, \gamma \leftarrow \gamma_{ms}$
      if $\phi \notin substring(\gamma)$ **then**
        $\hat{X}_t \leftarrow \emptyset$;
        break
      end if
    end for$\Sigma \leftarrow \Sigma'$
  end if
else
  $\hat{X}_t \leftarrow \{x \in X | \lambda(x) = \varphi(t)\}$
end if$\hat{X}_{t-1} \leftarrow \hat{X}_t$
if $|\hat{X}_t| = 1$ **then**
  return 安全通过检测
else
  $locatePair \leftarrow locatePair(\phi, IO(\lambda(x_{pre}) \cup IO(\lambda(x_{pre}), \varphi(t)))$
  return 隐蔽和避开故障检测的攻击
end if

对于攻击定位，一旦存在任何攻击我们需要分析并存储相关的受攻击路径和传感器。值得注意的是，我们可以通过比较采样的序列库和保存的受攻击路径（序列）来定位具体哪个部分受到攻击。为了实现更好的定位精度，我们还需要保存表示当前观测向量和前一个向量之间的驱动事件向量分量，例如温控设备开关和温度信号等物理变量。此外相应的采样序列的驱动事件变量分量应该被记录以定位所有可能的感染的传感器。

3.4 实验仿真

为了证明所提出的基于异常数据的检测方法的性能，我们给出了基于 Dspace 实时仿真平台的攻击检测验证实验。DSPACE 实时仿真系统由德国 dSPACE 公司开发，基于 MATLAB/Simulink 控制系统开发及硬件和软件仿真硬件和软件工作平台，与 MATLAB/Simulink/RTW 完全无缝连接。DSPACE 实时系统具有强大的实时性、高可靠性、良好的可扩展性等特点。DSPACE 硬件系统具有高速计算能力，并配有丰富的 I/O 支持，用户可根据需要进行组合; 软件环境，功能强大且易于使用，包括自动代码生成/下载和测试/调试的工具集。DSPACE 硬件和软件现在已成为快速原型设计和实时平台的硬件在线仿真的首选。图3–2展示了 Dspace 图片且 Dspace 详细的技术规则如表3–1所示。

图 3–2 Dspace 图片

仿真的系统是一个温度控制系统，其功能是满足温度要求，并将繁忙时间峰值功率分配给 5 个子系统的非繁忙时间。系统有 10 个输入（来自系统的测量值）和 5 个输出（来自控制器到执行器的信号）。图3–3显示了温度控制系统的简要结构。图3–4介绍了温度控制系统的 Simulink 仿真图。

首先，我们对控制系统实行蛮力和类故障攻击，在系统控制器的输入端和温度传感器采集信号端之间我们注入随机的数据或者数据序列。然后我们对其进行两种不同的检测机制：故障检测和基于异常数据的入侵检测机制。从最后的检测结果来看，两种检测都能很好的完成攻击的检测和报警，图3–5和图3–6分别展示了检测结果。

对于第一类的攻击，两种检测机制都采用了相同的算法通过模型的估计输出来判断观测信号是否安全，一旦发现攻击我们会保存当前状态的输入信号所涉及到的所有可能受攻击的传感器。在图3–7中我们需要保存的传感器变量为 $Ta3, s3, s4$ 和 $Tb1, s1, s2, s3, s4, s5$，而且只有分析可能受攻击的变量并调整到合法的范围才能使系统正常运作。图中红色的八边形表示了正在遭受蛮力和类故障攻击的状态节点。

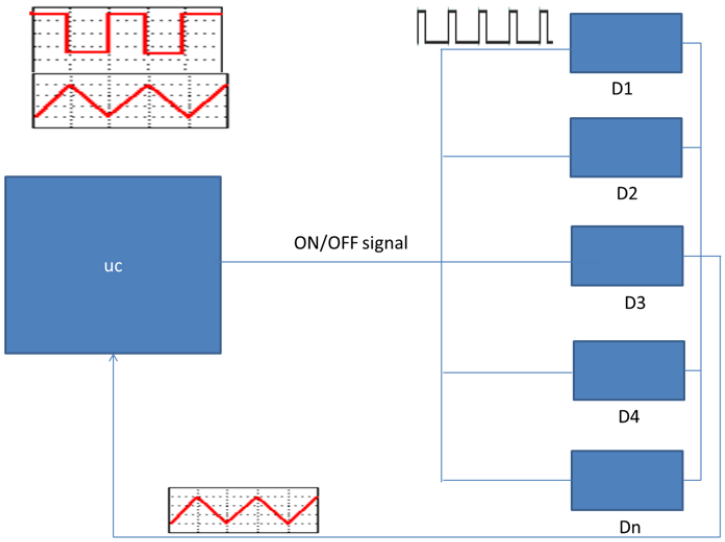


图 3–3 温度控制系统结构图

表 3–1 ds1006 的技术规范

参数	规格
处理器	四核 AMD 皓龙处理器 2.8 GHz，4 x 64 kB L1 数据高速缓存，4 x 64 kB L1 指令高速缓存，4 x 512 kB L2 高速缓存，6 MB L3 高速缓存
定时器	每个内核：3 个通用定时器，多处理器系统的同步时基单元（STBU）
中断控制器	每个内核：一个中断控制器，具有 18 个不同的中断源，连接的 I/O 板的中断源可以通过内部 Gigalink 从每个内核处理。
接口	RS232 接口，标准 UART 允许传输速率高达 115.2 Kbaud（串行）
接口(连接到 I/O 板)	PHS ++ 总线接口，用于模块化 I/O 配置的 32 位 I/O 总线，峰值传输速率为 20 MB / s，新 I/O 板，最多 64 个 PHS 总线中断。
主机接口	一个全尺寸 16 位 ISA 插槽，通过 8 个 16 位 I/O 端口（ISA 总线）接口
多处理器系统	构建具有更多 DS1006 处理器板的多处理器系统，多达 20 个 DS1006 处理器板，每个 DS1006 上通过一个 DS911 千兆模块最多 4 条高速链路，可能的电缆长度可达 100 米
物理特性	340 x 125 x 15 毫米（13.4 x 4.9 x 0.6 英寸）（物理尺寸）
物理特性	0-40°C（32-104°F）（环境温度）
物理特性	主动冷却（风扇）（冷却）
内存	1 GB DDR2-800 SDRAM 本地内存，用于应用程序和动态应用程序数据，4 x 128 MB DDR2-267 SDRAM 全局内存，用于主机数据交换

接下来，我们对控制系统实行隐蔽和避开故障检测的攻击，在系统控制器的输入端和温度传感器采集信号端之间我们注入特定的错误序列向量。例如我们通过从上章节的错误序列构造算法中生

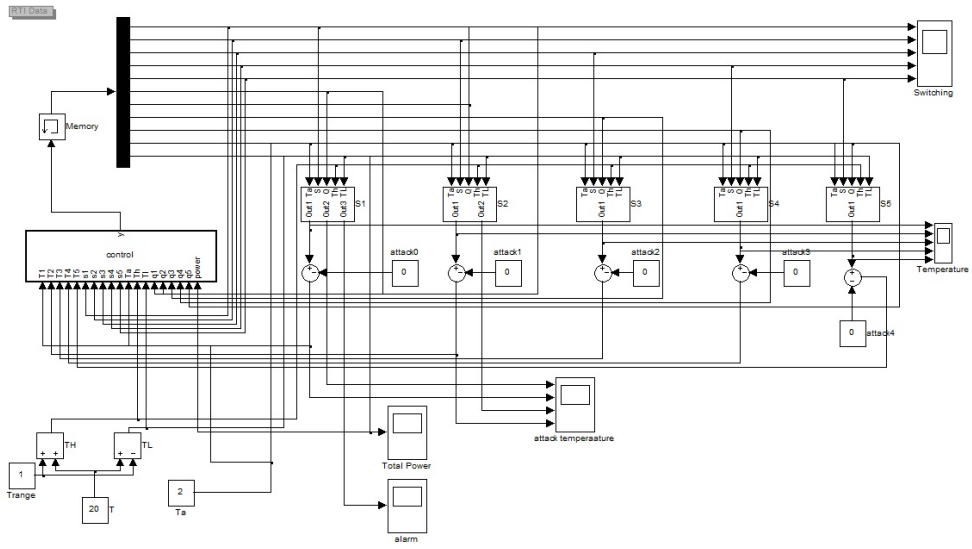


图 3–4 温度控制系统 Simulink 仿真图

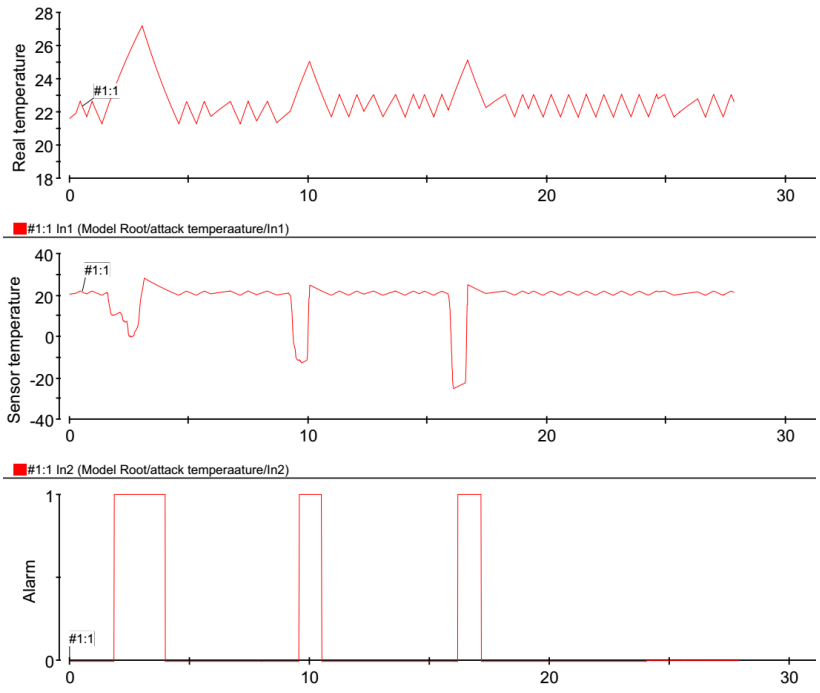


图 3–5 用故障检测机制检测蛮力和类故障攻击

成错误序列 $Q0Q81Q82Q84Q85Q86Q96Q88$，$Q*$ 代表控制器与物理设备交互的二进制输入和输出信号向量，比如 $Q0$ 和 $Q81$ 的含义为

$$Q0 \Leftrightarrow 0101010101000000$$

$$Q81 \Leftrightarrow 0101010101000111$$

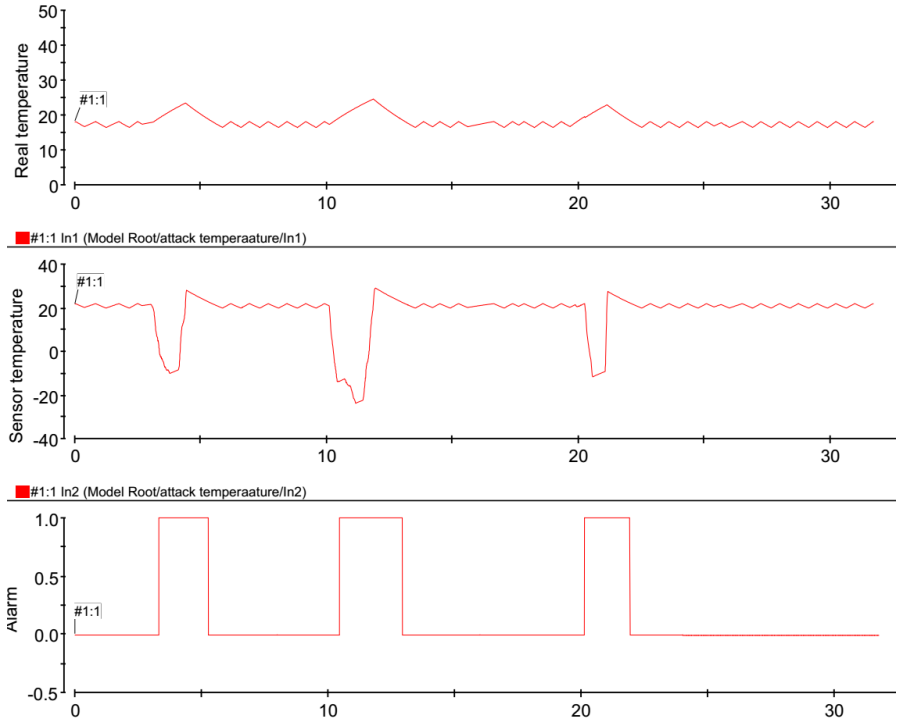


图 3–6 用基于异常数据的入侵检测机制检测蛮力和类故障攻击

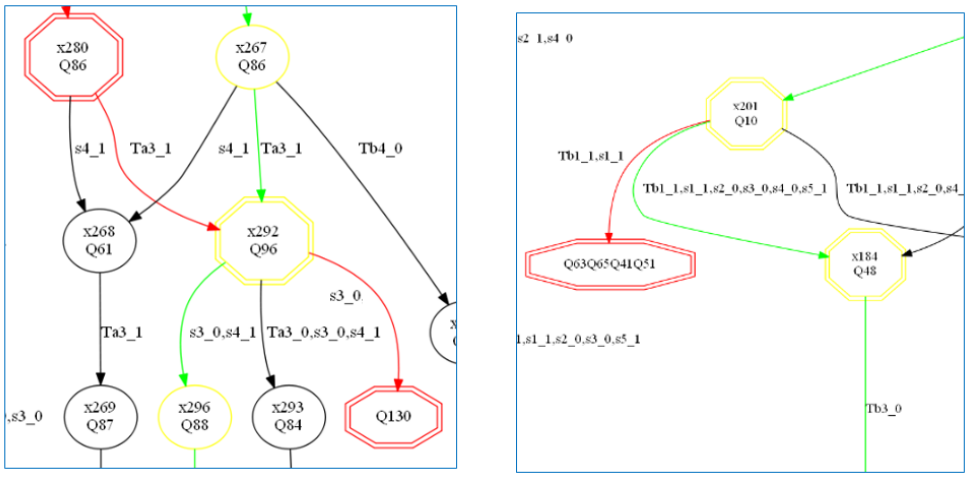


图 3–7 对蛮力和类故障攻击检测的状态转移图示

然后我们也对其进行两种不同的检测机制：故障检测和基于异常数据的入侵检测机制。从最后的检测结果来看，故障检测机制无法检测到攻击但是基于异常数据的入侵检测机制能很好的完成攻击的检测和报警，图3–8和图3–9分别展示了检测结果。

对于第二类的攻击，基于异常数据的入侵检测机制不仅可以通过模型的估计输出来判断观测信号是否安全而且能够通过从观测周期初始向量到当前观测向量组成的观测序列 ϕ 是否存在于原始用

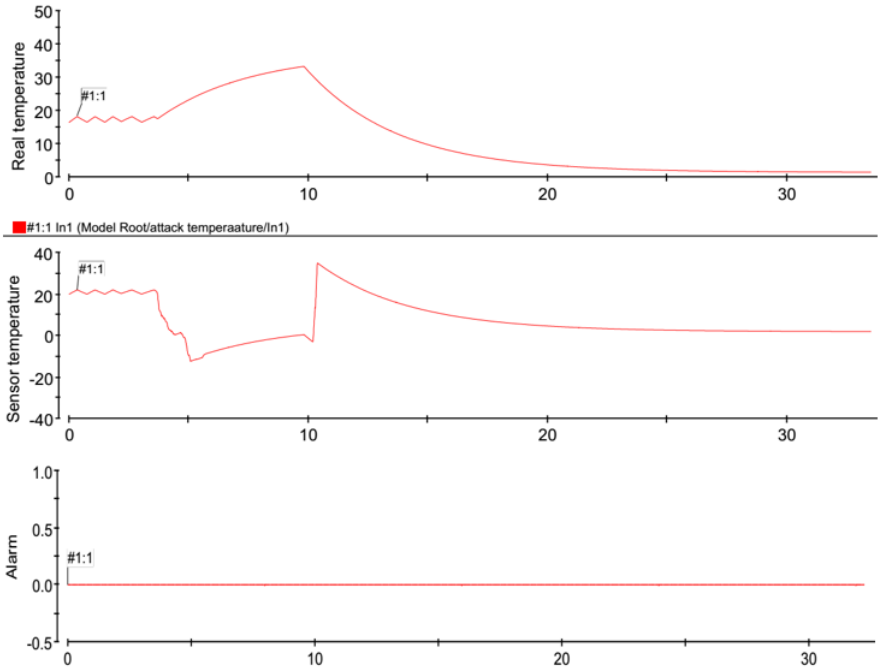


图 3–8 用故障检测机制检测隐蔽和避开故障检测的攻击

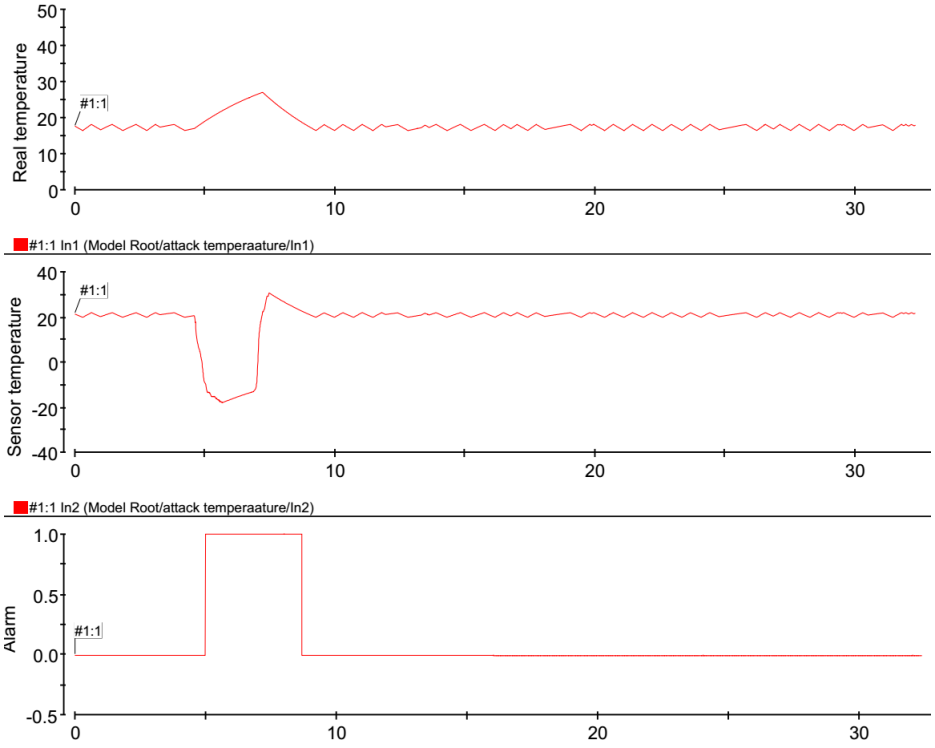


图 3–9 用基于异常数据的入侵检测机制检测隐蔽和避开故障检测的攻击

来辨识模型的采样序列库中来确定是否存在错误序列注入攻击，一旦发现错误序列注入攻击我们同样会保存当前状态的输入信号所涉及到的所有可能受攻击的传感器。在图3–10中我们需要保存的传感器变量为 $Tb3, s1, s2, s3, s5$ 和 $Ta1, Tb5$，而且只有分析可能受攻击的变量并调整到合法的范围才能使系统正常运作。图中红色的八边形表示了正在遭受隐蔽和避开故障检测的攻击攻击的状态节点。

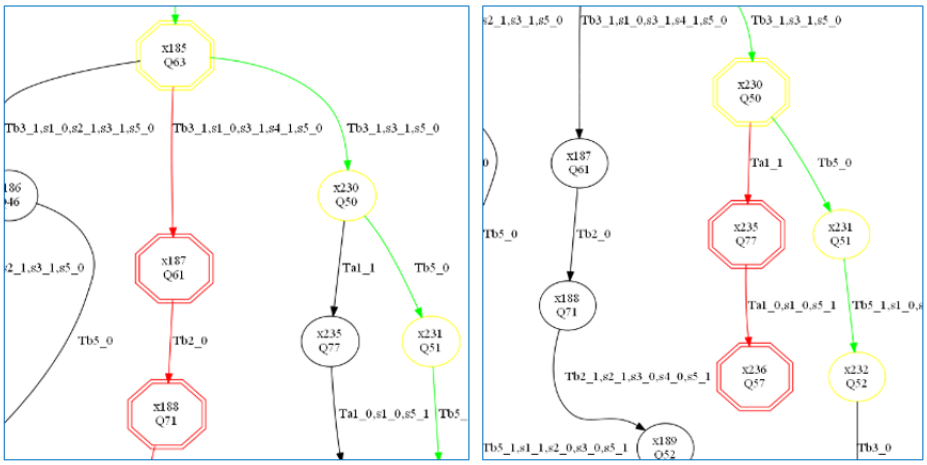


图 3–10 对隐蔽和避开故障检测的攻击检测的状态转移图示

3.5 本章小结

在本章节中，我们提出了基于异常数据的入侵检测和定位防御机制来应对包括传统蛮力和类故障攻击在内的新型隐蔽的错误序列注入攻击从而更好地保护过程控制系统。我们给出了基于异常入侵检测的整个实现过程，包括对错误序列注入攻击的构造并注入系统控制器的输入信号中。然而光对控制器输入和输出信号进行安全监测是不够的，实际工业应用中控制器本身包括控制程序和指令常常会受到类似震网病毒的网络侵入和破坏。这同样会严重影响到我们本章提出的数据检测，因为如果控制器不能正确工作的话，那么辨识模型所需采样的数据序列库也不能足够精确这将直接影响到检测结果。最后基于 Dspace 的实验仿真表明，我们的检测能够对异常数据注入的攻击威胁起到很好的保护作用并证明我们提出的方法的有效性。

第四章 基于规范的入侵检测设计

4.1 引言

在上一章节中我们讨论了如何对控制器与物理设备交互的输入输出信号进行检测保护，防御对象系统主要是故障和异常数据注入攻击，为此我们还专门设计错误序列注入攻击对其进行仿真验证。但是对控制器本身尤其是 PLC 这种可编程控制器，由于它们在整个控制系统中发挥至关重要的作用，近年来正成为对物理设备损坏性攻击的有吸引力的目标。最为典型的 Stuxnet 病毒可以向 PLC 上传恶意代码，以物理损坏他们控制的离心机。更有研究发现，PLC 控制器不仅容易被端口扫描，而且还可以被修改控制系统特定协议和被访问诊断系统。这些易受攻击的互联网控制器被计算机搜索引擎暴露，例如 Shodan。所以对可编程控制器本身控制程序和指令的保护并使其免受恶意代码注入的保护也同样重要。本章我们提出了基于规范的的入侵检测来应对控制器的恶意代码注入攻击，只有经过验证合法的程序和指令才能操作系统或控制服务器上传到指定的可编程控制器设备中。

4.2 入侵检测方案概述

本文我们采用 PLC 作为待检测可编程控制器，检测机制作为外部单独的功能处理器 BITS（Bump-in-the-wire）模式放置在控制系统网络和 PLC 之间。对于任何等待上传到工业设备的 PLC 代码，将被拦截并验证由过程安全工程师定义的一整套安全规范（safety properties），安全特性的示例包括数字设备参数（例如最大驱动速度和加速度）和安全互锁的界限并确保不发生物理上冲突的事件。图4–1展示了基于规范入侵检测机制的基本过程，首先将 PLC 代码（IL code）格式化整理并通过 IL2boolIL 算法转换为中间语言，我们定义为布尔逻辑指令表代码（Boolean IL code），旨在使抽象程序逻辑清晰更具一般性。然后布尔逻辑指令表代码通过模板实例化（Template Instantiate ）过程被迭代地执行转化，将通用模板代码实例化为验证工具 NuSMV 输入程序。前面两步是对 PLC 程序的形式化建模过程，接下来我们给出验证过程。我们通过将得到的形式化代码模型（NuSMV code）输入到验证工具 NuSMV 逐个地检查我们的安全规范。每个布尔规范表示有限状态机中的安全属性是否为真，如果存在任何可达的路径其属性为假，则会给出相应的反例并证明存在安全违规不能而且上传到 PLC 设备中。

4.3 系统建模和入侵检测设计

4.3.1 PLC 指令表程序介绍

指令表程序（Instruction List，IL）是一种较老的 PLC 编程语言，它使用类似于计算机的一些助记符记号来表示 PLC 的操作功能，并根据相应的词法和语法写入每一行程序，以实现工业要求的控制目标和运算处理。

指令表程序具有以下特点：

- 每条指令包含明确的功能而且变量之间的逻辑关系清晰。

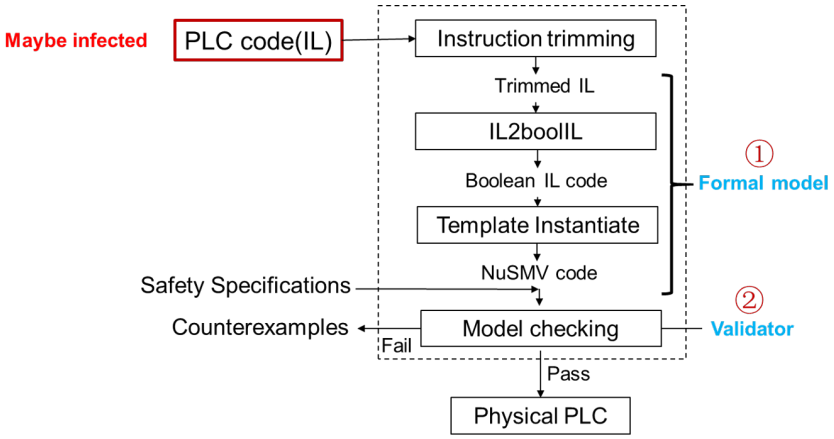


图 4–1 基于规范的入侵检测流程图

- 类似于汇编语言，与其他四种编程语言相比更适合现有的程序验证方法。
- 系统执行效率高。
- 和其他四种编程语言可以直接对应。

因此，为了真正了解 PLC 程序变量之间的逻辑关系，实现 PLC 程序的形式验证，首先我们必须掌握指令表程序，这也是指令表程序成为我们研究对象的重要原因。虽然国际电工委员会在 IEC61131-3 中定义了国际标准的指令表程序，但为了提高系统效率的实际应用，以西门子、洛克威尔和三菱 PLC 为代表的厂商都发展自己的具体指令表程序规范。尽管每个公司对于指令表程序具有不同的命名约定，但是这些指令表程序在逻辑级上的含义是相同的。由于我们更多关注指令表语义的逻辑，PLC 厂商这个因素不会影响我们的检测研究。本文将以西门子的指令程序语句表（Statement List，STL）为对象开始相关研究工作。

在西门子 S7-200 指令系统中，可以分为基本指令和应用指令。所谓的基本指令，最初是指代替传统继电器控制系统所要求的指令。随着 PLC 的功能越来越强大，越来越多的指令被涉及，基本指令的内容也在逐步地扩展。它包括基本逻辑指令、操作指令、数据处理指令、表函数指令和转换指令。基本逻辑指令包括基本位操作指令、逻辑堆栈指令、定时器指令、数器指令和比较指令。基本逻辑指令构成指令集的基本逻辑运算功能，是所有其他指令应用的基础。因此，本文选择基本逻辑指令作为建模对象进行研究工作。表4–1展示了我们涉及到的常用的基本指令。

4.3.2 PLC 程序的形式化建模

定义 1: IL 程序网络（network）是由 3 元组迁移函数表示：

$$\mathcal{T}_{\mathcal{IL}} = (C, c_0, \rightarrow)$$

其中

- C 是 IL 程序配置集
- c_0 是 IL 程序初始配置集
- $\rightarrow$ 是迁移关系

表 4–1 IL 指令表基本指令表

指令	指令类别	指令描述
LD	逻辑指令	装载指令，用于网络块逻辑运算的开始。
LDN	逻辑指令	装载反指令，用于网络块逻辑运算的开始。
A	逻辑指令	与指令，用于单个常开触点的串联连接。
AN	逻辑指令	与反指令，用于单个常闭触点的串联连接。
O	逻辑指令	或指令，用于单个常开触点的并联连接。
ON	逻辑指令	或反指令，用于单个常闭触点的并联连接。
S/R	逻辑指令	置位和复位指令。
OLD	逻辑堆栈指令	或块指令，用于串联电路块的并联连接。
ALD	逻辑堆栈指令	与块指令，用于并联电路块的串联连接。
LPS	逻辑堆栈指令	逻辑入栈指令，也称为分支电路开始指令。从堆栈角度看，LPS 指令的作用是把栈顶值复制后压入堆栈。
LRD	逻辑堆栈指令	逻辑读栈指令，从堆栈角度看，LRD 将第二个堆栈值复制到堆栈顶部。堆栈没有被弹出，但堆栈顶部的旧数值会被破坏。
LPP	逻辑堆栈指令	逻辑出栈指令，也称为分支电路结束指令。从堆栈角度看，LPP 将将堆栈的一个数值弹出堆栈，第二个堆栈数值称为堆栈数值的新顶部。
TON	逻辑定时器指令	接通延时定时器，用于单一时间间隔的定时。
CTU	逻辑计数器指令	增计数器，首次扫描其值为 0，在计数脉冲输入端 CU 的每个上升沿计数器计数一次直到 32767 后停止计数，复位输入端有效值为 0。
JMP/LBL	控制指令	跳转指令和标号指令，使主机可根据不同条件进行判断，选择不同的程序段执行程序

这里每一个 IL 程序配置 $c \in C$ 可以由

$$c = (\sigma, e, E)$$

其中

- σ 是 IL 程序变量状态
- e 是 IL 程序网络中将要执行的变量
- E 是 IL 程序网络中未执行的变量集

因为没有标准的步骤来验证 PLC 程序，我们形式建模的目的是为了通过生成中间语言来更好地或直接转换为 NuSMV 输入程序。PLC 程序的形式模型分为两个阶段。对于第一步，我们采用指令表（IL）作为我们的主要研究的 PLC 语言程序，并且我们需要在算法4–5中将 IL 代码转换为布尔逻辑代码（boolIL），即给出映射算法 $h : IL \rightarrow boolIL$ 使得对于 IL 程序中的每个将要执行的元素 e 都能映射到 boolIL 中的表示 $a = h(e)$。在网络中执行 IL 运算符的顺序由下一个映射决定：$2^{IL} \rightarrow IL$ 根据已经执行的元素集确定接下来执行哪个元素。而在 boolIL 中这一动作已由程序计数器执行并定义映射 $p : boolIL \rightarrow IN$ 将程序中每一条指令 a 映射到它所在的行数 pc。相应的我们给出 boolIL 语言的定义：

定义 2: boolIL 程序网络（network）是也由 3 元组迁移函数表示：

$$\mathcal{T}_{boolIL} = (B, b_0, \rightarrow)$$

其中

- B 是 boolIL 程序配置集
- b_0 是 boolIL 程序初始配置集
- $\rightarrow$ 是迁移关系

这里每一个 IL 程序配置 $b \in B$ 可以由

$$b = (\sigma, a, pc)$$

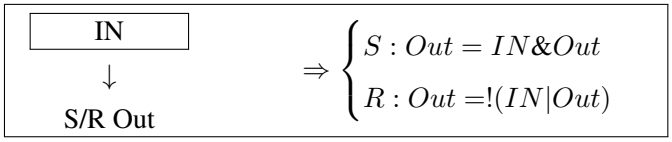
其中

- σ 是 boolIL 程序变量状态
- a 是 boolIL 程序网络中将要执行的语句
- pc 是 boolIL 程序网络中 a 语句所在的程序行号

第一步的主要目的是在程序中从复杂的 IL 代码通过逐行解释上下文和语义获得二进制函数输出。算法4–5首先将完整的程序划分为多个程序块，然后将每个程序块划分为多个网络块部分并初始化所有的集合。其次，我们对每个逻辑和控制指令执行符号解释以获得二进制函数结果。最后，当实现所有 IL 程序时，每个结果将从存储器输出。

从算法4–5我们看出有四类指令程序是无法直接进行解释执行的，因此我们需要对这四类指令进行抽象建模。

1. S/R 指令



算法 4–5 IL2boolIL 算法

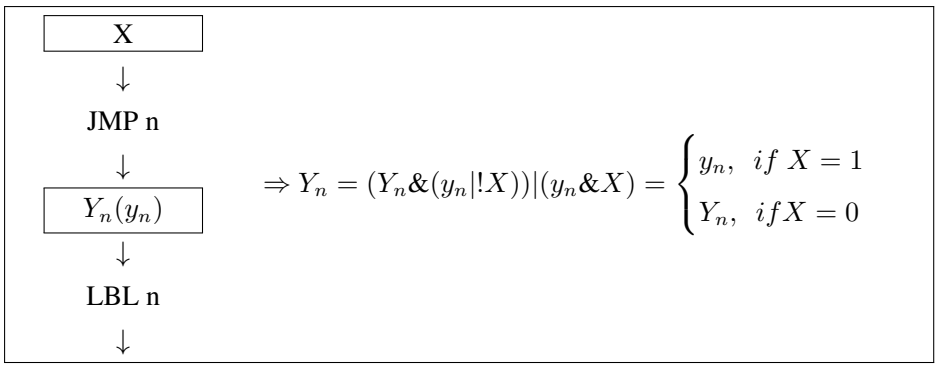
输入： IL 程序;

输出： boolIL 程序 $S_{x_{init}}$;

将完整程序划分成多个程序块 V' 并将每个程序块划分为多个网络块 F
初始化并设置 $V' = 1, F = \emptyset$;
for each $j \in SUM$ **do**
    switch (第 j 个网络块部分)
        case OP 部分:
            for each $i \in SUM$ **do**
                switch (第 i 行程序指令)
                    case $LD(LND)$:
                        设置相应的变量值为初始值（取反）输入并存入内存; break;
                    case $A(AN)$:
                        设置相应的变量值为（取反）输入与 (&) 上前一指令的运算结果并存入内存; break;
                    case $O(ON)$:
                        设置相应的变量值为（取反）输入或 (|) 上前一指令的运算结果并存入内存; break;
                    case $TON/CTU/S/R$:
                        定时器/计数器/置位/复位指令，需要进行抽象建模处理; break;
                end switch
            end for
        case $ALD(OLD)$:
            使前两个指令运算结果执行与 &（或 |）操作并将执行结果入栈; break;
        case $LPS/LPD/LPP$:
            存入/读取/弹出堆栈中前一指令的运算结果; break;
        case JMP/LBL:
            控制指令，需要进行抽象建模处理; break;
        case $=$:
            输出内存中的最终结果; break;
        default :
            Return $S_{x_{init}}$
    end switch
end for
Return $S_{x_{init}}$

我们发现 S/R 有自锁的特性，即当 S 的输出 Out 为 0 时，不管其前面的程序集的输入 IN 为何值，其结果总为 0，相应的 R 指令也有类似的特性。

2. 控制指令 JMP 和 LBL



这里的 X 是 JMP 指令的输入，Y_n 是 LBL 指令的输入，y_n 是 Y_n 的初始值。抽象建模后的 Y_n 的输出严格遵循控制指令的语义，即当 $X=1$ 时，控制指令激活并跳过 JMP 和 LBL 之间的程序，Y_n 的输出即为初始输出值 y_n；当 $X=0$ 也能得到类似的结论。

3. 定时器指令 TON

如图4–2所示，定时器的端口由输入 IN、输出 Q 和定时时间 PT 组成，右上方的状态转移图描述了定时器执行过程中的三个状态迁移过程。初始状态 NR 输出 Q 为 $false$，输入 IN 未激活置为 0；当 IN 激活后但未达到时间要求时，迁移到第二状态 R，此时输出 Q 仍为 $false$；如果在未达到定时时间时，输入被置 0 则又会回到初始状态 NR，一旦在激活状态且达到定时时间，定时器会到达它的第三状态 TO，此时输出 Q 为 $true$，如果输入又被置 0 则又会回到初始状态 NR。这里要特别说明我们并没有模拟具体的时间，对于是否满足时间要求我们只给出 $true$ 和 $false$ 两个状态。为了便于理解，图的下方给出了简单的例子，定时器 $T1$ 的运算结果由输入 $In1$ 和定时时间状态 $T1Ton$ 共同决定。

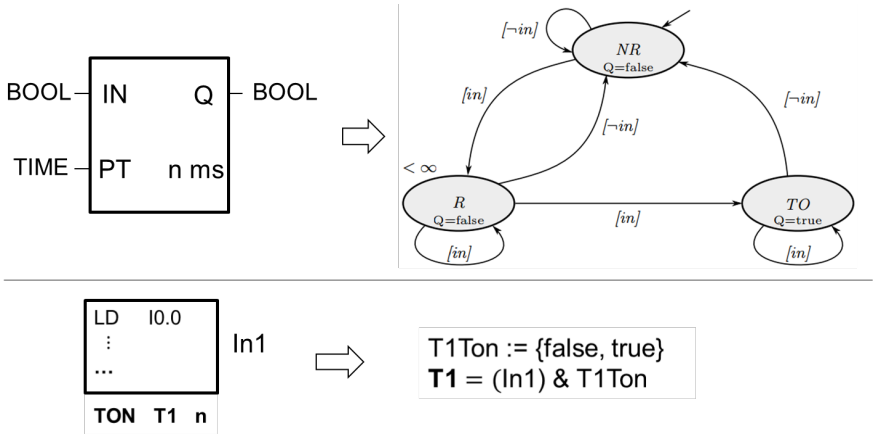


图 4–2 TON 指令的抽象建模

4. 计数器指令 CTU

如图4–3左上方所示，计数器的端口由输入 CU、重置 R、计数次数 PV 和输出端 Q（单独的端口）组成，右上方的状态转移图描述了定时器执行过程中的四个状态迁移过程。初始状态输出 Q 为 $false$，输入 CU 未激活置为 0 且重置 $R = 0$；当 IN 激活后且重置 $R = 0$ 但未达到计数次数要求时，迁移到第二种状态即激活状态，此时输出 Q 仍为 $false$；如果在未达到计数次数时，输入被置 0 则又会回到初始状态，一旦在激活状态时未被重置且达到计数次数 N，计数器会到达它的第三种状态即响应状态，此时输出 Q 为 $true$，如果重置 $R = 0$ 则会到达第四种状态即重置状态，此时 $Q = false$ 且 $PV = 0$ 被重置。为了便于理解，图的下方给出了简单的例子，计数器 $C20$ 的运算结果由输入 $C20CU$ 和计数次数状态 PV 共同决定。

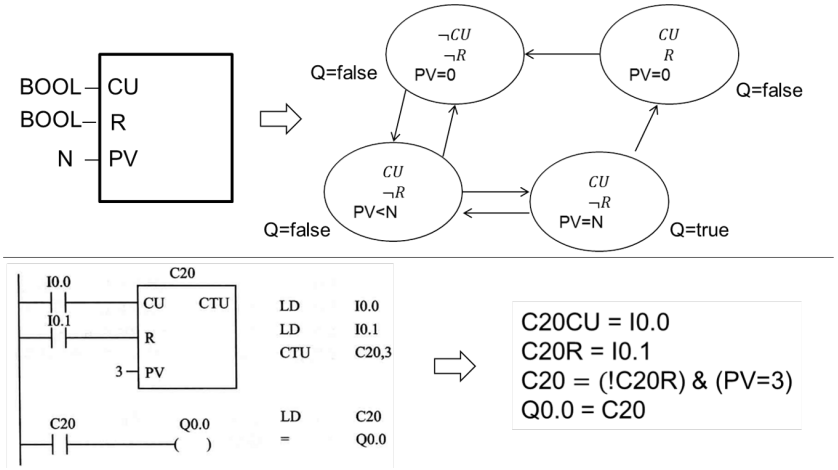


图 4–3 CTU 指令的抽象建模

对于步骤二，我们需要通过步骤一的二进制输入函数的输出来实例化验证工具程序模板得到可执行程序模型。在我们的工作中，我们采用 NuSMV 作为我们的模型验证器。NuSMV 是 SMV 符号模型检查器的重新实现和扩展并用于有限状态系统的形式化验证，它是基于二元决策图（BDD）的第一个模型检查工具。该工具被设计为用于模型检查的开放式架构。它的目的是可靠地验证工业大小的设计，用作其他验证工具的后端和作为形式验证技术的研究工具。它支持对计算树逻辑（CTL）和线性时间逻辑（LTL）进行规范验证，而且我们可以从我们的控制系统的安全规范中轻松获得 CTL 和 LTL。详细的实例化过程如图4–4所示。图4–4首先实例化所有输入，输出和 pc 计数。考虑初始化输入和输出是自动生成过程，因此主要任务是在执行下一个输出语句时为每个 case 语句分配对应 pc 计数器的执行语句。

4.3.3 模型检测

由于 PLC 在每个执行周期使用状态变量来描述状态的变化。因此，我们仅分析单个执行周期来检查所有时间内的安全性规范是远远不够的。在本节中，我们将会给出了完整的模型检测算法来验证 PLC 程序的形式模型树图是否存在违反安全规范的路径。上一节提出的转换算法是对在所有执行周期里发生的状态变换进行建模并构建形式化模型的执行图，本节中我们采用 CTL 规范对模型进行验证并检测攻击，在更详细地阐述模型检测过程之前，我们简要介绍下用于安全规范的 CTL。

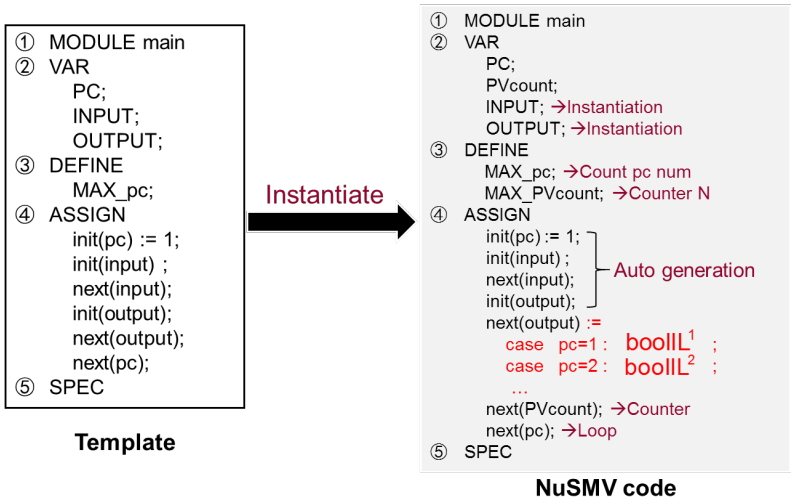


图 4–4 模板实例化过程

4.3.3.1 计算树逻辑 CTL

计算树逻辑，或简称 CTL，是一个分支时间逻辑，意味着它的时间模型是一个树状结构并且未来状态没有确定; 在未来有不同的路径，任何一个都可能是实现的路径 [54]。

定义 3 我们通过 Backus Naur 形式定义 CTL 公式：

$$\begin{aligned}\phi ::= \bot|\top|p|(\neg\phi)|(\phi\wedge\phi)|(\phi\vee\phi)|(\phi\rightarrow\phi)|AX\phi| \\ EX\phi|AF\phi|EF\phi|AG\phi|EG\phi|A[\phi U\phi]|E[\phi U\phi]\end{aligned} \tag{4–1}$$

这里 p 是一组原子公式.

注意到每个 CTL 时间连接是一对符号，第一对是 A 和 E 之一。A 意思是“沿所有路径”（不可避免），E 意思是“至少（存在）一个路径”（可能）。第二对是 X、F、G 或 U，分别表示“下一个状态”、“某些未来状态”、“所有未来状态（全局）”和“直到某个状态”。例如，E[φ_1U φ_2] 中的符号对是 EU。在 CTL 中，诸如 EU 的符号对是不可分割的。注意 AU 和 EU 是二进制的。符号 X，F，G 和 U 不能出现在 A 或 E 之前; 类似地，每个 A 或 E 必须具有 X，F，G 和 U 中的一个。

4.3.3.2 规范检测

我们定义的过程安全规范是安全描述和控制系统与物理设备的约束行为。基于规范的入侵检测尝试对控制逻辑模型 M 验证所有安全规范。如果发现反例，则给出可能存在恶意攻击注入智能控制器程序警报。

规范是由 *specifications* 的有序列表组成且名称为 *id* 的规范具有以下语法：

$$\begin{aligned}id :< input[input-list] >< output[output-list] > \\ < INIT[init-input-list] >< UNIQUE > \varphi\end{aligned}$$

例如，我们可以给出设备按钮的简单描述：“按下启动按钮时，原料 M 的阀门打开”，对应的规

范如下面的 *lbutton* 所示:

$$lbutton\ :input\ launch^*\ output\ v_M\ INIT\ launch^*$$

$$launch^* \Rightarrow AXv_M$$

安全属性ϕ 在计算树逻辑（CTL）中的定义是规范唯一强制性的部分。CTL 的规范定义的 $input$ 和 $output$ 来自控制程序的形式化模型集 IO_ϕ，即 $\{input-list\} \cup \{output-list\} \subseteq\ IO_\phi$。验证过程分为三个步骤：基于规范的入侵检测将会通过控制逻辑模型 M 验证ϕ 中的所有安全规范。

1. 选择 ϕ: $\phi \leftarrow Pop(SPECs)$ 从安全规范集中 $SPECs$ 取出一个属性 ϕ。
2. 将形式化模型和物理设备变量的输入 $input$ 和输出 $output$ 映射集 IO_M 应用到 ϕ 中. 定义为 ϕ/IO_M, 即“在映射集 IO_M 下得到的属性 ϕ”。
3. 验证 $M|=\phi/IO_M$。

这三个步骤适用于给定控制逻辑模型的所有安全规范，如算法4–6所示。首先，我们需要初始化输入和输出映射集 IO_M 和控制逻辑模型 M，然后我们需要遍历安全规范 $SPECs$ 中的每个属性，并转换为 IO_M 映射下的模型符号 CTL 规范。之后将通过 MuSMV 验证工具处理每个模型符号 CTL 规范 $M|=\phi/IO_M$。如果存在不能通过模型检查并生成反例的任何属性，那么基于规范的入侵检测将激活警报机制，需要工作人员检查可能存在的某些注入当前系统程序的恶意代码。

算法 4–6 模型检测

输入: 安全 $SPECs$, 输入和输出映射集 IO_M 和控制逻辑模型 M

输出: 如果检测到存在攻击，则返回 μ_{CE} 并激活报警机制; 否则返回安全通过;

```
1: 初始化 IO_M 和 M
2: for each φ ∈ SPECs do
3:     φ ← IO_M
4:     if NOT M|= φ/IO_M then
5:         μ_0 ← Counterexample Path
6:         μ_CE ← μ_CE ∪ μ_0
7:     end if
8: end for
9: if μ_CE.size() > 0 then
10:    激活 alarm
11: end if
12: return μ_CE
```

4.4 实验仿真

为了证明所提出的基于规范的检测方法的可行性和性能，我们给出了基于自动重合闸控制系统的仿真实验。该系统的功能是在瞬时故障发生时使用重合闸恢复电源，并在发生永久故障时使用备

用电源。该系统有 9 个输入（来自系统的测量）和 11 个输出（从 PLC 到执行器的信号）。图4–5显示了主电气连接图，图4–6显示了系统的输入和输出映射表。

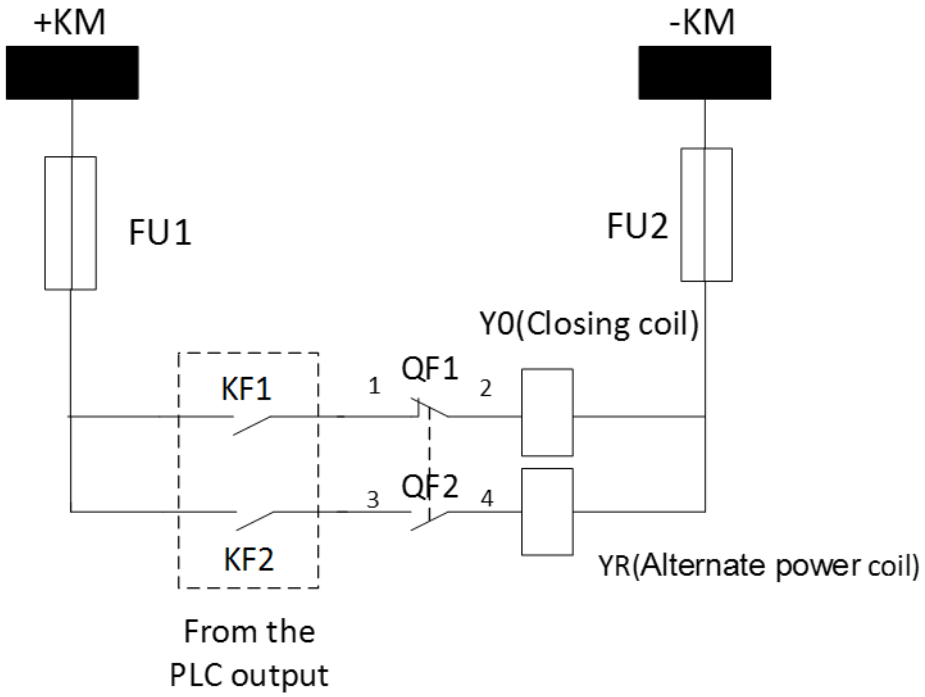


图 4–5 自动重合闸控制系统主电气连接图

首先，我们构建控制系统的形式模型，图4–7显示了如何使用算法4–5将部分指令表程序去重处理并获得二进制函数输出进而转化为布尔逻辑指令表 boolIL 程序。图4–8提供了 NuSMV 工具的实例化可执行代码模型的一部分，并且该模型应该写入以“.smv”结尾的文件。

为了使自动重合闸系统工作在预期的状态，我们为系统设计了 7 条主要的安全规范：

- 关闭指示器在手动关闭回路时工作正常。
- 手动打开回路时，跳闸指示灯工作正常。
- 当模拟瞬态故障时，自动重合闸继电器工作且重合闸成功指示灯闪烁。
- 模拟永久故障时，自动重合闸继电器不工作且重合闸故障指示灯闪烁。
- 当模拟永久故障时，需要自动交流电源为用户提供电源，并且正在使用的备用电源指示灯可以顺利工作。
- 它可以手动关闭备用回路。
- 它可以手动打开交替回路。

在我们获得自动重合闸控制系统的程序模型后，我们需要构造 CTL 形式的规范。例如将第一个

Input		Output	
I0.0	Manual closing	Q0.0	Closing relay
I0.1	Manually opening	Q0.1	Trip relay
I0.2	Transient fault	Q0.3	Reclosing relay
I0.3	Manual reset	Q0.4	Fault trip relay
I0.4	Permanent fault	Q1.0	Closing indicator
I0.6	Reset counter	Q1.1	Opening indicator
I0.7	General reset	Q1.2	Reclosing success indicator
I1.0	Alternate Closing	Q1.3	Reclosing failure indicator
I1.1	Alternate Opening	Q1.4	Alternate close relay
		Q1.5	Alternate trip relay
		Q1.6	Backup power-indicator

图 4–6 自动重合闸控制系统的输入和输出映射表

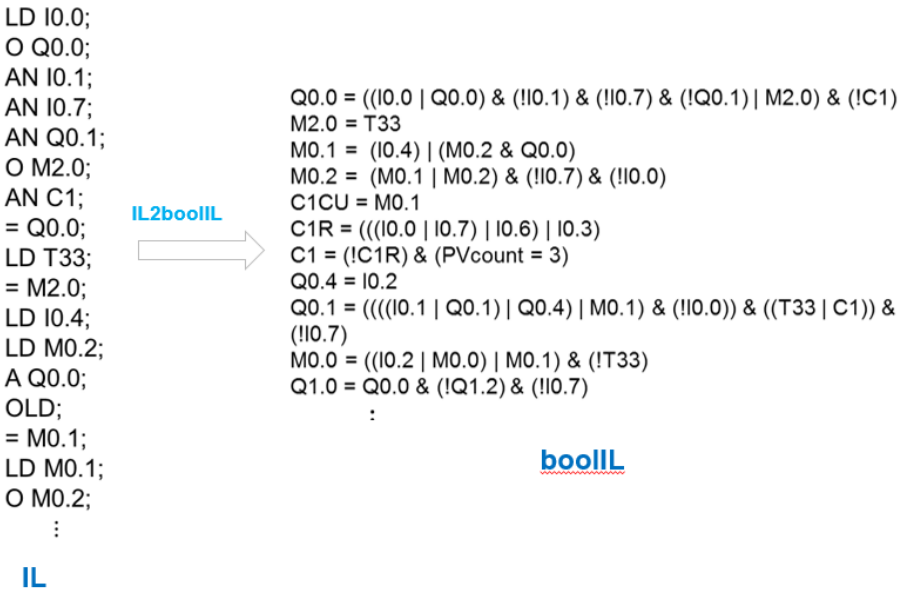


图 4–7 IL2boolIL 转化过程

规范要求用形式化 CTL 规范可以得到：

- $AG(I\_ManualClosing\&(!I\_ManualOpening))\&(!I\_GenReset) -> AF(O\_ClosingRelay));$
- $AG((O\_ClosingRelay\&(!I\_GenReset)) -> AF(O\_ClosingIndicator));$

```
① MODULE main
② VAR
      pc : 1..19;
      PVcount : 1..3;
      I0.0 : boolean;
      I0.1 : boolean;
      M0.0 : boolean;
      I0.2 : boolean;
      M0.1 : boolean;
      Q0.0 : boolean;
      Q0.1 : boolean;
      ...
③ DEFINE
      MAX_pc := 19;
      MAX_PV := 3;
ASSIGN
      init(pc) := 1;
      init(I0.0) :=FALSE;
      init(I0.1) :=FALSE;
      init(M0.0) :=FALSE;
      init(I0.2) :=FALSE;
      init(M0.1) :=FALSE;

      next(I0.0) :={FALSE,TRUE};
      next(I0.1) :={FALSE,TRUE};
      next(M0.0) :={FALSE,TRUE};
      next(I0.2) :={FALSE,TRUE};
```

```
init(Q0.0) :=FALSE;
init(Q0.1) :=FALSE;
 ...
next(Q0.0) :=
        case
              pc = 1 : ((I0.0 | Q0.0) & (!I0.1) & (!I0.7)
              & (!Q0.1) | M2.0) & (!C1);
              TRUE : Q0.0;
              esac;
next(M2.0) :=
        case
              pc = 2 : T33;
              TRUE : M2.0;
              esac;

   ⋮
next(pc) :=
        case
              pc+1 <= MAX_pc : pc+1;
              pc = 2 :1;
              TRUE : pc;
              esac;
⑤ SPEC
```

图 4–8 实例化过程

然而得到的规范都是由物理变量构成的，输入和输出映射表可以将物理变量替换成我们需要验证的程序变量。我们将根据映射中的 *input* 和 *output* 关键字之后给出的名称重新定义 CTL 的规范。最终的 CTL 规格如下:

$$\bullet AG(I0.0\&(!I0.1))\&(!I0.7) -> AF(O0.0));$$

$$\bullet AG((O0.0\&(!I0.7)) -> AF(O1.0));$$

类似地，其余的规范要求都可以转换为 CTL 规格。

如果没有对控制程序的任何攻击，在验证所有安全规范之后，每个规范将呈现“T”代表当前是安全状态。检测结果示于表4–2。

如果存在对控制程序的任何攻击，在验证所有安全规范之后，未能通过验证的安全属性将呈现“F”，意味着系统控制程序执行的结果与该安全规范相悖，系统正处于受攻击的不安全状态。检测结果如表4–3所示。一旦面临这种情况，设备的相关攻击部分将立即发出报警并从可编程控制器中下载受感染程序，然后工程师需要根据验证结果检查第三和第六规格以修复控制系统。

表 4–2 未受到攻击时的验证结果

Cycle	1	2	3	4	5	6	7
1	T	T	T	T	T	T	T
2	T	T	T	T	T	T	T
3	T	T	T	T	T	T	T
4	T	T	T	T	T	T	T
:	:	:	:	:	:	:	:

表 4–3 存在攻击下的验证结果

Cycle	1	2	3	4	5	6	7
:	:	:	:	:	:	:	:
112	T	T	F	T	T	F	T
113	T	T	F	T	T	F	T
114	T	T	F	T	T	F	T
:	:	:	:	:	:	:	:

4.5 本章小结

在本章节中，我们提出了在过程控制系统中基于规范的入侵检测机制来应对可编程控制器程序被恶意代码注入攻击从而更好地保护过程控制系统。我们给出了基于规范入侵检测的整个实现过程，包括对可编程控制器程序进行形式化建模。考虑到 PLC 程序的存在定时器和计数器等高级语言没有的指令，我们通过抽象建模给出等价的可执行二进制逻辑函数对其进行解释建模，这极大的增强了我们建模算法的通用性而不仅仅局限于基于运算指令。因为我们的检测对象是控制器本身包括控制程序和指令使其免受类似震网病毒的网络侵入和破坏。这可以确保控制器正确工作并保证上一章提出的基于异常数据检测的准确性和精度。最后基于自动重合闸控制系统的实验仿真表明，我们的检测能够对控制系统面临恶意代码注入的攻击威胁起到很好的保护作用并证明我们提出的方法的有效性。

第五章 总结与展望

5.1 工作总结

PCS 的网络物理安全问题变得越来越严重, 从网络信息安全和物理数据完整性的角度来看，PCS 的安全风险和入侵威胁正在增加。事实上大多数控制应用（例如监控和数据采集（SCADA））都需要保证严格的安全性，因为一旦由任何异常故障或恶意攻击引起的错误甚至恶意行为都会导致物理设备甚至整个系统遭受无法逆转的破害。因此，PCS 网络信息安全和数据安全已经越来越需要全面深入地研究国家基础设施和关键领域。为了检测 PCS 的入侵威胁并建立安全系统和关键的基础设施安全的综合防御系统，本文针对 PCS 网络物理安全的入侵类型，控制器本身及其与物理设备数据漏洞特征，面向 PCS 的攻击和入侵检测方法的研究与设计主要工作有:

1. 基于故障检测机制构造故障序列注入（FSI）攻击。它可以避免现有的系统故障检测机制，感染和篡改 PLC 的输入信号，迫使 PLC 进行误操作，破坏控制系统的关键设备。首先，我们收集在 PLC 控制器和物理设备之间交换的信号或堆栈数据。然后我们使用输入和输出向量数据库来识别类似于故障检测建模方法的无故障离散事件模型。最后，搜索故障机制无法检测到的所有假序列，获得适当长度的恶意序列，对受控传感器收集的输入信号进行攻击。值得注意的是对多个分支状态的检测将成为对这种攻击的有效防御。实验仿真表明我们的方法是对部署故障检测的控制系统造成一定的破坏性威胁并证明我们提出的方法的有效性。
2. 提出了基于异常数据的入侵检测和定位防御机制，能够应对包括传统蛮力和类故障攻击在内的新型隐蔽的错误序列注入攻击从而更好地保护过程控制系统。我们给出了基于异常入侵检测的整个实现过程，包括对错误序列注入攻击的构造并注入系统控制器的输入信号中。针对异常数据注入攻击的特点，分为两个阶段设计异常数据检测算法。第一阶段将监测的控制器输入信号与模型的预期输出残差对比判断是否数据异常；第二阶段对隐蔽的错误序列注入攻击设计额外 FSI 检测算法来避免此类型的攻击；最后对检测存在异常的数据分析判断，定位到具体攻击源并给出相应的应对措施。最后基于 Dspace 的实验仿真表明，我们的检测能够对面临异常数据注入的攻击威胁起到很好的保护作用并证明我们提出的方法的有效性。
3. 基于规范的入侵检测机制，其被设计为保护可编程控制器本身的程序和指令免受恶意代码注入。只有经过验证的程序和指令才能操作系统或控制服务器上传到指定的可编程控制器设备。我们给出了基于规范入侵检测的整个实现过程，包括对可编程控制器程序进行形式化建模。考虑到 PLC 程序的存在定时器和计数器等高级语言没有的指令，我们通过抽象建模给出等价的可执行二进制逻辑函数对其进行解释建模，这极大的增强了我们建模算法的通用性而不仅仅局限于基于运算指令。最后基于自动重合闸控制系统的实验仿真表明，我们的检测能够对控制系统面临恶意代码注入的攻击威胁起到很好的保护作用并证明我们提出的方法的有效性。

5.2 工作展望

本文重点介绍了针对过程控制系统控制器层面的入侵检测系统的研究与设计，但总体而言，控制器的入侵检测研究还处于起步阶段，特别是在国内相对研究较少甚至无人问津。因此，针对底层控制器的入侵检测系统的研究仍需要大量的研究工作。在本文中，以下问题值得进一步研究：

1. 过程控制系统内各层次入侵检 测协作和优化，从而提高入侵检测的整体效率和性能。包括共享入侵信息和检测信息，对建模和验证所需的计算资源优化调度等合作与优化机制。针对这个问题，本文提出了第三章基于异常数据检测和第四章基于安全规范检测两个层面的检测协调为一体，共同对控制器对象可能存在的内外攻击进行防护。然而，由于入侵检测方法的设计集中在现场设备层和过程监控层，不太容易进行深入研究和实现。而且对于不同的攻击对象，想建立一个通用的模型难度较大，所以协作分层设计检测机制是我们下一步重点研究的内容。
2. 在本文中，入侵检测方法对于现场设备层尤其针对可编程控制器设计，仅作为 PLC 控制器作为对象进行设计和测试验证，而没有更广泛的考虑其他现场控制器，比如常用的 DSP 和 RTU。对于这个问题，进一步的工作可以为其他现场控制器设计入侵检测方法，也可以设计用于各种现场控制器更为通用的入侵检测方法。
3. 对于过程控制系统的入侵检测方法，检测精度和实时要求高于传统 IT 系统，但检测精度和实时性往往存在一定的负相关性。实时在线检测势必需要计算机更快的运算速度和时间，这会严重影响到检测准确度和精度。反之更高的检测精度也会拖累在线检测的实时性。所以进一步的研究可以设计具有更高的检测精度，更好的实时入侵检测方法，并且研究在线实时检测效率和检测精度的平衡和优化。

参考文献

[1] K. Stouffer, J. Falco and K. Scarfone. "*Guide to industrial control systems (ICS) security*". *NIST special publication*, **2011**, *800*(82): 16–16.

[2] J. Weiss. *Protecting industrial control systems from electronic threats*. Momentum Press, **2010**.

[3] M. Hentea. "*Improving security for SCADA control systems*". *Interdisciplinary Journal of Information, Knowledge, and Management*, **2008**, *3*(1): 73–86.

[4] P. F. Roberts. "*Zotob, PnP Worms Slam 13 DaimlerChrysler Plants*". **2008**. `http://www.eweek.com`.

[5] B. Krebs. "*Cyber Incident Blamed for Nuclear Power Plant Shutdown*". **2008**. `http://www.washingtonpost.com`.

[6] S. Grad. "*Engineers who hacked into L.A. traffic signal computer, jamming streets, sentenced*". **2008**. `http://latimesblogs.latimes.com`.

[7] N. Leall. "*Lessons from an Insider Attack on SCADA Systems*". **2009**. `http://blogs.cisco.com/security/lessons_from_an_insider_attack_on_scada_systems/`.

[8] K. Zetter. "*Clues Suggest Stuxnet Virus Was Built for Subtle Nuclear Sabotage*". **2010**. `http://www.wired.com/threatlevel/2010/11/stuxnet-clues/`.

[9] J. Leyden. "*Polish Teen Derails Tram after Hacking Train Network*". **2008**. `http://www.theregister.co.uk/2008/01/11/tram_hack/`.

[10] J. Meserve. "*Sources: Staged Cyber Attack Reveals Vulnerability in Power Grid*". **2007**. `http://articles.cnn.com/2007-09-26/us/power.at.risk_1_generator-cyber-attack-electric-infrastructure?_s=PM:US`.

[11] E. Byres and J. Lowe. "*The myths and facts behind cyber security risks for industrial control systems*". **2004**, *116*: 213–218.

[12] L. Pietre-Cambacédes, M. Tritschler and G. N. Ericsson. "*Cybersecurity myths on power control systems: 21 misconceptions and false beliefs*". *IEEE Transactions on Power Delivery*, **2011**, *26*(1): 161–172.

[13] S. McLaughlin, D. Podkuiko and P. McDaniel. "*Energy theft in the advanced metering infrastructure*". In: *International Workshop on Critical Information Infrastructures Security*, **2009**: 176–187.

[14] S. McLaughlin *et al.* "*Multi-vendor penetration testing in the advanced metering infrastructure*". In: *Proceedings of the 26th Annual Computer Security Applications Conference*, **2010**: 107–116.

[15] Y. Liu, P. Ning and M. K. Reiter. "*False data injection attacks against state estimation in electric power grids*". *ACM Transactions on Information and System Security (TISSEC)*, **2011**, *14*(1): 13.

[16] Y. Mo *et al.* "*False data injection attacks against state estimation in wireless sensor networks*". In: *49th IEEE Conference on Decision and Control (CDC)*, **2010**: 5967–5972.

[17] S. Zonouz *et al.* "*SCPSE: Security-oriented cyber-physical state estimation for power grid critical infrastructures*". *IEEE Transactions on Smart Grid*, **2012**, *3*(4): 1790–1799.

[18] L. Jia, R. J. Thomas and L. Tong. "*Impacts of Malicious Data on Real-Time Price of Electricity Market Operations.*" In: *HICSS*, **2012**: 1907–1914.

[19] A. Teixeira *et al.* "*Optimal power flow: Closing the loop over corrupted data*". In: *2012 American Control Conference (ACC)*, **2012**: 3534–3540.

[20] L. Xie, Y. Mo and B. Sinopoli. "*Integrity data attacks in power market operations*". *IEEE Transactions on Smart Grid*, **2011**, *2*(4): 659–666.

[21] J. Meserve. "*Mouse click could plunge city into darkness, experts say*". **2007**. `http://www.cnn.com/`.

[22] D. Kushner. "*The real story of Stuxnet*". **2007**. `http://spectrum.ieee.org/telecom/security/the-real-story-of-stuxnet`.

[23] T. M. Chen and S. Abu-Nimeh. "*Lessons from stuxnet*". *Computer*, **2011**, *44*(4): 91–93.

[24] M. B. Line *et al.* "*Targeted attacks against industrial control systems: Is the power industry prepared?*" In: *Proceedings of the 2nd Workshop on Smart Energy Grid Security*, **2014**: 13–22.

[25] C. Miller and C. Valasek. "*Remote exploitation of an unaltered passenger vehicle*". **2015**. `https://www.defcon.org/html/defcon-23/dc-23-speakers.html#Miller`.

[26] K. Thomas. "*Hackers demo Jeep security hack*". **2015**. `http://www.welivesecurity.com/2015/07/22/hackers-demo-jeep-security-hack/`.

[27] E. J. Byres and P. Eng. "*Cyber security and the pipeline control system*". *Pipeline & Gas J*, **2009**, *236*(2): 58–59.

[28] L. Pietre-Cambacédes, M. Tritschler and G. N. Ericsson. "*Cybersecurity myths on power control systems: 21 misconceptions and false beliefs*". *IEEE Transactions on Power Delivery*, **2011**, *26*(1): 161–172.

[29] J. Weiss. "*Are the NERC CIPS making the grid less reliable*". *Control Global*, **2009**.

[30] D. Beresford. "*Exploiting Siemens Simatic S7 PLCs*". *Black Hat USA*, **2011**.

[31] T. Newman *et al.* "*SCADA & PLC vulnerabilities in correctional facilities*". *Core Security*, **2011**.

[32] S. Blumsack and A. Fernandez. "*Ready or not, here comes the smart grid!*" *Energy*, **2012**, *37*(1): 61–68.

[33] C. S. King. "*The economics of real-time and time-of-use pricing for residential consumers*". *American Energy Institute, Tech. Rep*, **2001**.

[34] S. McLaughlin, D. Podkuiko and P. McDaniel. "*Energy theft in the advanced metering infrastructure*". In: *International Workshop on Critical Information Infrastructures Security*, **2009**: 176–187.

[35] S. McLaughlin *et al.* "*Multi-vendor penetration testing in the advanced metering infrastructure*". In: *Proceedings of the 26th Annual Computer Security Applications Conference*, **2010**: 107–116.

[36] L. of Cryptography and S. S. (CrySyS). "*Duqu: A Stuxnet-like malware found in the wild*". **2011**.

[37] S. E. McLaughlin. "*On Dynamic Malware Payloads Aimed at Programmable Logic Controllers.*" In: *HotSec*, **2011**.

[38] C. Chevillat *et al.* "*Model-based generation of interlocking controller software from control tables*". In: *European Conference on Model Driven Architecture-Foundations and Applications*, **2008**: 349–360.

[39] PROFIBUS. "*IMS research estimates top position for PROFINET*". **2010**. `http://www.profibus.com/newspress/detail-view/article/ims-research-est%20imates-top-position-for-profinet/`.

[40] Y. Mo and B. Sinopoli. "*False data injection attacks in control systems*". In: *Preprints of the 1st workshop on Secure Control Systems*, **2010**: 1–6.

[41] S. McLaughlin and S. Zonouz. "*Controller-aware false data injection against programmable logic controllers*". In: *Smart Grid Communications (SmartGridComm), 2014 IEEE International Conference on*, **2014**: 848–853.

[42] P. Zhonghua *et al.* "*False data injection attacks for output tracking control systems*". In: *Control Conference (CCC), 2015 34th Chinese*, **2015**: 6747–6752.

[43] M. Roth *et al.* "*Fault detection and isolation in manufacturing systems with an identified discrete event model*". *International Journal of Systems Science*, **2012**, *43*(10): 1826–1841.

[44] D. Garcia-Alvarez, M. Fuente and G. Sainz. "*Fault detection and isolation in transient states using principal component analysis*". *Journal of Process Control*, **2012**, *22*(3): 551–563.

[45] S. Klein, L. Litz and J.-J. Lesage. "*Fault detection of discrete event systems using an identification approach*". *IFAC Proceedings Volumes*, **2005**, *38*(1): 92–97.

[46] M. Abadi *et al.* "*Control-flow integrity principles, implementations, and applications*". *ACM Transactions on Information and System Security (TISSEC)*, **2009**, *13*(1): 4.

[47] S. E. McLaughlin *et al.* "*A Trusted Safety Verifier for Process Controller Code.*" In: *NDSS*, **2014**.

[48] S. McLaughlin. "*Cps: Stateful policy enforcement for control system device usage*". In: *Proceedings of the 29th Annual Computer Security Applications Conference*, **2013**: 109–118.

[49] S. Zonouz, J. Rrushi and S. McLaughlin. "*Detecting industrial control malware using automated PLC code analytics*". *IEEE Security & Privacy*, **2014**, *12*(6): 40–47.

[50] H. Sandberg, A. Teixeira and K. H. Johansson. "*On security indices for state estimators in power networks*". In: *First Workshop on Secure Control Systems (SCS), Stockholm, 2010*, **2010**.

[51] Y. Wang *et al.* "*SRID: State Relation Based Intrusion Detection for False Data Injection Attacks in SCADA*". In: *European Symposium on Research in Computer Security*, **2014**: 401–418.

[52] T. Horsley, A. Wright and C. Barrier. "*Prospecting for new questions: integrating geophysics to define anthropological research objectives and inform excavation strategies at monumental sites*". *Archaeological Prospection*, **2014**, *21*(1): 75–86.

[53] L. Liu *et al.* "*Detecting false data injection attacks on power grid by sparse optimization*". *IEEE Transactions on Smart Grid*, **2014**, *5*(2): 612–621.

[54] S. Klein, L. Litz and J.-J. Lesage. "*Fault detection of discrete event systems using an identification approach*". *IFAC Proceedings Volumes*, **2005**, *38*(1): 92–97.

致 谢

时间过得飞快，转眼又迎来了研究生学涯的毕业季，此刻我对学校和实验室老师同学充满许多的感激。

首先，我要感谢导师邬晶老师。研究生三年的学习和生活，邬老师付出了不少努力，给予我很大的关心和帮助，无论是学习还是生活，邬老师的每一句话和每一次例会总是激励着我，鼓励我不断加油。在导师的细心指导下，我不仅知识一直在增长，研究视野也在扩大，顺利完成今天的文章是离不开老师的耐心审阅和教导，总之，没有邬老师细心培养我也不会有如此大的提高。再次表达我的感激之情，毕业不是师徒情的结束，生活才刚刚开始，未来仍然很长，感恩！其次，感谢我的项目团队组长堵益高师兄的关心和支持，他就像哥哥对我一样，还有实验室师姐师兄、同学和朋友，因为你们的陪伴，我的生活才更加精彩。最后，我要感谢我的父母。你们的无私的爱伴随我的成长，你们是我不断努力不断追求内在动力的梦想，谢谢你，我的家人。书写毕业论文是一个重新系统研究的过程，完成论文，也意味着开始一个新的学习生活。在这里，再次对我的老师、同学、朋友和家人说声：谢谢！

攻读学位期间发表的学术论文

[1] Xiao M, Wu J, Long C N, Li S Y. Construction of false sequence attack against PLC based power control system[C]. Control Conference (CCC), 2016 35th Chinese. TCCT, 2016: 10090-10095.

攻读学位期间参与的项目

[1] 自然基金项目“61172064, 61473184”