doi:10.11959/j.issn.1000-436x.2017133

基于仿真建模的工业控制网络入侵检测方法研究

高一为[1]，周睿康[2]，赖英旭[1]，范科峰[2]，姚相振[2]，李琳[2]
（1. 北京工业大学信息学部计算机学院，北京 100124；2. 中国电子技术标准化研究院，北京 100007）

摘 要：目前工业控制网络的入侵检测方法存在协议通用性差、误报率高和无法对未知入侵进行检测等问题。提出一种基于现场总线设备建模的入侵检测方法，利用仿真建模模拟控制器的真实功能，对控制器进行保护；并通过系统辨识建模的方法建立被控对象模型，保证控制器获得的被控对象数据真实准确，从而实现对工业控制网络的入侵检测。经实验验证，所提入侵检测方法检测效果较好。
关键词：仿真；编译原理；系统辨识；入侵检测
中图分类号：TP309 **文献标识码**：A

Research on industrial control system intrusion detection method based on simulation modelling

GAO Yi-wei[1], ZHOU Rui-kang[2], LAI Ying-xu[1], FAN Ke-feng[2], YAO Xiang-zhen[2], LI Lin[2]
(1. College of Computer, Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China;
2. China Electronics Standardization Institute, Beijing 100007, China)

Abstract: At present, intrusion detection system over fieldbus network layer was a basic protection method in industrial control system. However, it has some weakness, such as poor generality, high false-positive rate, and unable to detect unknown anomaly. An industrial control system intrusion detection method based on fieldbus network equipment simulation was proposed. The method prevented control program from being tampered or destroyed based on controller simulation modelling. Controlled object simulation modelling was designed for ensuring that the system input was credible. Thus the intrusion detection of industrial control network was realized. At last, the results indicate that the proposed intrusion detecting method is available.
Key words: simulation, compiler theory, system identification, anomaly detection

1 引言

工业控制系统（ICS, industrial control system）包括监控和数据采集（SCADA）系统、分布式控制系统（DCS）和可编程控制器（PLC）等。以往工业控制系统的封闭性和专用性由于互联网的接入而被彻底打破，加之企业没有健全的安全意识。2010 年“震网”（stuxnet）病毒[1]的爆发，让全球再一次明白，工业控制系统已成为黑客的主要目标[2]，随后“毒区”（duqu）[3]和“火焰”（flame）病毒又相继出现，与“震网”共同形成“网络战”攻击群。2015 年底发生的乌克兰大面积停电事件又一次为工控安全拉响警报。

目前，针对现场总线的入侵检测[4]研究可以分为 3 类：基于统计的入侵检测方法、基于知识的入侵检测方法以及基于机器学习的入侵检测方法。

基于统计的入侵检测方法主要包括系统多变量模型[5]、时间序列模型[6]等方法。Wei[7]和 Barbosa[8]等利用工业控制网络中流量的周期特性进行入侵检测，但是难以应对复杂的攻击行为。Wei 等又提出了采用 ARMA 方法对周期流量建模，通过计算残差序列方差值进行检测。该方法没有对原始数据

收稿日期：2016-11-24；**修回日期**：2017-03-14
基金项目：国家智能制造专项基金资助项目（No.[2015]1170）；青海省自然科学基金资助项目（No. 2017-ZJ-912）
Foundation Items: The National Manufacturing Special Program (No.[2015]1170), The Natural Science Foundation of Qinghai Province（No. 2017-ZJ-912）

中的噪声进行预处理，所以，模型准确率易受噪声影响，实际检测误差较大。Rrushi 等[9]通过比对流量分组中标明的内存修改位置和数值以及实际控制器内存中数据的变化情况进行检测，效果较好。但基于统计的入侵检测方法对于复杂的攻击行为，如篡改控制器的输出或篡改控制逻辑等攻击的检测能力有限。基于知识的入侵检测方法包括基于规则判断的方法[10,11]、状态描述语言方法[12]和有限状态机[13]等。Morris 等[10]针对现场总线网络中的 Modbus RTU/ASCII 协议设计了一种基于 Snort[11] 的入侵检测方法，利用 Snort 规则对上行数据和下行数据分别进行检测。Goldenberg 等[14]发现 Modbus 协议流量除了在时间上存在周期性之外，在数据分组载荷的长度上也存在一定的周期性。如果数据分组不属于任何一个模式，或自动机中出现不可达路径，则认为发生异常。基于知识的入侵检测方法的主要缺点是需要过多的先验知识，无法满足对未知攻击的入侵检测需求。基于机器学习的入侵检测方法主要包括贝叶斯网络方法[15]、模糊逻辑方法[16]、人工神经网络[17]以及遗传算法[18]等，但要求训练集较大，且无法解释判定为异常的原因。

因此，本文针对以上 3 种入侵检测方法中存在的主要问题展开研究，并试图找出能够在不依赖传输协议公开的前提下，对控制器和被控对象进行入侵检测的方案。本文从 2 个层面提出基于仿真模型的工业控制网络入侵检测方法：1) 基于控制器建立仿真模型，在仿真控制器输出的基础上监测真实控制器的控制逻辑和数据是否被篡改；2) 基于系统辨识方法建立被控对象模型，并在被控对象仿真模型的基础上建立相应的入侵检测方法。

2　控制器仿真建模

工业控制系统中的控制器，如可编程逻辑控制器（PLC）等是整个工业控制系统的基础，控制器建模的精度决定了入侵检测算法的精度。控制器的建模方法可以采用统计方法建模，如系统辨识方法；也可采用静态分析方法，如状态空间方程等方法，通过分析变量之间的依赖关系来建立模型。前者受限于统计方法自身存在的特性，建模精度较差。而后者则对程序代码过度依赖，每次变换程序代码都需要重新分析建模。相比以上 2 种方法，本文提出控制器仿真建模方法，利用模拟器编译并执行控制逻辑代码，实现一个同 PLC 功能完全相同的仿真系统，模型输出结果不仅完全同真实 PLC 的运行结果一致，且兼顾了程序代码的执行逻辑和内存中数据的变化，当控制程序发生变动时，不需要了解程序代码的功能细节就可以完成对模型的修正，模型精度高。

2.1　仿真原理

控制器的核心是对控制程序的编译和执行，因此，控制器仿真建模的核心就是模拟控制程序的编译和执行。PLC 中的控制程序通常由结构化控制语言（SCL, structured control language）编写，它是一种类似于 Pascal 的高级语言，适用于 SIMATIC s7-300、s7-400 和 C7 等设备。SCL 语言除了具有输入输出、定时器、计数器和符号表外，还具有其他高级语言的特性，如循环、选择、分支、数组以及高级函数等。

本文设计了一个将 SCL 语言翻译成 C 语言的仿真编译器，省去了在编译过程中代码优化、链接等步骤。仿真模型中编译器的工作流程如下。

步骤 1　词法分析器从程序源代码中以字符串流的形式读取控制程序，根据预先定义的字符串表对字符串流进行识别和分割形成单词。将单词传递给语法分析器。

步骤 2　语法分析器以单词流为输入，分析、识别出控制程序的各个语法成分，并构造出一个抽象语法树（AST, abstract syntax tree）以及用于存储变量的符号表。之后根据语义规则向语法树中添加变量和相应的执行规则，形成中间代码。

步骤 3　执行引擎通过遍历语法树执行规则，计算和修改符号表中的变量值来完成程序的执行。

2.2　词法分析器

在介绍词法分析器之前首先对基本概念进行定义。

2.2.1　定义

定义 1　终结符是语言中用到的基本元素，一般不能再被分解，如英文字母 A~Z 及其他标点符号、空格、回车符和换行符等。通常的程序设计语言以 ASCII 字符集作为终结符。

定义 2　非终结符是由终结符和至少一个非终结符号组成的串。非终结符通常由终结符构成字符串中的子串。

定义 3　产生式也称重写规则，是由 2 个字符串构成的有序对组成，中间用一个箭头分割。箭头两侧的字符串可以由终结符及非终结符组成，形式上符合语法限制。

定义 4 上下文无关文法（CFG, context free grammar）可以用一个四元组进行表示：$G=(\Sigma, N, P, S)$。其中，Σ 是终结符集合，是一个有限、非空集合。N 是非终结符集，也是一个有限、非空集合，且 $N \bigcap \Sigma = \varnothing$。$P$ 为产生式集，$S(S \in N)$ 是开始符号或目标符号。

定义 5 一步推导是指将字符串中的非终结符按照产生式进行替换的一次过程。

以一个例子说明一步推导过程。假设定义的一种 Z 语言由终结符集 $\{a,b,c\}$ 中的 2 个字符组合而成，即 Z 语言的非终结符集为 $\{aa,ab,ac,ba,bb,bc,ca,cb,cc\}$，则 Z 语言的文法可以定义如下

$$G=(\{a,b,c\},\{a,b\},\{a \to ab, a \to bb, a \to cb, b \to a, b \to b, b \to c\}, a\}$$

即当开始符号或目标符号为 a 时，可以用任何一个与 a 相关的产生式重新改写已得到的字符串 a，如用 cb 改写 a，得到字符串 cb，而 cb 中的 b 亦可以改写成为 a、b、c 中的一个，最终得到 Z 语言的全部终结符集，这一过程称为一步推导。

定义 6 记号（token）是指一组字符串所产生的相同标记。如正整数、负整数、浮点数可以统一定义为数字，可以定义记号为 *NUM*。同时，程序代码中的保留字，如 IF、ELSE 等也可以定义相应的保留记号，常见的结构在词法分析的过程中会被定义为记号，如关键字、操作符、标识符、常量、文字串和标点符号。

定义 7 模式（pattern）是指一个记号对应的所有字符串的组合形式，如数字记号 *NUM* 的模式是任何数字常数等。

定义 8 记号属性是指词法分析器为了向语法分析器传递信息，除了对文本信息进行记号分类外还要额外追加的其他信息。如 *NUM* 记号通常还要追加数字的值作为记号属性。

词法分析器的本质是将输入的控制程序代码，按照 SCL 语言的语法格式进行单词分割，将符合语法的字符串传递给语法分析器进行进一步处理。因此，在词法分析器的设计过程中，采用确定的有限状态自动机的形式对文法产生式建立模型，完成单词分割。

2.2.2 基于确定有限状态自动机的单词分割机制

为提高词法分析器的工作效率，本文采用确定有限状态自动机的方法对所有模式的产生式集进行表示，提高了单词分割的效率。

SCL 语言程序代码是由字母 a~z、A~Z、下划线及“#”组成的，以代码标识符的定义正则表达式[a−zA−Z_\#]+为例，可以由上下文无关文法来进行形式化的描述 $G=(\{r,R,"\#","\_","+"\}, \{T,F\}, P, T\}$。产生式集 P 表示为

$$P = T \to Tr"+"B, T \to TR"+"F, F \to F"\_""+"F, F \to F"\#""+"F, F \to T$$

其中，r 表示小写字母表中的任意符号，R 表示大写字母表中的任意符号，“+”表示允许同一终结符重复多次，T={*R*,*r*}表示 SCL 每行程序的起始字母，F={"T" "W"}表示 SCL 语言中以字母 T 和 W 开头，包含下划线及“#”。从文法定义上可以看出终结符集 $\zeta=(\{r,R,"\#","\_","+"\}$ 中 5 个终结符的不同优先级，r 和 R 的优先级高于其他 3 个终结符的优先级。

而对于以上的产生式，可以由一个确定有限状态自动机的形式表示，每一次产生式的一步推导可以视为一次有限状态自动机的状态转移过程。一个确定有限自动机由一个五元组 $M=(\Sigma', Q, \Delta, q_0, F)$ 组成。其中，Σ' 是输入字母表，Q 是有限的状态集合，Δ 是状态转移函数，$q_0 \in Q$ 是开始状态，$F \subseteq Q$ 是终止状态集合。标识符的确定有限状态自动机如图 1 所示。

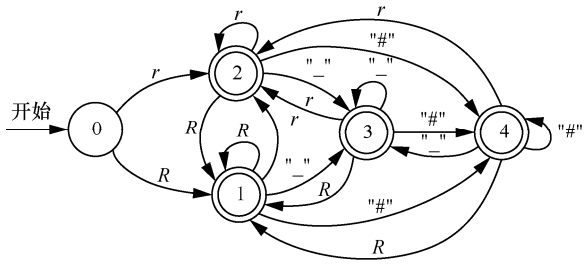


图 1 标识符的确定有限状态自动机

当对标识符字符串进行分析时，遇到空格或其他语法终结符以外的字符时完成对标识符的分析，并返回标识符模式的记号（变量记号为“T_VAR”）给语法分析器。

2.3 语法分析器

每一种程序设计语言都具有描述程序语法结构的规则。如 Pascal 语言开发的程序是由程序块组成，程序块由语句组成，语句由表达式组成，表达式由记号组成。如 2.1 节所述，SCL 语言是一种类似于 Pascal 的高级语言，其语法结构同 Pascal 语言几乎一致。而语法分析器的作用就是利用形式化的

语言创建能够描述 SCL 语言的语法树，并且利用语法树和词法分析器反馈的记号以及记号属性创建符号表及中间程序代码。

目前，大部分程序设计语言的语法规则都可以用上下文无关文法来描述。抽象语法树是一种描述上下文无关文法的推导工具，对于程序代码而言，抽象语法树上的每一个节点都代表源代码中的一种结构。以一段 SCL 代码为例。

例 1　对程序代码 rCycle:= DINT_TO_REAL (TIME_TO_DINT(CYCLE)) / 1 000.0 ；建立抽象语法树，其结果如图 2 所示。根据定义 8，例 1 中产生式集可以表示为 P = { 操作符→表达式":="表达式，操作符→表达式"/"表达式，表达式→操作符，表达式→关键字，表达式→标识符，表达式→常量，关键字→"DINT_TO_REAL"表达式，关键字→"TIME_TO_DINT"表达式}。虽然例 1 只是一段简单代码，但是根据其生成的产生式集可以推导出 SCL 语言包含赋值、除法、函数运算的一般表达式的所有组合形式。本文用 BNF 范式对 SCL 语言的全部语法树进行了描述。以例 1 形成的产生式集 P 为例，expression 表示表达式、number 表示常量、keyword 表示关键字、identifier 表示标识符、operator 表示操作符，那么产生式集用 BNF 范式的形式表示如下。

1) expression:
2)　　operator
3)　|　identifier
4)　|　keyword
5)　|　number
6);
7)operator:
8)　expression “:= ” expression
9)　|　expression “/” expression
10);
11)keyword:
12) “DINT_TO_REAL” expression
13)　|　“ TIME_TO_DINT” expression
14) ;

在获得了 BNF 范式描述的语法树后，需要一套分析方法将词法分析器反馈的记号和记号属性按照 BNF 范式描述的规则对应到语法树中，以便生成符号表和中间程序代码，如图 3 所示。

LALR(1)[19]分析法是自底向上分析法中最重要的分析法之一，其分析能力介于 SLR(1)和 LR(1)之间，且分析表的规模远小于 LR(1)分析器，因此，LALR(1)更适宜创建语法分析器。本文根据预先编写的 BNF 范式语法树描述自动生成 LALR(1)语法分析器。

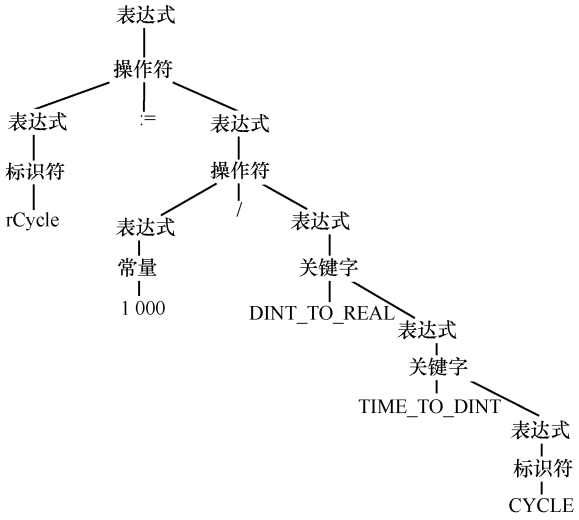


图 2　例 1 程序建立的语法树

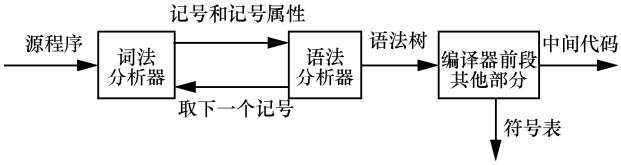


图 3　语法分析器在编译器中的位置

2.4　符号表及中间代码

语法分析器分析产生语法树的同时，会根据词法分析器传送来的标识符记号创建符号表。符号表中每个存储变量的存储结构体由 5 个部分构成：变量名称、存储变量值的联合体、标明变量类型的标识位、标识变量引用次数的计数器以及一个脏数据标识位。如果该变量在当前代码段执行结束时引用次数为 0，则会由资源回收机制释放该变量所占的存储空间；如果计数器小于 0 则表示当前变量值为脏数据；存储变量节点的数据结构如下。

1) Structofzval_struct{
2)　String Name;
3)　　Union Zval_value;
4) Int Refcount;
5) Int type;
6)　　Bool Is_dirty
7) }

本文提出的语法分析器在将函数声明语句的语法树转变为符号表的同时，还要将其他表达式根据语法树生成相应的中间代码。首先需要说明本文生成的中间代码的存储结构是一串一维的单向链表（ZNODE_ARRAY）。当语法分析器根据词法分析器反馈的标记创建语法树时会创建相应的中间代码节点（ZNODE），并将该 ZNODE 追加到链表尾部，当执行引擎执行中间代码时会从上到下遍历单向链表。

同上下文无关文法中产生式的思路一样，ZNODE 可以理解为一个终结符，有产生式同它对应，使 ZNODE 可以组成描述表达式的所有语法。该产生式可以表示为 $P=\{$ ZNODE → ZNODE_ARRAY，ZNODE → ZVAL_STRUCT，ZNODE → VMPLC_OP$\}$，其中，VMPLC_OP 是二元运算结构体，是语法树的内节点。VMPLC_OP 中包含运算节点 ZNODE_1、运算节点 ZNODE_2、运算方法指针 HANDLER 以及运算结果节点 ZNODE_ RESULT。ZNODE_ARRAY 的结构如图 4 所示。

以 ZNODE_ARRAY 形式存储的是各个程序块以及程序块中的程序代码，是不含运算符的上下文关系，而 VMPLC_OP 中存储的是含有运算符或关键词的表达式，如例 1 所示。这段程序代码表达式生成的中间代码如图 5 所示。ZVAL_STRUCT 是存储变量值的结构体，包含的变量节点是符号表中变量节点的指针地址，因此，只在形成中间代码时需要完成一次符号表的遍历工作。一旦中间代码生成完毕，执行引擎执行中间代码的过程中是不需要再次查询符号表的，其中，VMPLC_EQUAL、VMPLC_DIV_VARIABLE 分别是执行逻辑等和逻辑除法操作的函数句柄。

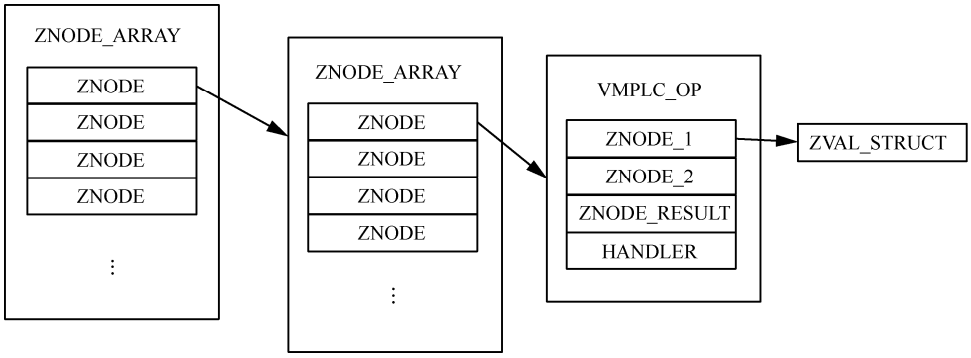


图 4 中间代码的存储结构

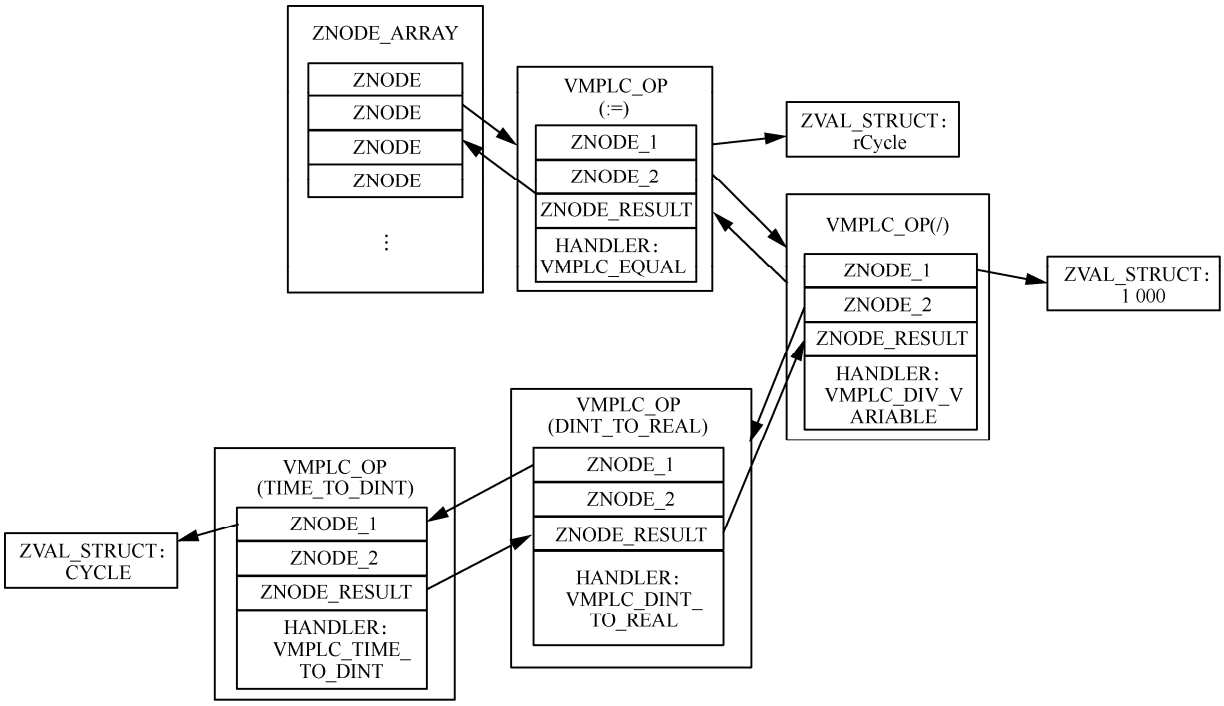


图 5 例 1 表达式生成的中间代码

2.5　执行引擎

按本文方法构建的词法分析器和语法分析器完成符号表建立及中间程序代码生成后，编译器程序已经可以在不依赖控制逻辑程序 SCL 代码的情况下，直接通过执行中间代码的方式实现控制逻辑程序的功能。

执行引擎执行中间代码时，首先创建一个指向根 ZNODE_ARRAY 结构体的指针，自上向下地遍历 ZNODE_ARRAY 结构体。执行步骤如下。

步骤 1　读取指针指向的 ZNODE 结构体，判断结构体中存储的数据类型。

步骤 2　如果ZNODE结构体中存储的是ZNODE_ARRAY，则将当前的根 ZNODE_ARRAY 结构体和指针压入堆栈中缓存，将当前的执行环境变量替换为目标 ZNODE_ARRAY 结构体，替换的环境变量包括创建新的缓存堆栈，创建新指针指向目标 ZNODE_ARRAY 顶端。

步骤 3　如果指针指向的是VMPLC_OP结构体，则判断 ZNODE_1 结构体及 ZNODE_2 结构体是否为空，如果为空，则判断是否符合 HANDLER 指针指向的最小输入参数要求，如果不符合，则报出语法错误。

步骤 4　读取 ZNODE_1 和 ZNODE_2 结构体中的数据。如果结构体中存储的是ZNODE_ARRAY或 VMPLC_OP，则重复步骤 2 或步骤 3。将步骤 2 或步骤 3 的执行结果返回，如果存储的是 ZVAL_STRUCT 变量节点，则直接读取变量值。

步骤 5　调用 HANDLER 指向的方法，并将 ZNODE_1 和 ZNODE_2 的数据作为输入传入到 HANDLER 方法中，如果 HANDLER 方法的输入超过 2 个，则直接将变量以 ZNODE_ARRAY 的形式传入。

步骤 6　将步骤 5 的执行结果以指针的形式存储到 ZNODE_RESULT 中，并将 ZNODE_RESULT 返回给上级 ZNODE 节点。如果存在赋值操作，则会在调用 HANDLER 方法时直接修改 ZVAL_STRUCT 中的变量值。

步骤 7　若当前 ZNODE_ARRAY 遍历结束，则从堆栈中弹出上一次压入的环境变量并恢复。若遍历未结束，则将指针下移指向下一个 ZNODE 节点。重复步骤 1~步骤 7 直至遍历完成。

在执行引擎执行中间代码的过程中，符号表会根据执行结果实时修改变量的值。

2.6　时钟同步

仿真系统的扫描周期同真实控制器的扫描周期时长是一致的，理论上可以做到仿真模型和控制器同时输出控制指令。但是仿真系统在 PC 上运行，计时器采用软时钟计时，而控制器采用的是硬件时钟，因此，仿真的时长并不精确，可能早于或晚于控制器结束扫描周期。由此会造成仿真系统执行结束，但控制器尚未将控制指令输出，仿真系统获取的输出还是上一个扫描周期结束的情况，造成入侵检测模块误报。

为了解决这一问题，本文采用加快仿真模型执行速率的方法，将仿真模型的扫描周期控制在控制器扫描周期的 0.5~1.0 倍之间，保证仿真模型可以快于控制器计算出控制指令，这样不会漏掉每一个控制器发出的控制指令。

同时，采用异步缓存机制解决仿真输出不同步存储问题。异步缓存机制会在发现仿真模型输出和控制器输出不一致时，将仿真模型输出值进行缓存。在第 2 个扫描周期设置一个时长为半扫描周期的定时器，再次获取缓存的仿真模型输出与真实控制器输出进行比对。

3　基于系统辨识方法的被控对象建模

被控对象建模是在控制器仿真基础之上提出的，被控对象建模的重点是验证控制器获得的反馈输入值是否符合预期。本文的被控对象建模方法分为 2 个部分，一部分是数据预处理，另一部分是系统辨识建模。系统辨识建模又包括模型定阶、模型选择以及模型验证。

3.1　数据预处理

以工业控制系统中直接测量的数据作为辨识模型的输入值，通常会有直流分量以及一定的高频噪声，影响建模准确性，因此，需要对原始的输入输出测量数据进行预处理。

首先需要从测量值中去除高频噪声，本文采用分段平滑的处理方法[20]，如式(1)所示，其中，$x(i)$ 为输入的测量值，m 为滑动窗口大小。

$$x(i)=\frac{\sum_{j=1}^{j=m}x(i-j)+\sum_{j=1}^{j=m}x(i+j)}{2m+1} \tag{1}$$

测量数据中直流分量可能是一个固定值，也可能是在被控对象运行过程中随时间变化的值，难以通过采样数据作出判断。一般采用多项式拟合、分

段平滑处理及分段线性化处理等方法去除直流分量。当被控对象处于稳定状态时，控制器输出也基本处于稳定状态。因此，本文选取分段线性拟合的方法进行控制器输出值的去直流分量操作。分段线性拟合的方法首先利用二阶线性拟合的方式对输入曲线进行拟合，利用拟合曲线中的极值点作为分段的分界点。之后分别用一阶线性拟合方法计算各分界点之间的拟合曲线，获得的拟合曲线就是测量值中的直流分量。

3.2 模型定阶

定阶是指确定系统辨识模型中参数的阶数。系统参数阶数的选择会对模型精度产生很大的影响。参数定阶的方法有很多种，如按残差方差定阶、FPE 定阶和 AIC 定阶等。FPE 定阶是指选取最终预报误差最小的阶数作为参数阶数，AIC 定阶则是用赤池定理确定模型阶数。因为 AIC 准则除了考虑模型的准确性之外，还充分考虑了模型的复杂性，因此，本文选择 AIC 准则作为定阶方法。

3.3 模型验证方法

当通过模型选择、模型参数定阶和辨识之后可以获得一个相对最优的系统模型，但是在实际应用模型进行入侵检测之前还需要进行模型验证。评价系统模型通常采用拟合误差法进行判断，拟合误差为

$$v=\sqrt{\frac{\sum_{i=1}^{n}(x_i-y_i)^2}{n}} \tag{2}$$

其中，x_i、y_i 是序列 X、Y 中的离散节点，n 为 X、Y 序列的长度。v 值越小表示 2 个离散数据序列的拟合程度越好，通常在系统辨识的模型评价中以拟合误差值作为评判标准。

如果模型的拟合误差值较小，且相关系数趋近于 1 就可以认为该模型是符合要求的，为了更加精确，本文再从模型误差的角度进行分析。定义 $e(k)$ 为模型输出值和实际测量值之间的残差序列，如果 $e(k)$ 可以近似为均值是 0 的白噪声序列，则可以认为模型是可靠的。因为当 $e(k)$ 为白噪声时，模型参数的估计值是一致性收敛的。如果得到的残差序列不是白噪声序列，则表明原有序列中还有规律信息未被提取，需要修改模型提取噪声中的剩余信息，否则建模结束。为了验证噪声 $e(k)$ 的白化性，可以对 $e(k)$ 做自相关曲线，如果自相关性很好，则表明 $e(k)$ 近似为白噪声，系统辨识获得的参数最佳，模型能够很好地描述实际系统。

样本残差的自相关函数可以表示为

$$\{R(l)\}=\frac{1}{M}\sum_{k=1}^{M}e(k)e(k+l) \tag{3}$$

残差的自相关函数方差可以表示为

$$Var\{R(l)\}\approx\frac{1}{M}R^2(0),l\neq 0 \tag{4}$$

如果噪声序列为正态分布白噪声，且置信水平为 0.95，则样本的自相关函数应尽可能地落在式(5)所示的置信区间范围内。

$$\frac{-1.96R(0)}{\sqrt{M}}<R(l)<\frac{1.96R(0)}{\sqrt{M}} \tag{5}$$

通过检验残差白噪声的自相关函数，可以判断系统辨识获得的模型残差序列是否近似为白噪声，进而判断模型是否符合实际的控制系统。

4 基于仿真建模的入侵检测系统

在检测攻击之前需要对攻击进行详细分析，本文将现场总线网络层作为目标的攻击方法进行了分类和总结。

4.1 现场总线网络攻击分类

现场总线网络攻击分为拒绝服务攻击、控制器代码篡改攻击、中间人攻击和重放攻击等。

4.1.1 拒绝服务攻击

拒绝服务攻击按照攻击原理可以分为 2 种：1) 资源耗尽式的拒绝服务攻击；2) 异常式的拒绝服务攻击。工业控制系统中的拒绝服务攻击以异常式的拒绝服务攻击为主，主要利用控制器和网络协议的安全漏洞构造异常的数据分组，导致控制器接收到数据分组后解析异常或堆栈溢出进而达到拒绝服务攻击的目的。典型的漏洞如 CVE-2014-5074 和 CVE-2015-2822 等。

4.1.2 控制器代码篡改攻击

通常来说，改变控制器的代码也是实现拒绝服务攻击的另一种变形，但是相比之下更难被察觉。攻击者可以通过多种手段修改控制器中的程序代码，如 Stuxnet 通过控制组态软件实现对 PLC 中控制逻辑代码的修改，CVE-2014-2249 利用跨站请求伪造漏洞，可的以直接远程执行未授权的操作实现代码篡改。

4.1.3 中间人攻击

中间人攻击的原理是在传输过程中截获数据分组，将数据分组中的内容篡改之后发出。因为控

制网络为了保证实时性，通常不采用加密机制，对于中间人攻击来说降低了攻击成本。这一类攻击无需利用设备或协议的漏洞，但是易受环境因素的影响。Stuxnet 攻击中 2 次利用了中间人攻击原理，一次是篡改组态软件对控制器的读写请求并注入恶意程序代码，另一次是篡改了控制器发出的数据分组，实现了恶意代码的隐藏。同样，如果不修改控制器中的控制逻辑代码，仅篡改控制器向被控对象发出的控制数据，也可以使原有控制逻辑失效。

4.1.4　重放攻击

由于现有的工业控制系统大量采用分布式部署方式，所以通常控制器和被控对象之间采用无线网络作为通信介质，这为重放攻击提供了便利。相比中间人攻击，重放攻击不需要拦截数据分组，只需按照数据分组的格式重构数据分组即可。此类攻击者只需要接入 Wi-Fi，并伪造虚假的传感器数据，就可以造成控制器因输入中包含脏数据而无法计算出正确的输出值，对被控对象造成影响。

4.2　基于仿真建模的入侵检测

本文的入侵检测方法是建立在现场总线设备正常模型的基础上，即分别对控制器和被控对象建模，并将上述模型整合成为一个工业控制网络入侵检测系统，工作原理如图 6 所示。

控制器入侵检测模块和被控对象入侵检测模块并行进行检测。控制器入侵检测模块主要应对针对控制器的攻击行为，如篡改控制器逻辑、拒绝服务攻击和篡改控制器输出值等。被控对象入侵检测模块主要是针对被控对象传感器数据分组的中间人攻击行为。只要两者中任意一个模型发现入侵，则认为当前工业控制系统的现场总线网络发生入侵。基于仿真建模的入侵检测系统的结构如图 7 所示。

4.2.1　控制器入侵检测方法

控制器仿真是为了保证如 PLC 发生入侵时能够被及时发现，即在假定输入信号可信的情况下，PLC 能够输出正确的数据给现场总线网络中的执行器。因此，入侵检测模块的重要工作是实时获取 PLC 中的输出，与仿真输出结果进行比较，如果 PLC 的输出结果偏离仿真结果，则认为发生入侵。

4.2.2　被控对象入侵检测方法

对于被控对象模型和实际控制器输出的测量值所产生的残差序列而言，常用的奇异点检测方法分为

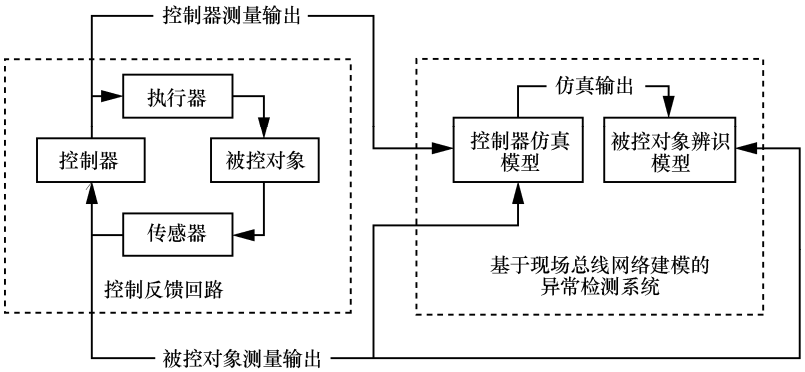


图 6　基于现场总线网络建模的入侵检测系统工作原理

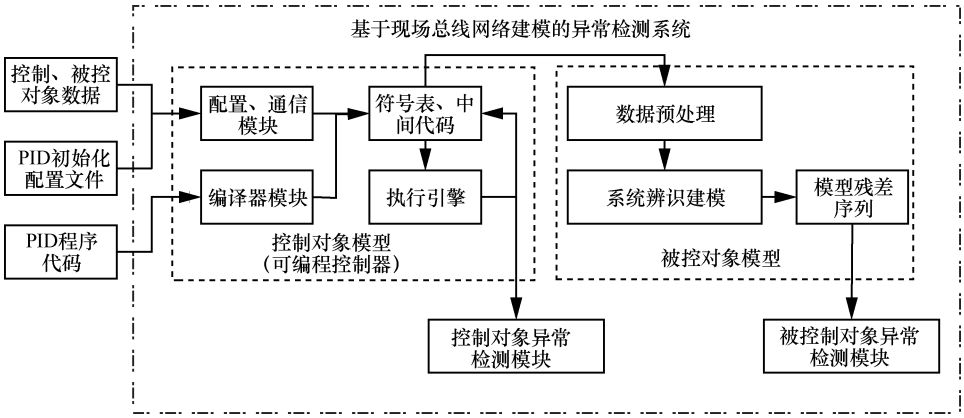


图 7　基于现场总线网络建模的入侵检测系统的结构

阈值法、自相关检测法以及小波变换检测法。残差阈值法通过判断残差是否在阈值范围内来进行检测，阈值过小容易产生误报，阈值过大容易漏报。自相关检测法通过判断残差曲线的自相关系数来进行检测，在打乱时序或攻击不明显的情况下，无法进行检测。小波变换经常被用于信号系统的故障检测中，小波变换的高频部分具有较高的时间分辨率和较低的频率分辨率。对于模型输出值和实际测量值之间的残差序列而言，小波变换获得的高频系数可以很好地反映残差序列短时间内的异常波动，通过小波变换可以放大残差序列的波动细节，因此，本文的异常检测方法是基于小波变换高频系数进行的。

5 实验结果与评估

本节将以水槽液位控制系统为例，在上文提到的建模方法基础上，阐述被控对象建模过程并进行入侵检测模型测试。

5.1 实验环境搭建

系统环境拓扑如图 8 所示。控制器采用西门子 PLC S7-300，外挂以太网通信模块，上位机运行西门子组态程序 Step7。

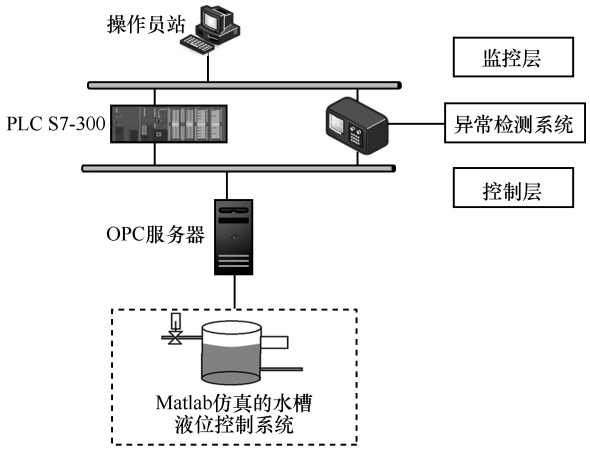


图 8 系统环境拓扑

水槽液位控制是经典的工业控制系统环节之一。水槽液位控制系统由一个水槽、出水口、入水口、液位传感器和阀门构成，其结构模型如图 9 所示。根据初始液位和设定值，控制器通过控制程序调节入水口阀门开度，控制进水量，同时，液位受进水量和出水量影响发生变化，液位的变化通过液位传感器反馈给控制器，控制器再次调节入水口阀门开度形成单回路反馈。本文根据水槽液位控制系统的原理利用 Matlab 实现了一个仿真的水槽液位控制系统，并通过仿真获取的入水口流量、出水口流量、液位数据以及控制器输出完成基于系统辨识方法的被控对象系统建模工作。

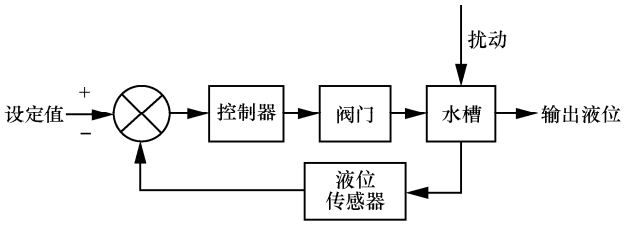


图 9 水槽液位控制系统反馈模型

操作员站利用 Step 7 V5.4 软件通过西门子私有的 S7 协议进行通信编程，使用西门子自带的 FB41 模块的 PID 算法，其中，*P*、*I*、*D* 参数分别为 2、0、0，*SP* 的值设为 1，即水槽通过控制算法要保持的液位高度。控制器是西门子 S7-300 PLC，型号为 CPU315-2 PN_DP。受控对象是基于 Matlab/Simulink 的仿真水槽，Matlab 选择了 R2014a 版本。仿真水槽高为 3 m，底面积为 1 m^2，出水管横截面积为 0.5 m^2，溢出传感器布置在水槽顶端，水槽的初始高度为 0.5 m。

5.2 基于系统辨识方法的水槽液位控制模型

槽控制器的工作原理如图 10 所示，根据 2.2 节所述方法，建立控制器的仿真模型。

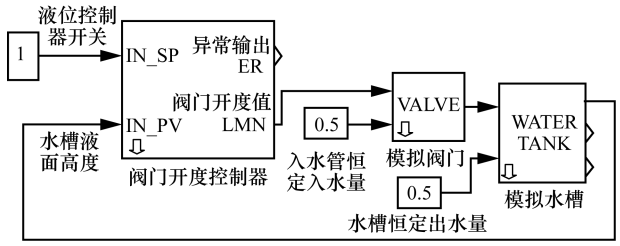


图 10 仿真的水槽液位控制系统

5.2.1 数据预处理

本文被控对象建模的重点是验证正常的控制器的输出值与反馈的输入值是否符合预期，因此，选取 Matlab 仿真获得的控制器输出和水槽出水口流量作为辨识模型的输入值，水槽的液位高度作为辨识模型的输出值来进行建模。首先以控制器输出值为例进行数据预处理，PLC 的输出值受控制程序调节入水口阀门开度的影响，逐渐达到设定液位，因此，PLC 对阀门开度的调节逐渐变小。

PLC 输出值序列和分段平滑处理后的输出值

序列如图 11 所示。本文平滑处理采用的滑动窗口长度是 15 s。

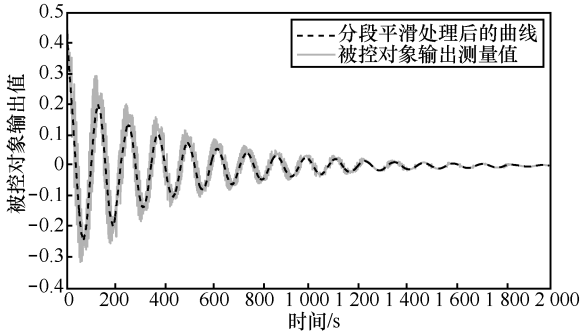


图 11　PLC 输出值序列和分段平滑处理后的输出值序列

从图 11 中可以看出，原有测量值输出曲线中的高频噪声基本消除。之后对分段平滑曲线进行二阶多项式拟合，拟合结果为 $y = 0.000\,013\,88x^2 - 0.003\,296x + 0.014\,46$。二阶线性拟合的极值点为 $(1\,187, -0.003\,3)$，如图 12 所示。

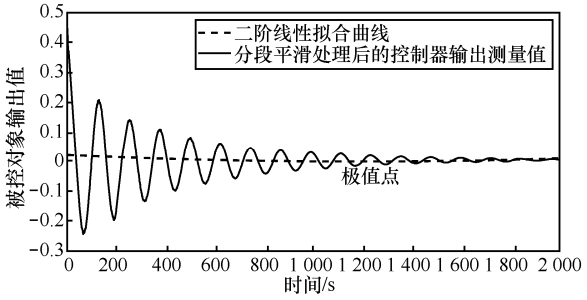


图 12　分段平滑曲线二阶拟合曲线的极值点

因此，设定$(1\,187, -0.003\,3)$点为分界点，对分界点前后分别进行一阶线性拟合，拟合结果如图 13 所示。分界点之前部分的一阶线性拟合函数结果为 $y = -0.000\,990\,53x + 0.012\,3$，分界点之后部分的一阶线性拟合函数结果为 $y = -0.000\,015\,959x + 0.029$。去除直流分量后的分段平滑曲线如图 14 所示，所得曲线即为系统辨识模型经过数据预处理之后的训练输入数据。

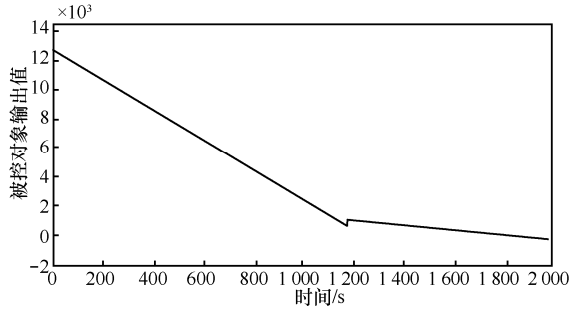


图 13　分段线性拟合曲线

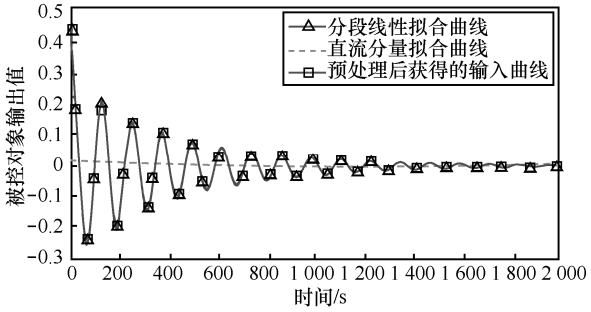


图 14　去除直流分量后的分段平滑

5.2.2　被控对象模型的建立

本文选取的模型类有 ARX、Box-Jenkins、N4SID 以及 ARMAX 模型，辨识工作采用 Matlab 系统辨识工具箱来完成。ARX、ARMAX、Box-Jenkins 以及 N4SID 模型的拟合相似度及拟合误差值如表 1 所示。可以看出相比其他 3 种模型 ARX 模型的拟合误差值最大，因为其抗干扰性能较差。根据本文 3.4 节中关于模型验证方法的描述，首先根据拟合误差值最小的原则选择模型，因此，ARMAX、Box-Jenkins 以及 N4SID 模型更适合本文被控对象的建模工作。在拟合误差值近似的条件下，可以继续利用拟合相似度的相关系数验证模型，N4SID 的拟合相似度相关系数为 0.856 3，明显低于 ARMAX 模型的 0.922 8 及 Box-Jenkins 模型的 0.964 4，因此，最终选出的模型是 ARMAX 模型和 Box-Jenkins 模型。

表 1　各系统辨识模型的拟合相似度及拟合误差

模型	拟合相似度	拟合误差
ARX 模型	0.891 9	0.272 3
ARMAX 模型	0.922 8	0.110 9
Box-Jenkins 模型	0.964 4	0.104 4
N4SID 模型	0.856 3	0.124 4

为了进一步验证辨识模型，对模型残差序列进行白噪声检验。利用 Matlab 计算得到 ARMAX 模型和 Box-Jenkins 模型的自相关函数，在置信水平为 0.95 的情况下，置信区间为$[-0.054\,48, 0.054\,48]$。2 种模型的残差自相关函数如图 15 所示。从图 15 中可以看出 Box-Jenkins 模型残差序列的自相关函数在 0 点附近的自相关指数并不趋近于 1，且未分布于置信区间内，这说明残差序列中还存在着一定的自相关特性，需要进一步挖掘。而 ARMAX 模型

的自相关函数完全分布于置信区间内，没有拖尾或截尾趋势，在 0 点处表现出无限接近于 1 的趋势，符合白噪声的自相关函数特性。因此，模型检验阶段最终选取 ARMAX 模型作为描述被控系统实际特性的模型。

5.3 攻击检测实例

利用上文中搭建的水槽控制系统，本文按照拒绝服务攻击、控制逻辑代码篡改、中间人攻击以及重放攻击的原理分别实现了攻击实例，并使用本文方法进行检测。攻击行为描述及检测结果如表 2 所示。

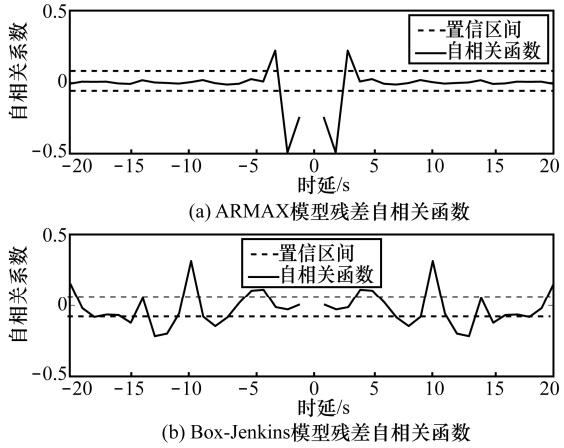


图 15 ARMAX 模型和 Box-Jenkins 模型的自相关函数曲线

5.3.1 基于控制器仿真建模的入侵检测

基于控制器仿真建模的入侵检测可以检测出拒绝服务攻击和逻辑代码篡改攻击。

表 2 攻击行为描述及检测结果

攻击类型	攻击描述	检测结果
拒绝服务攻击/重放攻击	截获组态软件用于关闭 PLC 的通信数据分组并重放导致 PLC 异常关闭	检出
逻辑代码篡改	修改 PLC 中控制逻辑的参数，造成控制结果发生偏差	检出
中间人攻击	修改液位高度传感器向 PLC 发送的实时液位高度，并恶意地将液位高度调高 0.1 m	检出

本文按照拒绝攻击原理，通过重放数据分组的形式实现针对 PLC 的拒绝服务攻击。PLC 接收到数据分组因为缺少加密、时间戳检验等抗重放攻击的手段，所以 PLC 会自动关闭。此时仿真模型发现 PLC 的输出数据为空（仿真模型将控制器输出值标记为 0），由于仿真输出和实际测量存在差异，故判断 PLC 出现异常。测量值和仿真输出值如图 16 所示。

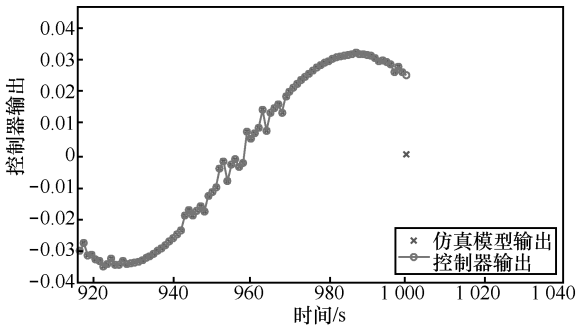


图 16 拒绝服务攻击发生时控制器仿真模型输出和控制器输出测量值

当 PLC 中控制逻辑代码被人为修改时，仿真模型中的控制逻辑代码未被修改，所以仍然运行正常的控制逻辑，因此，仿真输出和实际测量存在差异并被检测为异常，如图 17 所示。

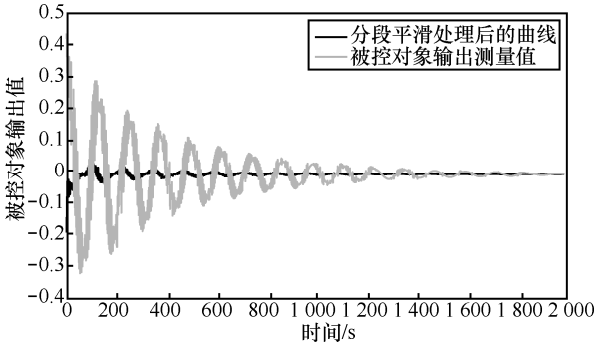


图 17 控制器控制逻辑被篡改后的输出测量曲线和仿真模型输出曲线

5.3.2 基于被控对象仿真建模的入侵检测

基于被控对象仿真建模的方法可以检测出中间人攻击。本文的入侵检测方法选取的小波移动尺度 $k=1\,000$，采用 $db(6)$ 小波函数进行 6 层分解，检测时利用高频系数 $d_{ig}(j,k)$ 进行入侵检测。由于在系统辨识建模之前进行了数据预处理操作，所以模型预测值和实际值的残差中包含有噪声信息和直流分量。通过对训练输入和模型输出的残差进行小波变换确定高频系数阈值，实验发现训练集的残差序列的高频系数 $d_{ig}(j,k)$ 的峰值为 0.3，因此，当残差序列的峰值高于 0.3 时，认为残差序列发生了突变，即被控对象出现了异常行为。

当中间人攻击对液位传感器的数据进行篡改时，会造成控制器因为错误的输入而产生错误的输出。通过 $db(6)$小波对获得的液位传感器和系统辨识模型的残差序列进行 6 层分解，其结果如图 18 所示。从结果中发现，如果部分曲线值超过阈值 0.3，则认为被控对象波动超过模型预期，认为被控对象发生异常。

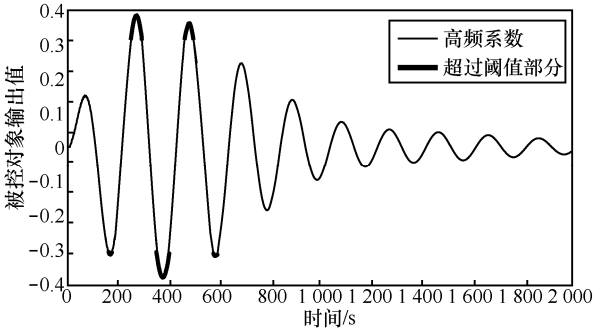


图 18　*db*(6)小波 6 层分解

6　结束语

本文提出了一种基于仿真建模的工控网络入侵检测方法。该方法由 4 部分构成，分别是控制器建模、被控对象建模、控制器入侵检测以及被控对象入侵检测，从 2 个层面对工控网络进行入侵检测。

本文提出的控制器模型是根据控制器原理实现的以编译器为核心的仿真模型。由于控制器仿真建模完全模拟了真实的控制器工作逻辑，且无法被篡改。因此，控制器入侵检测模块直接通过模型和实际值比对的方式检测异常。经实验验证可以检测针对控制器的已知威胁，如篡改控制器逻辑、拒绝服务攻击和修改控制器输出值等，也可在保证仿真模型不被篡改的前提下应对未知类型的攻击行为。

但是控制器入侵检测无法应对外部输入篡改攻击，因此，引入了被控对象模型。通过建立被控对象模型来保证控制器仿真模型和控制器获得的输入值是正确的。本文采用系统辨识的方法建立被控对象模型，具有先验知识要求小、通用性强以及能够在复杂的物理问题中建立数据对应关系的优势。并利用小波对测量值和估计值的残差序列进行奇异点检测，快速发现被控对象中存在的异常数据。经实验验证，本文所提被控对象入侵检测方法可以有效应对中间人攻击。

综上，本文所提控制器和被控对象仿真建模入侵检测方法可以有效地满足工业控制网络安全的需求。

参考文献：

[1] FALLIERE N, MURCHU L O, CHIEN E. W32 stuxnet dossier[R]. White Paper, Symantec Corp, Security Response, 2011.

[2] DONALD P C. The application of autonomic computing for the protection of industrial control systems[M]. Tucson: The University of Arizona, 2011.

[3] BENCSATH B, PEK G, BUTTYAN L, et al. Duqu: analysis, detection, and lessons learned[C]//ACM European Workshop on System Security (EuroSec). Bern, Switzerland, ACM, 2012: 1-6.

[4] STOUFFER K, FALCO J, SCARFONE K.SP 800-82, guide to industrial control systems (ICS) security[P]. National Institute of Standards & Technology,2011.

[5] 李琳，尚文利，姚俊，等. 单类支持向量机在工业控制系统入侵检测中的应用研究综述[J]. 计算机应用研究，2016，33(1): 7-11.
LI L, SHANG W L, YAO J, et al. Overview of one-class support vector machine in intrusion detection of industrial control system[J]. Application Research of Computers,2016, 33(1):7-11.

[6] CARDENAS A A, AMIN S, LIN Z S, et al. Attacks against process control systems: risk assessment, detection, and response[C]//The 6th ACM Symposium on Information, Computer and Communications Security. ACM, 2011: 355-366.

[7] WEI M, KIM K. Intrusion detection scheme using traffic prediction for wireless industrial networks[J]. Journal of Communications and Networks, 2012, 14(3): 310-318.

[8] BARBOSA R R R, SADRE R, PRAS A. Towards periodicity based anomaly detection in SCADA networks[C]//2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012). 2012: 1-4.

[9] RRUSHI J, KANG K D. Detecting anomalies in process control networks[M]. Critical Infrastructure Protection III. Springer Berlin Heidelberg, 2009: 151-165.

[10] MORRIS T H, JONES B A, VAUGHN R B, et al. Deterministic intrusion detection rules for Modbus protocols[C]//The 46th Hawaii International Conference on System Sciences (HICSS). 2013: 1773-1781.

[11] MORRIST, VAUGHN R, DANDASS Y. A retrofit network intrusion detection system for modbus RTU and ASCII industrial control systems[C]//The 45th Hawaii International Conference on System Science. 2012: 2338-2345.

[12] CARCANO A, COLETTA A, GUGLIELMI M, et al. A multidimensional critical state analysis for detecting intrusions in SCADA systems[J]. IEEE Transactions on Industrial Informatics, 2011,7(2): 179-186.

[13] FOVINO I N, CARCANO A, MUREL T D L, et al. Modbus/DNP3 state-based intrusion detection system[C]//2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA). 2010: 729-736.

[14] GOLDENBERG N, WOOL A. Accurate modeling of Modbus/TCP for intrusion detection in SCADA systems[J]. International Journal of Critical Infrastructure Protection, 2013, 6(2): 63-75.

[15] DAMIANI E. Composite intrusion detection in process control networks[D]. Uniersity Degli Studi Di Milano, 2009.

[16] LINDA O, MANIC M, VOLLMER T, et al. Fuzzy logic based anomaly detection for embedded network security cyber sensor[C]//IEEE Symposium on Computational Intelligence in Cyber Security. 2011:

202-209.

[17] LINDA O, VOLLMER T, MANIC M. Neural network based intrusion detection system for critical infrastructures[C]//International Joint Conference on Neural Networks. 2009: 1827-1834.

[18] ANOOP A, SREEIA M S. New genetic algorithm based intrusion detection system for SCADA[J]. International Journal of Engineering Innovations and Research, 2013, 2(2): 171-175.

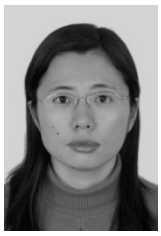
[19] 济晓. MATLAB 在振动信号处理中的应用[M]. 北京: 中国水利水电出版社, 2006.
JI X. The application of MATLAB in vibration signal processing[M]. Beijing: China Water Conservancy and Hydropower Press, 2006.

[20] 言俊科. 系统辨识理论及应用[M]. 北京: 国防工业出版社, 2003.
YAN J K. System identification theory and application[M]. Beijing: National Defence Industry Press, 2003.

作者简介：

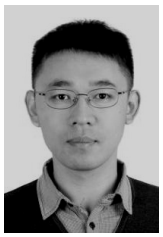
高一为（1990-），男，北京人，北京工业大学硕士生，主要研究方向为工业控制系统异常检测。

周睿康（1990-），男，浙江东阳人，中国电子技术标准化研究院助理工程师，主要研究方向为工业控制系统信息安全标准、检测、评估等。

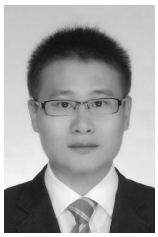


赖英旭（1973-），女，辽宁抚顺人，北京工业大学教授，主要研究方向为工业控制网络安全和软件定义网络安全。

范科峰（1978-），男，陕西礼泉人，博士后，中国电子技术标准化研究院高级工程师，主要研究方向为信息安全技术标准与测评方法等。



姚相振（1984-），男，山东济南人，博士，中国电子技术标准化研究院高级工程师，主要研究方向为信息安全技术标准与测评方法等。



李琳（1983-），男，山东济南人，博士，中国电子技术标准化研究院工程师，主要研究方向为信息安全、数据挖掘。