

基于异构执行体同步控制的入侵检测模型研究

于洪¹, 朱正彬¹, 魏帅¹, 郭威¹, 兰巨龙^{1,2}

(1. 信息工程大学, 河南 郑州 450001; 2. 嵩山实验室, 河南 郑州 450018)

摘要: 基于机器学习和深度学习的入侵检测及网络安全态势感知技术, 普遍存在依赖先验知识、需要提前训练的缺点, 使其在检测不同类型的攻击时准确率较低或动态适应性差。动态异构冗余 (dynamic heterogeneous redundancy, DHR) 构造技术可以通过判别异构执行体的行为一致性检测攻击。该发现为入侵检测提供了新的思路, 基于该思路提出了一种新的入侵检测模型——IDHES (intrusion detection model based on synchronization of heterogeneous executer), 可在无训练的前提下检测到多种类型的入侵。同时, 模型采用内外部事件转换的方式实现异构执行体目标功能同步, 降低了因执行体异构性导致的检测假阳率。通过对模型的检测原理及检测准确率进行理论分析, 得出 IDHES 检测准确率仅依赖异构执行体协同攻击成功率和目标功能同步效率这一推论。为验证模型有效性, 进一步构建了 DHR 架构的微控制器原型验证系统, 实现了基于内外部事件转换的目标功能同步方法。最后, 利用白盒插桩模拟攻击的方式对模型的攻击检测能力进行了测试, 测试结果验证了 IDHES 检测准确率仅依赖异构执行体协同攻击成功率和目标功能同步效率这一推论, 也表明 IDHES 可在不依赖先验知识、不提前训练的情况下, 对不同类型的攻击进行实时检测。

关键词: 入侵检测; 攻击感知; 内生安全; 执行体同步; 动态异构冗余

中图分类号: TP309.1

文献标志码: A

DOI: 10.11959/j.issn.2096-109x.2025006

Research on intrusion detection model based on synchronization of heterogeneous executer

YU Hong¹, ZHU Zhengbin¹, WEI Shuai¹, GUO Wei¹, LAN Julong^{1,2}

1. Information Engineering University, Zhengzhou 450001, China

2. Songshan Laboratory, Zhengzhou 450018, China

Abstract: Intrusion detection and cybersecurity situational awareness technologies based on machine learning and deep learning have generally been limited by their reliance on prior knowledge and pre-training, which restricts their accuracy in detecting various types of attacks. The dynamic heterogeneous redundancy (DHR) construction technique, which detects attacks by evaluating the behavioral consistency of heterogeneous executors, offers a new per-

收稿日期: 2024-05-10; 修回日期: 2025-01-20

通信作者: 于洪, yuhong_3210@163.com

基金项目: 国家重点研发计划 (2022YFB4401401)

Foundation Item: The National Key R&D Program of China (2022YFB4401401)

引用格式: 于洪, 朱正彬, 魏帅, 等. 基于异构执行体同步控制的入侵检测模型研究[J]. 网络与信息安全学报, 2025, 11(1): 54-65.

Citation Format: YU H, ZHU Z B, WEI S, et al. Research on intrusion detection model based on synchronization of heterogeneous executer[J]. Chinese Journal of Network and Information Security, 2025, 11(1): 54-65.

spective for intrusion detection. Based on this concept, a novel intrusion detection model called IDHES was proposed. This model was capable of detecting multiple types of intrusions without requiring pre-training. Additionally, synchronization of the target functions of heterogeneous executors was achieved through internal and external event conversions, thereby reducing the false positive rate caused by the heterogeneity of executors. Through theoretical analysis of the model, it was concluded that the detection accuracy of the IDHES model depends solely on the success rate of coordinated attacks by heterogeneous executors and the efficiency of target function synchronization. To verify the effectiveness of the model, a prototype MCU system based on the DHR architecture was constructed, and the target function synchronization method was implemented through internal and external event conversions. Finally, the attack detection capability of the model was tested using white-box instrumentation. The test results confirm the conclusion that the detection accuracy of the IDHES model depends solely on the success rate of coordinated attacks by heterogeneous executors and the efficiency of target function synchronization. Furthermore, the results demonstrate that IDHES can perform real-time detection of various types of attacks without relying on prior knowledge or pre-training.

Keywords: intrusion detection, attack awareness, endogenous security, executor synchronization, dynamic heterogeneous redundancy

0 引言

互联网技术的飞速发展,使其融入了人类活动的方方面面,也催生了带有不同目的、多种形式的网络攻击行为。越早发现网络攻击行为,就可越大程度减少其带来的伤害。目前检测网络攻击最常用的技术是入侵检测系统(intrusion detection system, IDS)^[1]和网络安全态势感知(network security situation awareness, NSSA)^[2]。其中,入侵检测系统有基于主机数据的入侵检测系统^[3](host-based IDS, HIDS)、基于网络数据的入侵检测系统^[4](network-based IDS, NIDS),以及结合了HIDS和NIDS的分布式入侵检测系统(distributed IDS, DIDS)。HIDS通常布置在主机上,通过分析主机上的操作、事件和状态数据^[5],来判断主机是否被入侵。NIDS^[6-7]则是监视整个网络,分析网络上的数据包,获取网络连接、流量特征,以此分析网络入侵行为。NSSA^[8-12]在IDS技术的基础上,融合处理网络整体状态信息,对网络整体安全形势进行总结,除了感知攻击活动,还要理解其意图,剖析攻击者的动机和目标。

IDS和NSSA都可以检测、感知到网络威胁事件,但是作为外挂式的安全手段,它们都存在一定缺陷。一是不同程度地依赖先验知识,因此对于已知类型的攻击检测准确度高,对未知类型的攻击不具备检测能力,或者检测准确率低^[13],目前没有一种方法可以在模型训练时间、检测时

间、先验知识、检测准确度之间做很好的平衡。二是训练需要的数据量大,如果花费人力对数据样本进行分类,则代价高昂,如果使用算法学习,则检测精度较低,误报率较高。三是IDS使用的数据集,需要先在安全的环境下离线运行,检测出了问题,才可以在主机/网络中继续使用,无法提供很好的实时性。四是方法的针对性很强,无法适应攻击方式或者攻击代码的动态改变,对未知类型的攻击检测效率低。

针对IDS和NSSA存在的问题,研究以内生安全和动态异构冗余(dynamic heterogeneous redundancy, DHR)构造^[14-15]为基础,寻找新的入侵检测方案。

DHR构造通常包含多个目标功能等价的异构执行体、输入代理、裁决输出、反馈控制、策略管理和动态调度几个单元(如图1所示)。输入代理将外部输入分发到所有异构执行体上,异构执行体进行独立、并行处理,得到的结果经过系统裁决后输出。其安全的根本在于对重点保护组件的动态异构冗余,通过动态调度异构执行体,增加攻击者对组件脆弱性和漏洞进行扫描探测的难度,同时也可斩断攻击链,增加攻击者侵入组件造成破坏的难度。冗余使系统在个别执行体被击穿的情况下,仍可提供服务,提升了系统的可靠性和弹性可恢复性。异构执行体在攻击下行为不一致,结合系统裁决可检测识别攻击,通过及时处理可以减小攻击伤害,也可增加攻击者将非法数据带出组件的难度。

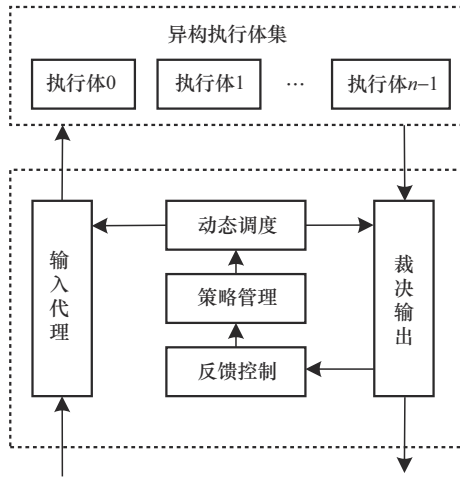


图 1 DHR 基本构造

Figure 1 DHR basic structure

利用 DHR 提升系统内生安全属性的研究很多^[16-21], 这些研究中, 执行体的目标功能等价性和非目标功能异构性常被作为一个前提来使用, 辅助入侵检测及调度裁决研究, 异构执行体的目标功能同步作为检测的前提被忽略。除此之外, 缺乏利用 DHR 构造进行入侵检测的模型化描述和系统性研究。

本文针对上述 IDS 和 NSSA 检测技术缺陷及 DHR 入侵检测存在的问题, 利用模型化的描述语言深入分析 DHR 架构存在的执行体不同步问题, 并计算其对网络入侵行为检测成功率的影响, 提出一种基于异构执行体同步控制的入侵检测模型——IDHES (intrusion detection based on synchronization of heterogeneous executor, IDHES), 分析异构执行体不同步原因, 利用内部事件同步控制, 达成异构执行体的目标功能同步, 降低了执行体异构性带来的入侵检测误报率, 达到精确检测网络入侵的目的。为验证 IDHES 的有效性, 本文设计了一个三模冗余的微控制器 (micro-controller unit, MCU) 原型验证平台, 利用白盒插桩方式模拟攻击, 测试 IDHES 检测准确率, 结果表明 IDHES 在检测准确率方面相比大多数基于深度学习的检测方法有所提升。并且, 区别于 IDS 和 NSSA 这些外挂式的检测手段, IDHES 利用自身 DHR 构造带来的动态性、异构性、冗余性优势, 脱离传统方式对先验知识的依赖, 将传统的入侵检测问题, 从依赖先验知识有无的 (0,1) 问题, 转换为依赖执行体同步效率和执行体协同攻击成

功率的概率问题, 并且不受限于特定种类的攻击模式, 不需要提前学习, 可以实时、准确、高效地检测各种已知和未知的系统入侵行为。IDHES 解决了 IDS 和 NSSA 在检测入侵方面存在的固有问题, 为入侵检测提供了新的思路。

1 IDHES

本节构建了 IDHES, 分析了 DHR 架构中存在的不同步问题及协同攻击问题对 IDHES 的影响。

为了方便后文描述, 表 1 中列出了后续使用的符号及定义。

1.1 同步问题描述

前面提到 DHR 构造通常包含多个目标功能等价的异构执行体, 这里的目标功能指系统外部关注、使用到的功能, 更靠近应用程序层。与之对应的概念为非目标功能, 即系统外部不关注、用不到的功能, 也叫暗功能, 更靠近操作系统和硬件层。目标功能等价, 指导异构执行体在相同状态下, 接收相同的输入, 可产生一致的输出和后续状态。非目标功能则具有特异性, 在面对相同的输入和操作时, 可能产生不一致的输出和状态。非目标功能的特异性是由执行体的异构性赋予的。比如, 内存排布方式使内存数据读写参数不同、系统调用编号使系统调用参数不同、操作系统调度方式使进程执行顺序不同等。

当系统工作时, 可用一个状态的无限序列 $S(i)$ 来表示执行体 E_i 上的程序运行 ($S_{i,j}$ 及 $S'_i(j, j+1)$ 的含义见表 1):

$$S(i) = [S_{i,0}, S'_i(0,1), S_{i,1}, S_{i,2}, S'_i(2,3), S_{i,3}, \dots] \quad (1)$$

因为目标功能和非目标功能会交替运行, 因此目标状态和非目标状态会相互交叉, 且作为前序的非目标状态会影响后序的目标状态。由于系统外部不关注非目标功能, 也不关注非目标状态, 因此, 系统外部实际只关注执行体 E_i 的目标状态序列, 即:

$$\text{Stgt}(i) = [S_{i,0}, S_{i,1}, S_{i,2}, \dots] \quad (2)$$

DHR 构造在理想情况下, 期望所有执行体的目标功能集合都相等, 且非目标功能不存在交集, 即对 $\forall i, j \in [0, n)$, 存在:

$$\text{Ftgt}(i) = \text{Ftgt}(j), \text{Ftgt}(i) \cap \text{Ftgt}(j) = \emptyset \quad (3)$$

表 1 符号及定义

Table 1 Symbols and definitions

符号	含义
$\text{Exe} = \{E_0, E_1, \dots, E_{n-1}\}$	异构执行体集合, 其中 E_i 代表单个执行体, n 为奇数
$\text{Ftgt}(i) = \{F_{i,0}, F_{i,1}, \dots, F_{i,m-1}\}$	执行体 E_i 的目标功能集合, $F_{i,j}$ 表示执行体 E_i 的某个目标功能
$\text{F'tgt}(i) = \{F'_{i,0}, F'_{i,1}, \dots, F'_{i,k-1}\}$	执行体 E_i 的非目标功能集合, $F'_{i,j}$ 表示执行体 E_i 的某个非目标功能
$\text{F'tgt}(i, j) = \{F'_0(i, j), F'_1(i, j), \dots, F'_{l-1}(i, j)\}$	执行体 E_i 和 E_j 之间共同的非目标功能集合
$\text{Stgt}(i) = [S_{i,0}, S_{i,1}, S_{i,2}, \dots]$	执行体 E_i 上目标功能运行产生的目标状态序列, 其中目标状态 $S_{i,j}$ 指一个目标状态
$S'_i(p, p+1)$	执行体 E_i 上非目标功能运行产生的介于目标状态 $S_{i,p}$ 和 $S_{i,p+1}$ 之间的非目标状态
α_i	执行体 E_i 被攻击成功的概率
β	表示任意执行体协同漏洞存在概率, β^{k-1} 表示 k 个执行体协同攻击漏洞存在概率
γ	目标功能同步的效率
$\text{Scn}(i)$	第 i 种系统场景
$P(\text{Scn}(i))$	第 i 种系统场景出现的概率

这样 DHR 构造就可以防护所有基于非目标功能的已知和未知漏洞/后门攻击。因为针对非目标功能的攻击行为在异构执行体上会有不一致的呈现, 可以直接被 DHR 结构探查。针对目标功能共性特征的攻击, 不在 DHR 构造防护范围之内。比如, 基于协议本身漏洞的攻击, 即使在异构执行体上适配不同的协议版本, 依然存在协议漏洞, 针对这个漏洞的攻击 DHR 无法防护。此类攻击不在本文讨论范围之内。

如果执行体之间存在共同的非目标功能, 且攻击者基于此功能发动了协同攻击, 一旦协同攻击数目超过所有执行体的半数, 攻击行为就无法被 DHR 系统有效识别和抵御。这个困境可以采用适当的执行体调度策略解决^[22], 在本文中不涉及。

现实情况下, DHR 还存在一个问题, 即在目标功能等价的情况下, 异构执行体的目标状态序列也不会完全相同, 即对 $\forall i, j \in [0, n)$, 存在:

$$\text{Stgt}(i) \neq \text{Stgt}(j) \quad (4)$$

典型原因有硬件层面异构存在的性能差异导致的程序运行速度不一致。在同一时刻 t , 性能较高的 E_i 处于状态 $S_{i,2}$, 而性能较低的 E_j 处于状态 $S_{j,1}$, 此时接收到消息 M , 处于不同状态的执行体就会输出不同的结果, 或者进入完全不同的状态, t 时刻后的系统状态将变得不可控。系统时钟的细微差异也可能使内部定时器事件超时

机不一致。对于周期定时器来说, 这种偏差的累积会逐渐增大。以 RIP 定时发送 response 报文为例, 初始时几毫秒的发送时差在 30 min 过后, 就可能超过 500 ms, 1 h 后超过 1 000 ms。这种情况下, 裁决等待会造成对端协议超时, RIP 无法正确更新路由, 造成网络中断。软件层面存在的操作系统调度策略差异也会导致对应用事件响应顺序的不一致。在同一时刻 t , 假设 E_i 处于状态 $S_{i,2}$, 而 E_j 处于状态 $S_{j,2}$, 且 $S_{i,2} = S_{j,2}$, 待执行事件集合也一致。 E_i 的调度策略与 E_j 不同, 从待执行事件集合中选择的下一个执行事件不一致, 则 t 时刻后的执行体的状态序列将出现分化。最后驱动层面的异构也会影响系统非目标功能运行快慢, 在非目标状态的持续时间存在的差异最后也会反映到目标状态上。

1.2 IDHES 抽象

IDHES 通过对异构执行体的目标状态进行同步控制, 使各执行体的目标状态序列相同, 即正常情况 (所有执行体均未受攻击) 下:

$$\forall i, j \in [0, n), k \geq 0, S_{i,k} = S_{j,k} \quad (5)$$

排除使用 DHR 构造检测攻击的干扰, 通过比较使系统能精准识别针对执行体非目标功能的攻击并进行相应的处理。

IDHES 对异构执行体目标功能进行同步的效率 γ 可表示为:

$$\gamma = \sum_{i=1}^a \varphi_1(\kappa_i) \varphi_2(\kappa_i) \quad (6)$$

其中 κ_i 为系统的待同步因子, $\varphi_1(\kappa_i)=1$ 表示对 κ_i 进行同步控制, $\varphi_1(\kappa_i)=0$ 表示不对 κ_i 进行同步控制, $\varphi_2(\kappa_i)$ 表示 κ_i 出现的次数占所有待同步因子出现次数的百分比。

对同步后 DHR 系统, 存在以下几种场景。

1) Scn(1): 所有执行体均未遭受攻击, $P(\text{Scn}(1))$ 表示其出现概率;

2) Scn(2): 部分执行体遭受攻击, 其他执行体正常, $P(\text{Scn}(2))$ 表示其出现概率;

3) Scn(3): 所有执行体遭受攻击, 但非全部协同, $P(\text{Scn}(3))$ 表示其出现概率;

4) Scn(4): 全部执行体遭受协同攻击, $P(\text{Scn}(4))$ 表示其出现概率。

系统对上述 4 种情况的判别可以看作一个二分类问题。其中, TP (true positive) 表示系统遭受且感知到入侵这种情况, FN (false negative) 表示系统遭受入侵但没有感知到入侵这种情况, TN (true negative) 表示系统没有遭受也没有感知到入侵, FP (false positive) 表示系统没有遭受入侵但感知到入侵。如果采用简单的大数裁决策略, 则系统对上述情况判别如表 2 所示。

表 2 不同场景下 DHR 结构判别结果

Table 2 Voting result of DHR in different scenarios

场景	系统整体判别	结论	出现概率
Scn(1)	受攻击	FP	$(1-\gamma)P(\text{Scn}(1))$
	未受攻击	TN	$\gamma P(\text{Scn}(1))$
Scn(2)	受攻击	TP	$P(\text{Scn}(2))$
	未受攻击	FN	0
Scn(3)	受攻击	TP	$P(\text{Scn}(3))$
	未受攻击	FN	0
Scn(4)	受攻击	TP	0
	未受攻击	FN	$P(\text{Scn}(4))$

用 α_b 表示执行体 E_b 的受攻击概率, β^{n-1} 表示 n 个执行体受协同攻击的影响因子, 有:

$$P(\text{Scn}(1)) = C_n^n \prod_{b=1}^n (1 - \alpha_b) \quad (7)$$

$$P(\text{Scn}(4)) = C_n^n \prod_{b=1}^n \alpha_b \beta^{n-1} \quad (8)$$

IDHES 的检测准确率为 TP 和 TN 样本占所有样本的比例, 可表示如下:

$$\text{ACC}_{\text{IDHES}} = \frac{\sum_{i=2}^3 P(\text{Scn}(i)) + \gamma P(\text{Scn}(1))}{\sum_{i=1}^4 P(\text{Scn}(i))} \quad (9)$$

IDHES 的检测精确率为 TP 样本占 TP 和 FP 样本和的比例, 可表示如下:

$$\text{precision}_{\text{IDHES}} = \frac{\sum_{i=2}^3 P(\text{Scn}(i))}{\sum_{i=2}^3 P(\text{Scn}(i)) + (1 - \gamma) P(\text{Scn}(1))} \quad (10)$$

IDHES 的检测召回率为 TP 样本占 TP 和 FN 样本之和的比例, 可表示如下:

$$\text{recall}_{\text{IDHES}} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\sum_{i=2}^3 P(\text{Scn}(i))}{\sum_{i=2}^4 P(\text{Scn}(i))} \quad (11)$$

IDHES 的 $F1_Score$ 为精确率和召回率的调和平均数, 可表示如下:

$$F1_Score_{\text{IDHES}} = \frac{2}{1 - \frac{P(\text{Scn}(4)) + (1 - \gamma) P(\text{Scn}(1))}{\sum_{i=2}^3 P(\text{Scn}(i))}} \quad (12)$$

从表 2 及计算分析可以看出, 只有在 Scn(4) 出现时, 检测会出现 FN 误判的情况, 而误判的概率完全取决于 $P(\text{Scn}(4))$; 在 Scn(1) 出现且同步失效时, 会出现 FP 误判的情况, 且误判概率取决于同步技术的效率 γ 以及 Scn(1) 的出现概率 $P(\text{Scn}(1))$ 。由此, IDHES 将传统入侵检测成功率问题由依赖先验知识的 (0,1) 问题转变为了依赖异构执行体同步效率和协同攻击成功的概率问题。

由式(9)和式(10)可知, 想提高 IDHES 检测准确率和精确率, 需要提高执行体目标功能同步控制率 γ 。接下来, 分析异构执行体中导致目标功能不同步的主要因素 (内部事件失序), 提出一种基于内部事件转换为外部事件的异构执行体同步控制技术, 用于提高 γ , 获取更高检测准确率和精确率。

2 异构执行体目标功能同步控制

内部事件为应用程序目标功能内部触发的，典型的如定时器事件及内部读写请求等，与之对应的外部事件是应用程序外部触发的，如消息处理事件等。1.1节中提到导致执行体目标功能状态不同步的典型原因，不论是性能差异、时钟差异，还是调度策略差异，最终都会反映到内部事件排序上。

在DHR系统中，调度器具有输入代理功能，外部输入的消息都要通过调度器统一分发到所有异构执行体。因此，外部事件天然可以通过调度器保持同步，内部事件则会由于执行体的异构等触发顺序不一致，从而产生异构执行体程序的状态转换不一致的情况。

如果将内部事件转换为外部事件，则可以消除硬件性能原因造成的内部事件延迟和系统调度原因造成的内部事件执行失序，使发生时间和顺序不可控的事件变得可控。

本节以定时器事件为例，说明基于内外部事件转化的同步方法处理流程。

一般定时器事件处理流程如图2所示，中断处理执行完后应用程序状态发生转变。由于该状态转变是由内部定时器事件引起的，基于前述多种原因，不能保证在其他异构执行体上，一定会发生同时的、一致的改变。

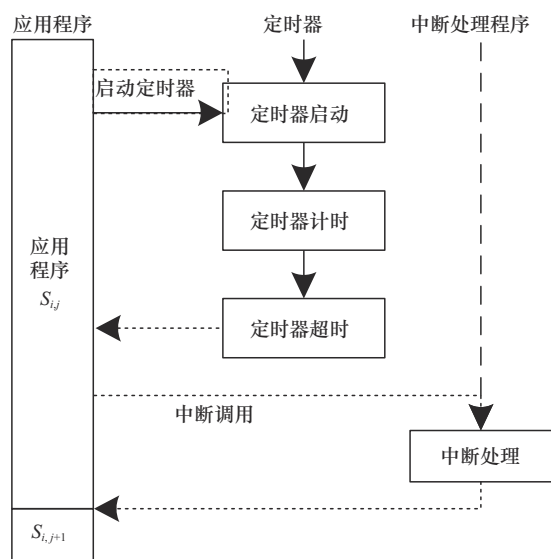


图2 一般定时器事件处理流程
Figure 2 Regular timer event processing

于是，对定时器事件的处理流程进行了修改，如图3所示，将内部事件转变为外部事件。

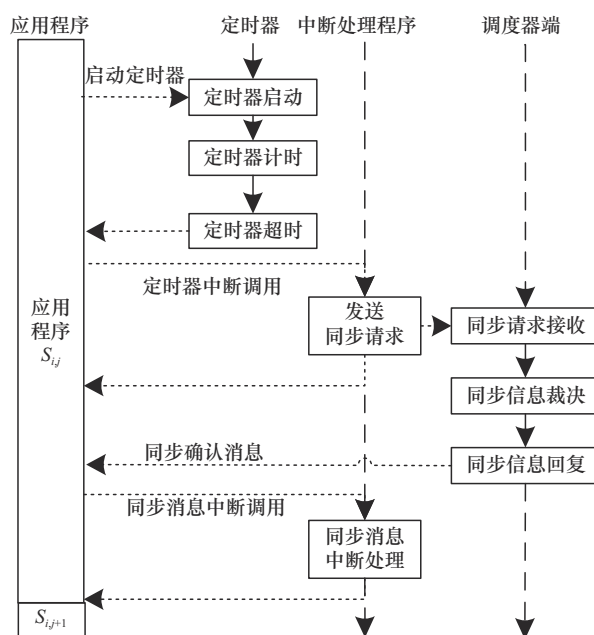


图3 修改后的定时器事件处理流程
Figure 3 Modified timer event processing

在定时器超时引起的中断处理中，向调度器发送一个携带有定时器标识等信息、不会引起应用程序的状态改变同步请求消息。调度器端在接收到同步请求消息后，对其进行裁决，合法地予以回复。应用程序接收到同步确认消息，调用消息处理接口函数，根据回复消息携带的定时器信息，完成对对应定时器超时操作。消息处理完毕，应用程序才会发生状态转换。

通过这样的方式，将定时器事件转换成了消息事件，通过调度器的统一分发，保证了所有执行体的事件执行顺序，由此保证异构执行体的状态转换一致性。

其他内部事件的处理流程，与定时器处理流程类似。在此不再赘述。

3 IDHES检测原型系统

3.1 MCU原型系统架构

为了对IDHES的有效性和性能进行测试，本节设计一个带IDHES的MCU原型系统，采用从硬件、驱动到应用程序的三模异构冗余，其架构如图4所示。

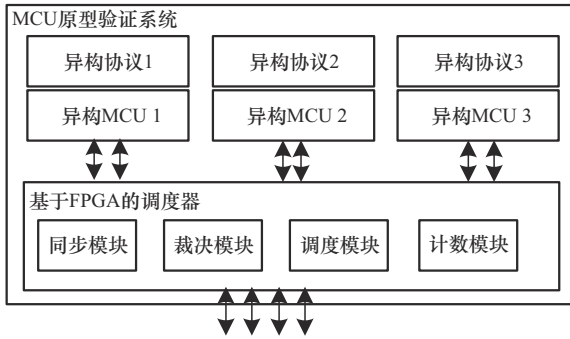


图 4 基于DHR的MCU架构

Figure 4 MCU architecture based on DHR

其中执行体硬件采用MIPS、ARM和AVR这3种架构的32位MCU，从3个核分别引了两路串口与调度器相连，一路管理接口传输同步消息，另一路数据接口传输其他消息。

执行体应用程序采用了开源Modbus-RTU协议主程序，对代码进行了针对异构处理器的适配。

调度器使用FPGA实现，包含同步、裁决、调度和计数等功能模块。

3.2 IDHES功能实现

执行体应用Modbus-RTU主程序中用到了查询传感器状态的周期性定时器（周期为1s），定时器的中断处理程序被修改为向调度器发送同步请求消息。在接收到来自调度器的响应消息后，执行器调用消息处理中断，执行原定时器中断操作，异构执行体端同步消息处理流程如图5所示。

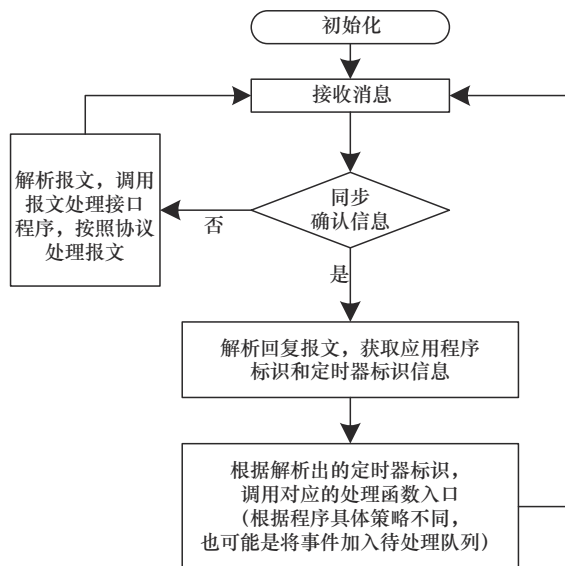


图 5 异构执行体端同步消息处理流程

Figure 5 Synchronization message process on executors

应用程序中还设置了方便测试的桩函数，用于模拟执行体受攻击的情况。在应用程序消息处理中加入一段解析函数，如果消息中包含特定的字段，桩函数就会被激活。桩函数被激活的执行体会每隔一秒随机向外发送一个处于(0,10]区间的整数，通过一个管理界面配置使能和关闭，用以模拟不同的场景（模拟攻击场景配置及对应表象如表3所示）。

调度器由FPGA实现，同步模块功能包括同步消息的接收、仲裁和回复，调度器端同步消息处理流程如图6所示。

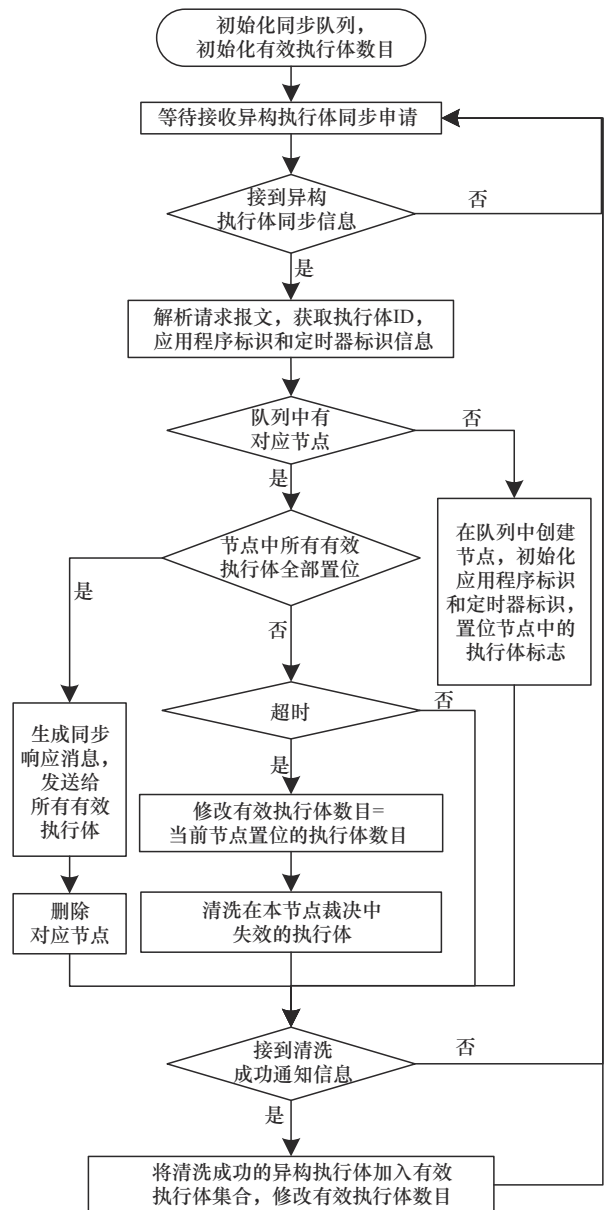


图 6 调度器端同步消息处理流程

Figure 6 Synchronization message process on scheduler

表3 模拟攻击场景配置及对应表象

Table 3 Simulated attack scenario settings and corresponding semblance

场景	配置	对外可能呈现的表象
没有执行体受攻击	所有执行体桩函数不使能	3个执行体的输出一致
执行体 E_1 受攻击	使能 E_1 的桩函数	执行体 E_1 输出与 E_2, E_3 不一致
执行体 E_2 受攻击	使能 E_2 的桩函数	执行体 E_2 输出与 E_1, E_3 不一致
执行体 E_3 受攻击	使能 E_3 的桩函数	执行体 E_3 输出与 E_1, E_2 不一致
执行体 E_1 与 E_2 受攻击	使能 E_1 与 E_2 两个执行体的桩函数	执行体 E_3 输出与 E_1, E_2 不一致 3个执行体的输出各不相同
执行体 E_1 与 E_3 受攻击	使能 E_1 与 E_3 两个执行体的桩函数	执行体 E_2 输出与 E_1, E_3 不一致 3个执行体的输出各不相同
执行体 E_2 与 E_3 受攻击	使能 E_2 与 E_3 两个执行体的桩函数	执行体 E_1 输出与 E_2, E_3 不一致 3个执行体的输出各不相同
执行体 E_1, E_2, E_3 均受攻击	使能3个执行体的桩函数	3个执行体的输出各不相同 执行体 E_1 输出与 E_2, E_3 不一致 执行体 E_2 输出与 E_1, E_3 不一致 执行体 E_3 输出与 E_1, E_2 不一致 3个执行体的输出均相同

注：正常执行体无额外输出，但是这个表里，“无输出”也是一种输出状态。

调度器维护所有有效执行体信息以及一个全局变量“有效执行体数目”，代表目前功能等价异构执行体集合中，正常对外提供服务的执行体数目。同时维护一个同步请求队列，队列中的一个节点代表一个定时器事件请求，使用应用程序标识和定时器标识来唯一确定。节点中有用于执行体同步标志字段（每个位对应一个执行体），当收到一个异构执行体的请求，就将对应的位置1。当所有有效执行体的对应标志位都置1，则代表调度器收到了所有有效执行体针对某定时器的同步请求，调度器可以进行下一步操作。初始状态下，同步请求队列为空。调度器收到一个来自某异构执行体 E_i 的定时器 $Timer_i$ 的同步请求信息，首先确认队列中是否存在对应节点。如果不存在，则创建一个新的节点，并将同步标志字段中表示 E_i 的位置1；如果已经存在对应节点，则进一步判断是否所有有效执行体的对应标志位都置1。如果都置1，代表调度器已经收到所有有效执行体针对 $Timer_i$ 的同步请求，可以生成同步确认消息；如果没有，则在超时范围内继续等待。超过预定时间，调度器还没有集齐所有请求，则判断这一轮中同步标志位没有置1（但上

一轮置1）的执行体失步，可以向其发出清洗恢复指令。

调度器还实现了大数裁决模块用于比较裁决执行体输出，判断系统是否遭受攻击；调度模块用于同步过程中清洗执行体；计数模块用于分类统计调度器裁决次数。

计数模块统计4类数据。

- 1) SysAttCnt：系统受攻击次数。
- 2) nSysAttCnt：系统未受攻击次数。
- 3) AttCnt(i)：执行体 E_i 受攻击次数。
- 4) nAttCnt(i)：执行体 E_i 未受攻击次数。

计数模块的工作策略如表4所示，当3个执行体的输出一致时，系统未受攻击次数和各执行体未受攻击数均+1；某执行体输出与其他执行体不一致时，系统受攻击次数与该执行体受攻击次数均+1；3个执行体输出均不相同，系统受攻击次数与各执行体受攻击次数均+1。

通过上述修改和实现，Modbus-RTU 原来的内部定时器事件被转换为外部消息事件，解决了内部事件排序不一致和差异累积的问题，统一了异构执行体上程序状态转换的顺序，提升了异构执行体的目标功能同步率 γ 。由式(9)

和式(10)知, 可获取更高检测准确率和精确率。

表 4 计数模块的工作策略

Table 4 Counting module work strategy

对外呈现表象	计数模块操作	
3个执行体的输出一致	nSysAttCnt+1	nAttCnt(1)+1 nAttCnt(2)+1 nAttCnt(3)+1
执行体 E_1 输出与 E_2, E_3 不一致	SysAttCnt+1	AttCnt(1)+1
执行体 E_2 输出与 E_1, E_3 不一致	SysAttCnt+1	AttCnt(2)+1
执行体 E_3 输出与 E_1, E_2 不一致	SysAttCnt+1	AttCnt(3)+1
3个执行体的输出各不相同	SysAttCnt+1	AttCnt(1)+1 AttCnt(2)+1 AttCnt(3)+1

4 IDHES 实验

IDHES 实验环境如图 7 所示。

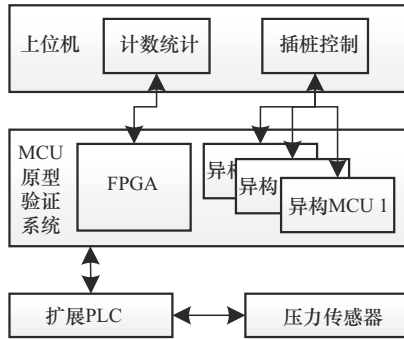


图 7 实验环境

Figure 7 Test environment

采用如下的实验设置。

- 1) 启用 E_1 的桩函数, 设置发送上限为 1 000 次。
- 2) 启用 E_1, E_2 的桩函数, 设置发送上限为 1 000 次计数。
- 3) 启用 E_1, E_2, E_3 的桩函数, 设置发送上限为 1 000 次计数。

计数模块测试数据如表 5 所示。

由实验数据可以看出: 所有正常数据通过 FPGA 都没有发生 FP 误判, IDHES 在 MCU 验证系统上的同步效率 γ 为 100%。主要原因在于 modbus-RTU 程序比较简单, 定时器事件单一,

如果在更复杂的系统上实现, 如带操作系统的, 目前仅同步定时器事件的方案将无法达到 100% 的效率, 需要配合其他事件 (如随机数生成) 的处理。

表 5 计数模块测试数据

Table 5 Counting module test data

类别	计数		
	设置 1	设置 2	设置 3
SysAttCnt	1 000	1 000	991
nSysAttCnt	1 000	1 000	1 009
AttCnt(1)	1 000	904	808
nAttCnt(1)	1 000	1 096	1 192
AttCnt(2)	0	904	811
nAttCnt(2)	1 000	1 096	1 189
AttCnt(3)	0	1 000	810
nAttCnt(3)	1 000	1 000	1 190

另外也可以推出: 在实验设置 1 下, 系统能够准确识别针对 E_1 的 1 000 次攻击行为; 在实验设置 2 下, E_1, E_2 的 1 000 次攻击中, 产生了 96 次协同攻击行为, 导致系统对单个执行体攻击识别的失误, 但对系统整体的攻击识别准确; 在实验设置 3 下, E_1, E_2 产生了 91 次协同攻击行为, E_1, E_3 产生了 92 次协同攻击行为, E_2, E_3 产生了 89 次协同攻击行为, 这些攻击行为导致了系统对单个执行体攻击识别的失误, 但是对系统整体的攻击识别准确; E_1, E_2, E_3 产生了 9 次协同攻击行为, 导致系统对整体攻击识别产生 FN 结果。

将上述结果代入式(9), 计算 IDHES 在 MCU 上的理论检测准确率为:

$$ACC_{MCU} = \frac{\sum_{i=2}^7 P(\text{Scn}(i)) + \gamma P(\text{Scn}(1))}{\sum_{i=1}^8 P(\text{Scn}(i))} = 99.875\%$$

由于插桩函数手动控制, 为了方便结果展示, 将每个执行体的被攻击成功概率 (表 1 中的 α_i) 都设置为 50%。同时由于桩函数的特性, 两个执行体间协同漏洞存在概率 (表 1 中的 β) 为 10%。在此情况下, 三冗余系统出现不同场景的概率如表 6 所示。

表6 MCU验证系统运行场景及概率

Table 6 MCU prototype working scenarios and corresponding rate

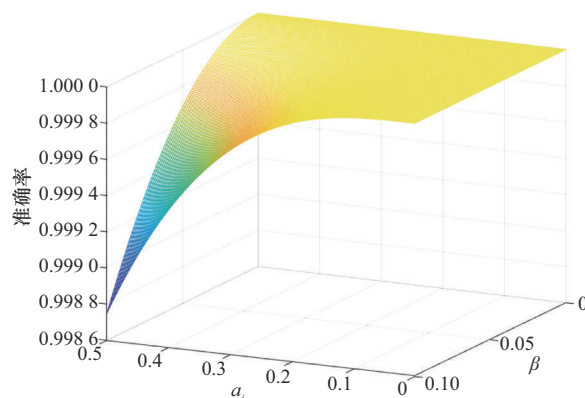
场景	概率
无执行体受攻击	0.125
E_1 受攻击	0.125
E_2 受攻击	0.125
E_3 受攻击	0.125
E_1, E_2 同时受攻击 (不协同)	0.112 5
E_1, E_2 同时受攻击 (协同)	0.012 5
E_2, E_3 同时受攻击 (不协同)	0.112 5
E_2, E_3 同时受攻击 (协同)	0.012 5
E_1, E_3 同时受攻击 (不协同)	0.112 5
E_1, E_3 同时受攻击 (协同)	0.012 5
E_1, E_2, E_3 同时受攻击 (不全部协同)	0.123 75
E_1, E_2, E_3 同时受攻击 (全部协同)	0.001 25

利用表6中的场景概率对表5中的数据进行概率修正,可以计算出IDHES在MCU上的实际检测准确率为:

$$ACC_{MCU} = \frac{TP + TN}{TP + FP + FN + TN} = \frac{6\,991 + 1\,000}{8\,000} = 99.88\%$$

该结果超过了大部分传统机器学习和深度学习IDS的检测准确率^[1],且非常接近其理论检测准确率,符合1.2节对IDHES的推论,说明只有当同步效率未达到100%时,系统会出现FP误判,只有当所有执行体产生协同攻击行为时,IDHES才会对系统的受攻击状态产生FN误判。IDHES检测方法不依赖任何先验知识。

在现实情况下,DHR系统中单个执行体被攻击成功的概率达不到50%,两个执行体间协同漏洞存在概率也远远达不到10%。将 α_i 和 β 的值逐步减少,然后对表5中的数据进行概率修正,得到了IDHES在更低 α_i 和 β 值情况下的检测准确率(如图8所示)。可以看出,随着 α_i 和 β 的值逐步降低,IDHES的检测准确率逐步趋近于1。

图8 IDHES(MCU)在不同 α_i 和 β 值下的检测准确率Figure 8 IDHES(MCU) detect accuracy with various α_i and β value

5 结束语

本文基于内生安全和DHR构造,提出了一种基于异构执行体同步控制的入侵检测模型——IDHES,将应用程序内部事件转换为外部事件对异构执行体的目标功能进行同步控制,达到提高入侵检测准确率和精确率的目的。IDHES利用自身DHR构造带来的动态性、异构性、冗余性优势,脱离传统方式对先验知识的依赖,检测攻击前不需要提前训练,可以实时检测已知和未知类型的攻击。构建数学模型分析IDHES的检测准确率完全取决于对异构执行体目标功能进行同步的效率,以及所有异构执行体被协同攻击的效率(见式(9))。本文提出了一种基于内部定时器事件转换为外部消息事件的目标功能同步方法,并且构建了DHR架构的MCU原型验证系统,实现了异构Modbus-RTU协议的同步,利用白盒插桩的方式进行了攻击模拟,测试了IDHES的检测准确率。测试结果显示,IDHES的检测准确率符合式(9)提出的推论。最后绘制了在不同 α_i 和 β 组合值下的IDHES的检测准确率图,为相应情况提供参考。

在未来工作中,将会对IDHES模型进行进一步细化,加入DHR架构对单个执行体受攻击概率的影响,以及执行体调度操作对后续检测的影响等因素,从系统角度构建动态演化模型,更完整地模拟DHR架构模型从入侵检测到异常行为处理这一变化过程。

参考文献:

- [1] SHARMA S, KUMAR A, RATHORE N S, et al. Intrusion detection and prevention systems in industrial IoT network[J]. Sadhana-Academy Proceedings in Engineering Sciences, 2024, 49(3): 1-13.
- [2] 龚俭, 臧小东, 苏琪, 等. 网络安全态势感知综述[J]. 软件学报, 2017, 28(4): 1010-1026.
GONG J, ZANG X D, SU Q, et al. Survey of network security situation awareness[J]. Journal of Software, 2017, 28(4): 1010-1026.
- [3] BRIDGES R A, GLASS-VANDERLAN T R, IANNAcone M D, et al. A survey of intrusion detection systems leveraging host data[J]. ACM Computing Surveys, 2020, 52(6): 1-35.
- [4] HAMED T, DARA R, KREMER S C. Network intrusion detection system based on recursive feature addition and bigram technique[J]. Computers & Security, 2018, 73(0): 137-155.
- [5] AJAYI O, GANGOPADHYAY A, ERBACHER R F, et al. Developing cross-domain host-based intrusion detection[J]. Electronics, 2022, 11(21): 3631.
- [6] ZHOU W D, LEI S W, XIA C H, et al. LDA-ID: an lda-based framework for real-time network intrusion detection[J]. China Communications, 2023, 20(12): 166-181.
- [7] WU J S, DAI H, KENT K B, et al. Open set dandelion network for iot intrusion detection[J]. ACM Transactions on Internet Technology, 2024, 24(1): 1-26.
- [8] SOKOL P, STAŇA R, GAJDOŠ A, et al. Network security situation awareness forecasting based on statistical approach and neural networks[J]. Logic Journal of the IGPL, 2023, 31(2).
- [9] WANG M L, SONG G X, YU Y, et al. The current research status of ai-based network security situational awareness[J]. Electronics, 2023, 12(2309): 2309.
- [10] WU L W, YANG C. Research on hierarchical network security situational awareness data fusion method in big data environment[J]. Applied Mathematics and Nonlinear Sciences, 2024, 9(1).
- [11] ZHANG J W, FENG H M, LIU B, et al. Survey of technology in network security situation awareness[J]. Sensors, 2023, 23(5): 2608.
- [12] 罗智勇, 宋伟伟, 张文博, 等. 基于 Markov 攻击图和博弈模型的区块链安全态势感知方法[J]. 电子与信息学报, 2023, 45(4): 1374-1382.
LUO Z Y, SONG W W, ZHANG W B, et al. Blockchain security situational awareness method based on Markov attack graph and game model[J]. Journal of Electronics & Information Technology, 2023, 45(4): 1374-1382.
- [13] DATTA N. A hybrid intrusion detection system using k -means clustering and J48 classification[J]. International Journal of Engineering Research in Computer Science and Engineering, 2018, 5(4).
- [14] 邬江兴. 网络空间内生安全发展范式[J]. 中国科学: 信息科学, 2022, 52(2): 189-204.
WU J X. Development paradigms of cyberspace endogenous safety and security[J]. Sci Sin Inform, 2022, 52(2): 189-204.
- [15] 邬江兴, 邹宏, 薛向阳, 等. 内生安全赋能网络弹性的构想、方法与策略[J]. 中国工程科学, 2023, 25(6): 106-115.
WU J X, ZOU H, XUE X Y, et al. Cyber resilience enabled by endogenous security and safety: vision, techniques, and strategies[J]. Strategic Study of CAE, 2023, 25(6): 106-115.
- [16] 宋克, 刘勤让, 魏帅, 等. 基于拟态防御的以太网交换机内生安全体系结构[J]. 通信学报, 2020, 41(5): 18-26.
SONG K, LIU Q R, WEI S, et al. Endogenous security architecture of ethernet switch based on mimic defense[J]. Journal on Communications, 2020, 41(5): 18-26.
- [17] CHEN H S, HAN X T, ZHANG Y Y. Endogenous security formal definition, innovation mechanisms, and experiment research in industrial internet[J]. Tsinghua Science and Technology, 2024, (2): 492-505.
- [18] 朱维军, 郭渊博, 黄伯虎. 动态异构冗余结构的拟态防御自动机模型[J]. 电子学报, 2019, 47(10): 2025-2031.
ZHU W J, GUO Y B, HUANG B H. A mimic defense automaton model of dynamic heterogeneous redundancy structures[J]. Acta Electronica Sinica, 2019, 47(10): 2025-2031.
- [19] 马海龙, 王亮, 胡涛, 等. 网络空间拟态防御发展综述: 从拟态概念到“拟态+”生态[J]. 网络与信息安全学报, 2022, 8(2): 15-38.
MA H L, WANG L, HU T, et al. Survey on the development of mimic defense in cyberspace: from mimic concept to “mimic+” ecology[J]. Chinese Journal of Network and Information Security, 2022, 8(2): 15-38.
- [20] 王敏, 付文昊, 王宝通, 等. 基于演化博弈的拟态防御策略优化[J]. 计算机应用研究, 2024, 41(2): 576-581.
WANG M, FU W H, WANG B T, et al. Defense strategy optimization of cyber mimic defense based on evolutionary game theory[J]. Application Research of Computers, 2024, 41(2): 576-581.
- [21] HE J J, YUAN Y L, LIANG S C, et al. A Function-Aware Mimic Defense Theory and Its Practice[J]. China Communications, 2024, 21(8): 192-210.
- [22] 朱正彬, 刘勤让, 刘冬培, 等. 拟态多执行体调度算法研究进展[J]. 通信学报, 2021, 42(5): 179-190.
ZHU Z B, LIU Q R, LIU D P, et al. Research progress of mimic multi-execution scheduling algorithm[J]. Journal on Communications, 2021, 42(5): 179-190.

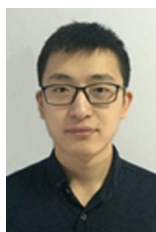
[作者简介]



于洪 (1988), 女, 四川资阳人, 信息工程大学副研究员, 主要研究方向为网络空间内生安全、拟态防御。



朱正彬（1996），男，湖北荆门人，信息工程大学博士生，主要研究方向为软件定义网络、网络安全。



郭威（1990），男，北京人，信息工程大学副研究员，主要研究方向为计算机体系结构、人工智能与大数据、内生安全。



魏帅（1984），男，河南邓州人，信息工程大学副教授，主要研究方向为软件定义晶上系统、网络空间内生安全。



兰巨龙（1962），男，河北张家口人，信息工程大学教授、博士生导师，主要研究方向为信息通信网络、主动防御技术、高效能计算。