安全和能量关键的分布式协作任务调度

江 维[1],常政威[2],桑 楠[1],熊光泽[1]
(1.电子科技大学计算机科学与工程学院,四川成都 611731;2.四川电力试验研究院,四川成都 610072)

摘 要: 传统分布式任务调度严重地忽略了任务的能耗和安全因素,不适用于安全关键分布式嵌入式系统.针对安全和能量关键的分布式协作应用,提出了一种安全感知和能量感知的任务映射调度算法 SEATMS.作为一种多项式复杂度的启发式算法,SEATMS 能够在满足协作应用的实时需求和能耗约束前提下,最大程度地降低系统安全风险.和同类算法相比,所提算法在实时确保、能量预算确保和低安全风险方面有明显优势.

关键词: 分布式嵌入式系统; 协作任务; 能量消耗; 安全风险; 映射调度
中图分类号: TP316 **文献标识码:** A **文章编号:** 0372-2112 (2011) 04-0757-06

Scheduling for Security and Energy-Critical Distributed Collaborative Tasks

JIANG Wei[1], CHANG Zheng-wei[2], SANG Nan[1], XIONG Guang-ze[1]
(1. *School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan* 611731, *China*;
2. *Sichuan Electric Power Test and Research Institute, Chengdu, Sichuan* 610072, *China*)

Abstract: Due to the ignorance of energy and security factors, traditional scheduling policies are not suitable for security-critical distributed embedded systems. A Security and Energy-Aware Task Mapping and Scheduling (SEATMS) algorithm is proposed for security and energy-critical distributed collaborative applications. The proposed algorithm can strive to reduce the security risk while without sacrificing the real-time and energy constraints, and was proved to be an algorithm with polynomial time complexity. Simulation results show superiority of SEATMS on real-time guarantee, energy budget guarantee, and low security risk.

Key words: distribute embedded system; collaborative task; energy consumption; security risk; mapping and scheduling

1 引言

随着嵌入式处理器、无线网络、传感器等技术飞速发展,网络化嵌入式系统越来越多的应用于任务关键系统.许多任务关键应用需要通过分配在多个计算节点上的一组任务共同协作完成,并且需要访问、存储和管理安全敏感的数据[1,2].执行安全关键分布式任务的网络化嵌入式系统称之为安全关键分布式嵌入式系统(Security-Critical Distributed Embedded System, SCDES).SCDES 面临着网络化带来的严重安全威胁,比如机密性、完整性和可用性的攻击,如何降低应用于 SCDES 的分布式协作任务(Collaborative Task)的安全风险成为重要挑战[3].另外,电池技术与集成电路技术的不对称发展,导致能量成为制约 SCDES 快速发展的关键因素.大多嵌入式系统采用电池供电方式,过多过快地消耗电池能量都将导致嵌入式设备过早失效,进而引发分布式协作应用的失效,最终可能产生不堪设想的后果[4,5].因此,研究能量有效的安全关键分布式系统是势在必行和十分有意义的工作.

大多任务关键分布式协作应用属于实时应用,协作任务的运行要求在限定的时间内产生正确的计算结果.大量文献研究了分布式环境下的实时应用[6,7],提出了许多高水平成果.近年来,为增强任务关键系统的安全性,安全感知的单机和分布式任务调度得到密切关注[3,8-11].面向网络化嵌入式系统,许多学者研究了分布式协作应用的能量有效任务映射和调度问题[12~14].然而,上述研究没有同时考虑 SCDES 的安全和能耗因素,并且忽略了异构分布式环境下协作任务的映射调度问题,因而不能很好地应用于 SCDES 系统.

本文研究面向 SCDES 的分布式实时任务调度机制,着重解决能量和实时约束的安全关键分布式协作任务的映射调度问题.本文主要贡献在于,建立了安全风险驱动的系统模型,设计了多项式复杂度的分布式调度算法,既确保协作任务的实时需求和能耗约束,又可最大程度地降低系统安全风险.

收稿日期:2010-04-07;修回日期:2010-08-26
基金项目:国家自然科学基金(No.61003032,No.61003230);核心电子器件、高端通用芯片及基础软件产品专项资金(No.2009ZX01039-003-001-002);中央高校基本科研业务专项资金(No.ZYGX2009J062)

2 系统模型

2.1 SCDES 体系结构

本文考虑异构安全关键分布式嵌入式系统. SCDES 由多个异构嵌入式节点构成,节点面临着多类安全威胁(如窃听、篡改等)[3,7,10],其体系结构表示为 SCDES = (P, CP, SP). $P_u \in P$ 表示第 u 个嵌入式计算节点,向量 $CP = (cp_1, cp_2, \cdots, cp_M)$表示 M 个计算节点的能量消耗率(即功耗), $SP_u = (sp_u^1, sp_u^2, \cdots, sp_u^R) \in SP$ 表示节点 P_u 提供 R 种安全保护的能力,比如 sp_u^2 表示 P_u 可为第 2 类安全威胁提供 sp_u^2 级别的安全服务.本文考虑非动态电压/频率支持的节点,节点功耗波动较小,因而假定节点执行任务时的能量消耗率固定.另外,本文考虑任务间没有数据交换,或只有控制信号的交互,因而忽略节点间的通信开销和通信安全问题.

2.2 安全关键分布式协作任务模型

分布式协作任务通常用有向无环图(Directed A-cyclic Graph, DAG)来表示. 安全感知协作任务定义为 SADAG = (T, E),顶点 $\tau_i \in T$ 表示第 i 个子任务,有向边 $e_{ij} = (\tau_i, \tau_j) \in E$ 表示子任务 τ_i 与 τ_j 之间的通信依赖或控制依赖, τ_i 为 τ_j 的前驱任务.安全感知子任务表示为 $\tau_i = (C_i, d_i, SD_i, W_i)$. $C_i = (c_i^1, c_i^2, \cdots, c_i^M)$为任务 τ_i 的执行时间向量,即表示该任务在不同节点上的基本执行时间; d_i 表示任务 τ_i 需要安全保护的数据大小; $SD_i = (sd_i^1, sd_i^2, \cdots, sd_i^R)$表示执行任务 τ_i 所期望的安全级别向量; W_i 则表示任务 τ_i 的安全损失值,即因安全失效而产生的后果.图 1 例示了一个 SADAG 和 SCDES,及它们之间的映射调度关系.

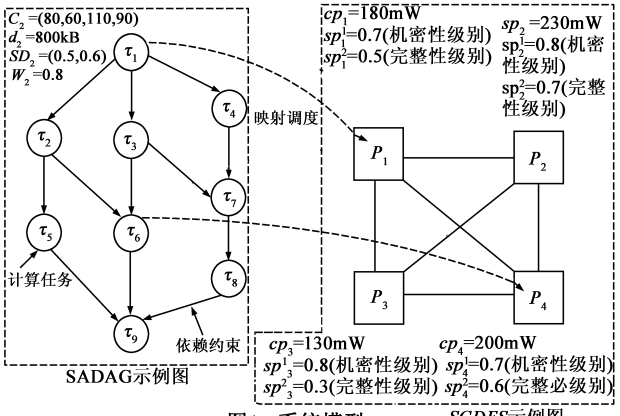


图1 系统模型

2.3 任务的运行时间和能耗开销

执行额外安全服务必然能够增强任务的安全性,但同时将导致其他性能的牺牲.比如任务运行 IDEA 加密服务来保护数据,则必定导致任务的执行时间延迟和能耗开销.

假定任务 τ_i 分配到节点 P_u 上执行,并且要求提供 R 种安全服务,则 τ_i 的执行时间为基本执行时间和所有安全服务的执行时间之和,即

$$Exe(\tau_i, P_u) = c_i^u + \sum_{r=1}^{R} d_i \cdot \theta_u^r(sp_u^r), \text{where } sp_u^r \in SP_u \tag{1}$$

其中, c_i^u 表示 τ_i 在节点 P_u 上的基本执行时间, θ_u^r 表示节点 P_u 提供第 r 种安全服务的单位安全开销函数, $\theta_u^r(sp_u^r) \cdot d_i$ 则表示以 sp_u^r 安全级别处理大小为 d_i 的安全数据的时间开销.

安全感知子任务的能耗表示为节点能量消耗率(功耗)与任务总体执行时间的乘积,即任务 τ_i 分配到 P_u 上执行的能耗表示为

$$En(\tau_i, P_u) = cp_u \cdot Exe(\tau_i, P_u) = cp_u \cdot (c_i^u + \sum_{r=1}^{R} d_i \cdot \theta_u^r(sp_u^r)) \tag{2}$$

2.4 安全风险建模

为定量描述安全关键任务的安全质量,引入安全风险评估准则,建立面向 SCDES 的任务安全风险模型.

当 τ_i 分配到节点 P_u 运行时, τ_i 由第 r 种安全威胁导致的失效概率表示为

$$P_{risk}(\tau_i, P_u, sp_u^r) = \begin{cases} 1 - \exp[-\lambda(sd_i^r - sp_u^r)], & \text{if } sd_i^r > sp_u^r \\ 0, & \text{if } sd_i^r \leqslant sp_u^r \end{cases} \tag{3}$$

其中 λ 是安全风险系数.等式(3)表明,当 P_u 提供的安全级别大于等于 τ_i 的安全级别需求时,则不会产生由安全因素导致的失效;反之,则 τ_i 有产生安全失效的可能,并且安全级别差别越大,失效概率就越高.

进而,任务 τ_i 由所有安全威胁导致的失效概率定义为

$$P_{risk}(\tau_i, P_u) = 1 - \prod_{r=1}^{R} (1 - P_{risk}(\tau_i, P_u, sp_u^r)) \tag{4}$$

于是,当 τ_i 分配到节点 P_u 运行时, τ_i 的安全风险(SR)为

$$SR(\tau_i, P_u) = W_i * P_{risk}(\tau_i, P_u) = W_i * [1 - \prod_{r=1}^{R} (1 - P_{risk}(\tau_i, P_u, sp_u^r))] \tag{5}$$

3 系统目标

本文的任务映射调度目标为:在满足能量约束、实时约束和分配约束前提下,将安全感知协作任务分配到相应的节点中执行,使得协作任务的系统安全风险最小.给定协作任务 SADAG = (T, E)和安全关键分布式系统 SCDES = (P, CP, SP),本文定义 $H = (h_1, h_2, \cdots, h_N)$为 SADAG 到 SCDES 的一种任务映射调度方案. $h_i = (\tau_i, P_u, St_i, Ft_i)$表示任务 τ_i 的映射和调度方案,其

中 P_u 表示 τ_i 的执行节点,St_i 表示 τ_i 的开始执行时间,Ft_i 表示 τ_i 的结束时间.于是,本文的目标为最小化以下等式:

$$OSR(H)=\sum_{\tau_i\in T}SR(\tau_i,P_u)=\sum_{\tau_i\in T}W_i*\left[1-\prod_{r=1}^{R}(1-P_{risk}(\tau_i,P_u,sp_u^r))\right] \quad (6)$$

$$\text{S.T.}\begin{cases}Finish(H)=\max\limits_{\tau_i\in T}\{Ft_i\}\leqslant Deadline\\ OEn(H)=\sum\limits_{\tau_i\in T}En(\tau_i,P_u)\leqslant En_{budget}\end{cases} \quad (7)$$

OSR 表示系统安全风险,$Finish(H)$表示在方案 H 下 SADAG 的结束时间,$Deadline$ 表示 SADAG 的死限约束;OEn 表示系统能量消耗,En_{budget}表示 SADAG 的能量预算.

4　分布式协作任务映射调度

本文目标旨在满足实时约束和能耗约束的前提下,最小化分布式协作应用的安全风险.由于分布式协作任务的映射调度是一个 NP 完全的问题[12],因此需要寻求启发式解决方案.基于高效率和较低复杂度 BU 算法[15],本文提出了一种安全和能量感知的实时任务映射调度算法(Security and Energy Aware Task Mapping and Scheduling,SEATMS).如图 2 所示,SEATMS 算法是三阶段启发式算法,它由任务定级分组(RANKING)、任务映射(MAPPING)和任务调度(SCHEDULING)三个阶段组成.任务定级分组阶段主要完成子任务映射级别的设置和子任务的分组;任务映射阶段则负责 SADAG 任务的安全优化分配,即依据任务映射级别将任务分配到相应的节点,使得系统安全风险最小;在任务调度阶段,SEATMS 确定各子任务的最终执行时序.

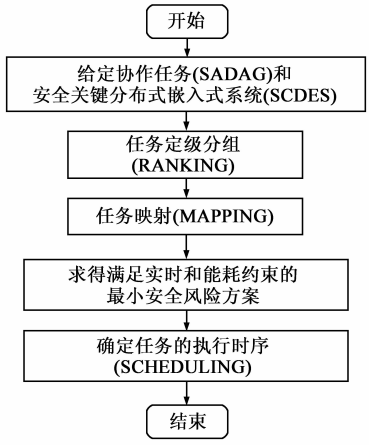


图2　SEATMS算法流程

4.1　任务定级分组

在 RANKING 阶段,首先,为 SADAG 的全部子任务分配相应的静态映射级别;然后,将子任务按级别分组,即映射级别相同的子任务为一组.

任务 τ_i 的静态级别 $L(\tau_i)$定义为

$$\begin{cases}L(\tau_i)=0, & \text{if}\quad pred(\tau_i)=\Phi\\ L(\tau_i)=\max\limits_{\tau_j\in pred(\tau_i)}(L(\tau_j))+1, & \text{else}\end{cases} \quad (8)$$

任务定级分组流程如图 3 所示.

```
Function RANKING (SADAG)
(01)For i = 1 to N
(02)  { L(τ_i) = 0, if pred(τ_i) = Φ
      { L(τ_i) = max_{τ_j ∈ pred(τ_ii)} (L(τ_j)) + 1, else
(03)  If L(τ_i) = k
(04)      GL(k) = GL(k) ∪ {τ_i}
(05)  EndFor
(06)  Return(GL = {GL(1), GL(2), ···, GL(maxL(τ_i))})
```
图 3　任务定级分组

4.2　任务映射

在 MAPPING 阶段,我们采取自底向上(Bottom Up)方式,从最高映射级别到最低映射级别的顺序对所有子任务实施优化分配.MAPPING 的核心思想是基于最小成本函数的优化分配.对每个任务 – 节点分配组合(τ_i,P_u),其分配成本函数 $\text{COST}(\tau_i,P_u,\alpha,\beta,\gamma)$表示任务 τ_i 分配到节点 P_u 运行的综合成本.其中,α,β,γ 分别表示安全风险、能耗和运行时间的加权因子.

定义 1　$Exe_{\max}(\tau_i)$表示任务 τ_i 分配到所有节点上的最大运行时间开销,即

$$Exe_{\max}(\tau_i)=\max\{Exe(\tau_i,P_u)\mid P_u\in P\}$$

定义 2　$NExe(\tau_i,P_u)$表示任务 τ_i 分配到节点 P_u 上的归一化运行成本,即

$$NExe(\tau_i,P_u)=Exe(\tau_i,P_u)/Exe_{\max}(\tau_i)$$

定义 3　$En_{\max}(\tau_i)$表示任务 τ_i 分配到所有节点上的最大能耗开销,即

$$En_{\max}(\tau_i)=\max\{En(\tau_i,P_u)\mid P_u\in P\}$$

定义 4　$NEn(\tau_i,P_u)$表示任务 τ_i 分配到节点 P_u 上的归一化能耗成本,即

$$NEn(\tau_i,P_u)=En(\tau_i,P_u)/En_{\max}(\tau_i)$$

定义 5　$SR_{\max}(\tau_i)$表示任务 τ_i 分配到所有节点上的最大安全风险,即

$$SR_{\max}(\tau_i)=\max\{SR(\tau_i,P_u)\mid P_u\in P\}$$

定义 6　$NSR(\tau_i,P_u)$表示任务 τ_i 分配到节点 P_u 上的归一化安全成本,即

$$NSR(\tau_i,P_u)=SR(\tau_i,P_u)/SR_{\max}(\tau_i)$$

这样,分配任务 τ_i 到节点 P_u 的综合成本函数定义为

$$COST(\tau_i,P_u,\alpha,\beta,\gamma)=\alpha\cdot NExe(\tau_i,P_u)+\beta\cdot NEn(\tau_i,P_u)+\gamma\cdot NSR(\tau_i,P_u) \quad (9)$$

$$\text{S.T.}\begin{cases}\alpha+\beta+\gamma=1\\ 0\leqslant\alpha,\beta,\gamma\leqslant1\end{cases} \quad (10)$$

MAPPING 伪代码如图 4 所示.首先初始化系统安全风险值为无穷大,并循环调整成本函数的加权因子(α,β)值(1 ~ 3 行);接着自底向上,从最高级别到最低级别依次映射各子任务组的所有任务(4 ~ 16 行).在子任务组中,首先单独为每个任务选择成本函数值最小

的映射节点(5~9行),然后在所有低成本(任务-节点)映射组合中选择成本函数值最小的(任务-节点)组合,将该任务分配到相应的节点,并在子任务组中删除该任务(10~15行).最后,MAPPING确定满足能耗预算和实时约束的最小安全风险方案(17~19行).

```
Function MAPPING (GL, Δ, SCDES)
(1)  OSR* = ∞
(2)  For α = 0 to 1, α + = Δ
(3)   For β = 0 to 1 − α, β + = Δ
(4)    For k = max(L(τ_i)) to 0
(5)     For τ_i ∈ GL(k)
(6)      For P_k ∈ P
(7)       Calculate COST(τ_i, P_k, α, β, γ)
(8)      Find P_k* = arg min_{P_k ∈ P} [COST(τ_i, P_k, α, β, γ)]
(9)     EndFor
(10)    For τ_i ∈ GL(k)
(11)     Find
         (τ_i*, P_k*) = arg min_{T_i ∈ GL(k)} [COST(τ_i, P_k*, α, β, γ)]
(12)          GL(k) = GL(k) − τ_i*
(13)          If GL(k) = Φ
(14)           Then break
(15)        EndFor
(16)     EndFor
(17)    If { Finish(H) = Ft_N ≤ Deadline &&
           OEn(H) ≤ En_budget && OSR(H) ≤ OSR*
         }
(18)     Then OSR* = OSR(H)
(19)          H* = H
(20)   EndFor
(21)  EndFor
(22)  Return OSR* and H*
```

图4 任务映射伪代码

4.3 任务调度

在SCHEDULING阶段,依据任务映射阶段得到的安最小全风险方案,确定子任务的最终执行时序. $pred(\tau_i)$表示任务 τ_i 的前驱任务集,$TP(P_u)$表示由已分配到 P_u,但不包括任务 τ_i 的前驱任务的其他子任务组成的集合.任务调度函数伪代码如图5所示.不失一般性地,子任务 τ_i 的开始时间是该任务所有前驱任务最晚结束时间与已分配到 P_u 的其他子任务的最晚结束时间之和;子任务 τ_i 的结束时间则为该任务开始时间与基本执行时间、安全服务时间的总和.

```
Function SCHEDULING (OSR*, H*)
(1)  For i = 1 to N
(2)  St_i = max_{τ_k ∈ pred(τ_i)} (Ft_k) + max_{τ_j ∈ TP(P_u) && j < i} (Ft_j)
(3)  Ft_i = St_i + c_i^u + Σ_{r=1}^{R} d_i · θ_u^r(sp_u^r)
(4)  EndFor
(5)  Return H* = {h_1, h_1, ⋯, h_N}
```

图5 任务调度伪代码

4.3 算法复杂度

定理1 SEATMS是多项式时间复杂度的算法,即 $O(\text{SEATMS}) = (1/\Delta)^2(N \cdot M + \frac{N^2}{k})$. Δ 表示适应度函数中加权参数的离散程度,k 为SADAG的所有子任务的映射级别总数,M 为SCDES的节点数.

证明:RANKING和SCHEDULING的复杂度均为 $O(N)$.在MAPPING函数中,外部For循环(第2和3行)的复杂度为 $O(1/\Delta)$.假定SADAG协作应用中子任务的最大级别为 k,则每个任务分组的平均子任务数为 N/k.于是,第4行的For循环复杂度为 $O(k)$,第5和10行的复杂度为 $O(N/k)$.第6和8行的复杂度均为 $O(M)$;第11行需要 $O(N/k)$时间求解最小成本值的(任务-节点)组合.这样,SEATMS的总体时间复杂度为

$$O(SEATMS) = O(N) + O((\frac{1}{\Delta})^2 k[\frac{N}{k}(O(M) + O(M)) + \frac{N}{k} * O(\frac{N}{k})]) + O(N)$$
$$= O(\frac{1}{\Delta^2}(N \cdot M + \frac{N^2}{k}))$$

5 仿真实验

本文通过仿真实验来验证所提算法的有效性.为方便比较,引入了SecLIST和BEATA[13]算法.SecLIST算法是考虑任务安全开销的列表(LIST)调度算法[16].除系统安全风险和系统能量消耗外,引入调度长度评价指标,其表示SADAG中所有子任务的最晚结束时间.本文通过两组实验来测试SEATMS算法的性能,每组仿真均独立运行三次,实验结果取平均值.SCDES的节点数范围为10~40个,节点的CPU频率和能量消耗率参照Intel Strong-ARM 1100[13]的参数产生,并假定频率范围为60~250MHz,功耗为50~300mW.SADAG由TGFF[17]随机产生,子任务数目范围为25~200,子任务的最大前趋任务数为4.SADAG任务的截止时间为500~4000ms,能量预算范围为1~5J.本文主要考虑提供机密性和完整性两类安全服务的任务[8].任务基本执行时间为50~200ms,任务安全损失值为0.1~1,需处理的安全关键数据为200kB~2MB,任务机密性和完整性安全服务的需求级别分别服从[0.5~0.8]和[0.4~0.7]范围的均匀分布.节点提供的机密性和完整性安全级别则分别服从[0.1~0.9]和[0.1~0.7]范围的均匀分布,并设置风险系数 λ 为2.

在第一组实验中,我们主要验证节点数变化对系统性能指标的影响.SCDES的节点数依次从10个到40个逐渐递增,SADAG的子任务数目固定为65个,死限和能量预算分别为1500ms和2Joule.其他参数则在约束范围内随机生成.由图6得知,SEATMS算法的风险明

显低于另外两种算法,并分别比 SecLIST 和 BEATA 降低 33.6%和 34.7%,各算法的系统安全风险是 SEATMS 最小风险的归一值.就系统能耗指标而言(图 7),SEATMS 能耗居于另外两算法之间,并且一直都能满足系统的能量预算需求.具体来说,SEATMS 平均能耗比 SecLIST 降低 29.3%,比 BEATA 增加 90.5%.从图 8 可知,算法调度长度按递增顺序为 SecLIST、SEATMS 和 BEATA.SEATMS 和 SecLIST 总是不违背 SADAG 的死限,而 BEATA 则基本上都超出了死限需求.综合图 6、7 和 8 得到,在不同节点数情况下,SEATMS 总能得到满足实时约束和能量约束的最小系统安全风险方案.

在第二组实验中,测试不同 SADAG 的死限对算法的影响.SCDES 节点数固定为 20 个,SADAG 的子任务数目固定为 100 个,能量预算为 2500mJ,SADAG 的死限从 800ms 到 3600ms 变化.其他参数同样在其约束范围内随机产生.图 9、10 和 11 分别显示了各项性能指标在不同死限约束情况下的结果,可得如下结论:①SEATMS 算法的系统安全风险同样保持着明显优势,平均安全风险分别比 SecLIST 和 BEATA 降低 47.6%和 45.1%.②SecLIST 和 BEATA 的系统安全风险、系统能耗和调度长度都变化不大.③SEATMS 算法产生的系统能耗始终低于能量预算,且 SEATMS 的平均系统能耗比 SecLIST 降低 24.9%.④SEATMS 的系统安全风险随着死限增大,越来越小,最后趋于不变.这是因为,死限越大,选择低风险节点分配任务的机会越多;当死限足够大的时候(如 3200ms、3600ms 的情况),最低风险方案的调度长度已远低于死限(图 11 所示).第二组实验表明了 SEATMS 在不同死限约束情况下的健壮性,进而再次验证了 SEATMS 是一种性能优越的安全关键分布式协作任务的映射调度算法.

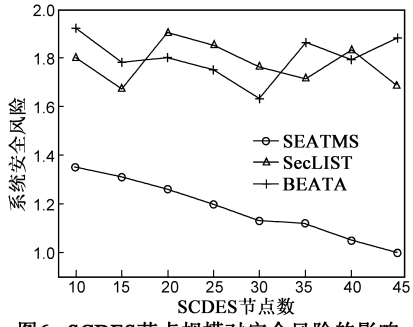


图6　SCDES节点规模对安全风险的影响

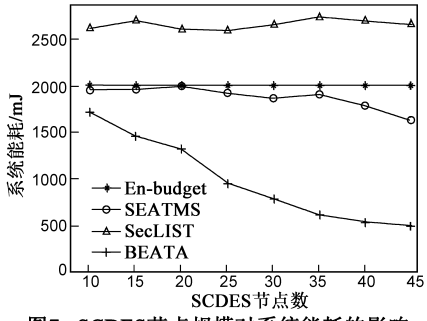


图7　SCDES节点规模对系统能耗的影响

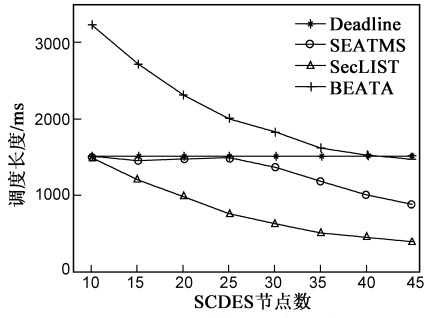


图8　SCDES节点规模对调度长度的影响

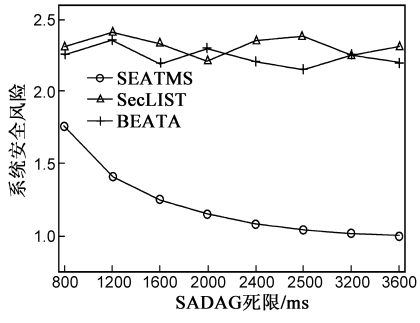


图9　SADAG死限对安全风险的影响

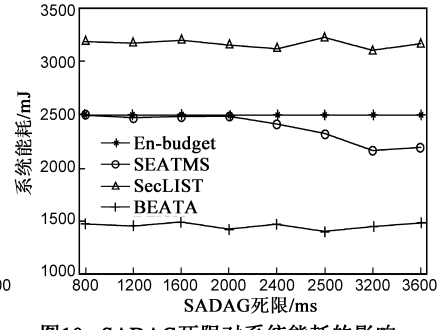


图10　SADAG死限对系统能耗的影响

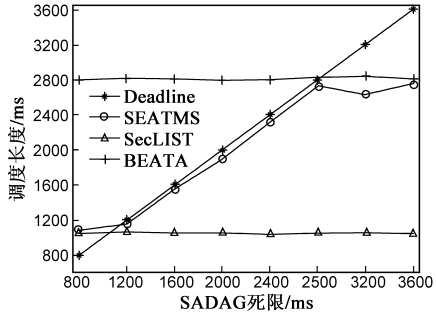


图11　SADAG死限对调度长度的影响

6　结论

针对安全关键分布式协作应用有强烈的实时和能耗需求,同时又希望最大程度降低系统安全风险的情况,本文研究了安全感知协作任务的映射调度机制.建立了异构分布式系统的任务能耗模型和面向多类安全威胁的任务安全风险模型,设计了安全和能量感知的实时任务映射调度算法 SEATMS.SEATMS 是一个三阶段启发式算法,其核心思想是自底向上为任务选择最小成本值的分配方案.SEATMS 是基于 BU 算法的改进算法,具有较低的时间开销,并且被证明为具有多项式时间复杂度.实验结果显示了本文算法的优越性.与其他算法相比,SEATMS 可维持最低的系统安全风险,并且不违背系统的实时和能量预算约束.

考虑到更切实切的系统模型,我们将在以下方面深入开展,①基于动态电压/频率调整的安全关键分布式任务调度,②通信敏感的安全关键分布式任务调度.

参考文献:

[1] S Ravi, A Raghunathan, P Kocher, et al. Security in embedded systems: design challenges[J]. ACM Transactions on Embedded Computing Systems, 2004, 3(3): 461 – 491.

[2] N R Potlapally, S Ravi, A Raghunathan, et al. A study of the energy consumption characteristics of cryptographic algorithms

and security protocols[J]. IEEE Transactions on Mobile Computing, 2006, 5(2): 128 – 143.

[3] S Song, K Hwang, Y K Kwok. Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling[J]. IEEE Transactions on Computers, 2006, 55(6): 703 – 719.

[4] 黎忠文,熊光泽,李乐民.分布式系统安全保障新体系的研究[J].电子学报,2003,31(4):564 – 568.
Z Li, G Xiong, L Li. Research on new security and safety assurance structure of distributed system[J]. Acta Electronica Sinica, 2003, 31(4): 564 – 568. (in Chinese)

[5] R Chandramouli, S Bapatla, K P Subbalakshmi, et al. Battery power-aware encryption[J]. ACM Transactions on Information and System Security, 2006, 9(2): 162 – 180.

[6] 黎忠文,等.基于防危核(壳)的安全关键硬实时系统响应时间的分析[J].电子学报,2006,34(4):647 – 652.
Z Li, et al. Response time analysis for safety-critical hard real-time systems based on safety kernel/shell scheme[J]. Acta Electronica Sinica, 2006, 34(4): 647 – 652. (in Chinese).

[7] 江维.任务关键实时系统的可信感知调度研究[D].四川成都:电子科技大学,2009.
Jiang Wei. Research on Dependability-Aware Schedulings for Mission-Critical Real-Time Systems[D]. Chengdu, Sichuan: University of Electronics Science & Technology of China, 2009. (in Chinese)

[8] T Xie, X Qin. Scheduling security-critical real-time applications on clusters[J]. IEEE Transactions on Computers, 2006, 55(7): 864 – 879.

[9] T Xie, X Qin. Performance evaluation of a new scheduling algorithm for distributed systems with security heterogeneity[J]. Journal of Parallel and Distributed Computing, 2007, 67(10): 1067 – 1081.

[10] T Xie, X Qin. Security-aware resource allocation for real-time parallel jobs on homogeneous and heterogeneous clusters[J]. IEEE Transactions on Parallel and Distributed Systems, 2008, 19(5): 682 – 697.

[11] M Lin, L Xu, L T Yang. Static security optimization for real time systems[J]. IEEE Transactions on Industrial Informatics, 2009, 5(1): 22 – 37.

[12] Y Tian, E Ekici. Cross-layer collaborative in-network processing in multihop wireless sensor networks[J]. IEEE Transactions on Mobile Computing, 2007, 6(3): 297 – 310.

[13] T Xie, X Qin. An energy-delay tunable task allocation strategy for collaborative applications in networked embedded systems [J]. IEEE Transactions on Computers, 2008, 57(3): 329 – 343.

[14] X Wang, et al. Power-aware CPU Utilization control for distributed real-time systems[A]. Proceedings of the 15th IEEE Real-Time and Embedded Technology and Applications Symposium[C]. Washington DC: IEEE, 2009. 233 – 242.

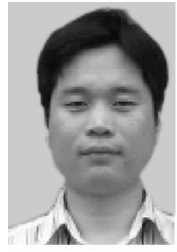
[15] S Shivle, R Castain, H J Siegel, et al. Static mapping of subtasks in a heterogeneous ad hoc grid environment[A] Proceedings of IEEE Parallel and Distributed Processing Symposium[C]. Santa Fe: IEEE, 2004. 110 – 110.

[16] G Q Liu, K L Poh, M Xie. Iterative list scheduling for heterogeneous computing[J]. Journal of Parallel and Distributed Computing, 2005, 65(5): 654 – 665.

[17] R P Dick, et al. TGFF: Task graphs for free[A]. Proceedings of the 6th International Workshop on Hardwae/software Codesign[C]. Seattle, USA: ACM, 1998. 97 – 101.

作者简介:

江 维 男,1981 年出生于四川乐山.2009 年毕业于电子科技大学,获得博士学位.现在电子科技大学从事嵌入式实时系统和可信计算的研究和教学工作.
E-mail: weijiang@uestc.edu.cn



常政威 男,1981 年出生于河南安阳.2009 年毕业于电子科技大学,获得博士学位,其后在四川电力试验研究院从事嵌入式系统、智能电网和人工智能研究工作.
E-mail: changzw@ustc.edu