基于门级信息流分析的安全体系架构设计

胡 伟，慕德俊，黄兴利，邰 瑜

(西北工业大学自动化学院 西安 710072)

【摘要】现代处理器架构中的缓存器、分支预测器等部件通常都包含难以检测的隐通道，成为攻击者入侵系统的切入点。现有方法难以有效地检测硬件相关的隐通道，从而使得这些安全漏洞往往在攻击造成严重损失后才暴露出来。该文构建了一种基于执行租赁机制的安全体系架构，以严格控制不可信执行环境的影响边界，保证不同执行环境之间的严格隔离，并采用门级抽象层次上的信息流分析方法，建立硬件架构的信息流模型，实现对硬件中全部逻辑信息流的精确度量，通过捕捉有害信息流动来检测硬件架构中潜在的安全漏洞，进而通过指令集架构的信息流模型向上层提供信息流度量能力，以实现软硬件联合安全验证。

关 键 词 隐通道； 门级信息流分析； 信息流控制； 安全体系架构； 安全漏洞
中图分类号 TP309 文献标志码 A doi:10.3969/j.issn.1001-0548.2015.03.019

Crafting Verifiably Secure Architecture Through Gate Level Information Flow Analysis

HU Wei, MU De-jun, HUANG Xing-li, and TAI Yu

(School of Automation, Northwestern Polytechnical University Xi'an 710072)

Abstract Components such as caches and branch predictors in modern processor architectures tend to include hard-to-detect covert channels, which provide a foot-holder for attackers to perform malicious activities. However, existing methods are inefficient in detecting hardware-specific covert channels. As a consequence, these security holes expose only after significant damages are inflicted. In this paper, a secure architecture based on the execution lease mechanism is built in order to tightly bound the effects of untrusted execution contexts and enforce the strict isolation of execution contexts. Further, the information flow model of the hardware architecture is constructed by using the gate level information flow analysis method, which allows the precise measurement of all digital flows in the underlying hardware and the detection of security vulnerabilities by capturing harmful flows of information. In addition, hardware/software security co-verification can be achieved with the aid of information flow measurement capability provided by the information flow model of the instruction set architecture.

Key words covert channel; gate level information flow analysis; information flow control; secure architecture; security vulnerability

为提高处理速度，现代计算机普遍采用多核结构，并引入缓存器、分支预测器等高性能部件。这些结构和部件在显著提高运算效率的同时也往往会引发一些不确定性的系统行为和不期望干扰，造成难以检测的隐通道，并成为系统的安全脆弱点。例如，文献[1]构建了一种多核处理器下共享缓存隐通道的预测模型，并准确估测了AES和Blowfish算法实现潜在的隐通道；文献[2]给出了一种通过缓存器时序攻击破解AES密码算法密钥的方法；文献[3]则提出了一种通过观测分支预测器状态来分析得到密码算法密钥的方法。大量安全事件表明：以安全脆弱点、隐通道、内嵌恶意代码等安全漏洞为切入点对系统发起攻击通常比直接破解密码算法或突破访问控制机制更为有效，并且这些安全漏洞难以检测，往往在攻击造成严重损失之后才暴露出来。

传统计算架构在设计阶段并未充分考虑安全性问题，因此现有计算系统大多采用被动防御机制来保障系统安全，如密码算法、访问控制、虚拟化和隔离技术等。然而，密码算法无法防止敏感数据在运算过程中发生泄露以及算法执行硬件平台中安全漏洞所导致的密钥泄露；访问控制可有效管理数据的分发，但无法进一步监控数据的传播，也无法防

收稿日期：2013－09－17；修回日期：2014－03－20
基金项目：国家自然科学基金(61303224)；教育部博士点基金(20126102110036)；中国博士后科学基金面上项目(2013M532081)
作者简介：胡伟(1982－)，男，博士，主要从事硬件安全、可重构计算以及嵌入式系统等方面的研究.

止旁路效应所引发的信息泄露；虚拟化和隔离技术能够防止不同执行环境之间的相互干扰，但无法消除由共享部件(如缓存器)所引发的不期望交互。并且，上述机制大多位于软件层面，无法捕捉到底层硬件中的隐通道，如硬件相关的时间隐通道(timing channel)和存储器隐通道(storage channel)。

由于信息流分析方法在漏洞检测方面具有特有的优势，本文拟构建基于门级信息流分析的安全体系架构设计与验证方法，在设计阶段即检测和消除硬件架构中潜在的安全漏洞，特别是硬件相关的隐通道，自硬件底层为系统构建一个可靠的安全基础，为解决上层软件的安全问题提供一种安全属性的度量与验证能力，进而实现软硬件联合安全验证。

1　门级信息流分析方法

文献[4]首先提出了信息流的概念，以及一种通过静态验证来强化信息流安全策略的方法。信息流分析方法通常采用格模型(lattice model)来描述信息流的信道与策略。任何一个信息流策略都可采用一个形如 $L=(SC,\sqsubseteq)$ 的安全格(security lattice)来描述。其中，SC 是客体安全类的集合；$\sqsubseteq$是定义在该安全类集上的偏序关系，规定了不同安全类之间许可的数据流向，即只允许信息在同一安全类之内或者向更高级别的安全类流动。定义函数 $L:O\to SC$ 返回对象 O 的安全类。给定任意对象 a 和 b，若 $L(a)\sqsubseteq L(b)$，则信息从 a 流向 b 是安全的。

信息流分析方法能够监控信息的传播，防止敏感信息泄露，并能有效检测以安全漏洞为切入点的攻击。然而，现有的信息流分析方法大多采用粗粒度(字或字节粒度)的标签和保守的标签传播策略，简单地以所有源操作数安全类的最小上界作为输出的安全类，未考虑输入对输出的实际影响，因此无法对系统中实际存在的信息流进行准确地度量。此外，现有信息流分析方法大多针对程序语言/编译器[5]、操作系统[6]和指令集架构等抽象层次[7-8]。在上述抽象层次上，硬件的实现细节和一些特性(如寄存器间时序)是完全透明的，因此上述方法无法检测到硬件相关的隐通道。

为有效捕捉硬件相关的时间隐通道，文献[9]提出了门级信息流跟踪(gate level information flow tracking, GLIFT)方法，从逻辑门级抽象层次准确地度量每个二进制位信息的流动，能够有效捕捉硬件电路中全部的逻辑信息流，包括显式流、隐式流以及硬件相关时间隐通道和存储器隐通道所引发的有害信息流动。该方法针对二级线性安全格 LOW $\sqsubset$ HIGH 。完整性分析中，可信$\sqsubset$非可信；机密性分析中，非保密$\sqsubset$保密。以二输入与门(AND-2)为例，表1定义了门级信息流跟踪方法下AND-2的安全标签传播规则。由表格的第二行和第二列，若AND-2的任一输入为 $(\mathrm{LOW},0)$，则其输出固定为 $(\mathrm{LOW},0)$，与另一个输入无关，而非简单地以全部输入安全类的最小上界作为输出的安全类。可见，门级信息流分析方法以信息流的定义为基础，考虑了输入对输出的实际影响，因而能够更准确地对系统中实际存在的信息流进行度量。

表1　门级信息流跟踪下AND-2的标签传播规则集

(a_t, A)	(b_t,B)			
	(LOW,0)	(LOW,1)	(HIGH,0)	(HIGH,1)
(LOW,0)	(LOW,0)	(LOW,0)	(LOW,0)	(LOW,0)
(LOW,1)	(LOW,0)	(LOW,1)	(HIGH,0)	(HIGH,1)
(HIGH,0)	(LOW,0)	(HIGH,0)	(HIGH,0)	(HIGH,0)
(HIGH,1)	(LOW,0)	(HIGH,1)	(HIGH,0)	(HIGH,1)

以 A，B 和 O 来表示与门的输入和输出，以 a_t，b_t 和 o_t 来分别表示它们的安全标签，并假定当 $a_t=0$ 时，$L(a)=\mathrm{LOW}$，当 $a_t=1$ 时，$L(a)=\mathrm{HIGH}$。则由表1可推导出AND-2的门级信息流分析逻辑如式(1)。可见，门级信息流分析方法在计算输出的安全标签时不仅考虑输入的安全标签 a_t 和 b_t，还考虑了输入的值 A 和 B 对输出 O 的实际影响，因此比传统保守的信息流分析方法更为准确。

$$o_t = Ab_t + Ba_t + a_t b_t \tag{1}$$

类似地，可进一步推导或门、非门、异或门等基本逻辑单元的信息流分析逻辑，从而构建一个基本逻辑单元信息流分析逻辑库。基于该信息流分析逻辑库，即可离散式地为复杂硬件电路中的每一个逻辑单元实例化信息流分析逻辑，从而产生复杂硬件电路的门级信息流分析逻辑，实现对硬件电路中全部逻辑信息流的精确度量。

在门级抽象层次上，所有的逻辑信息流，如显式流、隐式流以及时间隐通道所引发的信息流都具有统一的数学描述，并均以二进制位为单位显式地流动。因此，门级信息流分析方法能够通过捕捉有害信息流动来检测硬件中潜在的安全漏洞。现有工作中，文献[10-11]对门级信息流分析方法的基本理论进行了深入研究，主要包括信息流分析逻辑的性质定理、形式化描述、生成算法及复杂度理论等。文献[12-13]给出了利用门级信息流分析检测共享总线(I2C、USB和Wishbone)架构中时间隐通道的方法，并构建了一种采用门级信息流分析方法检测和消除

SoC系统中不同信任级别IP核之间有害信息流的测试框架[14]。本文研究了基于门级信息流分析的安全体系架构设计与验证方法，并提出一种基于门级信息流分析的软硬件安全联合验证方法。

2 安全体系架构设计与验证

2.1 安全体系架构设计

为防止由不同执行环境之间相互干扰所引发的有害信息流动，需严格限制不同执行环境的时间和空间边界。如图1所示，本文向现有硬件体系架构中引入执行租赁单元。该租赁架构下，每个进程都有特定的安全级别(HIGH/LOW)、执行时间定时器值(timer)和存储器边界(memory)。进程启动时，租赁单元设置PC，加载定时器和存储器边界值。进程执行时，租赁单元负责存储器访问中的安全属性和边界检查。当定时器溢出后，当前进程被挂起，直至再次调度并加载定时器时重新启动；执行租赁单元重置PC，并实施环境切换和清理，以防止执行环境的相互干扰和敏感信息泄露。

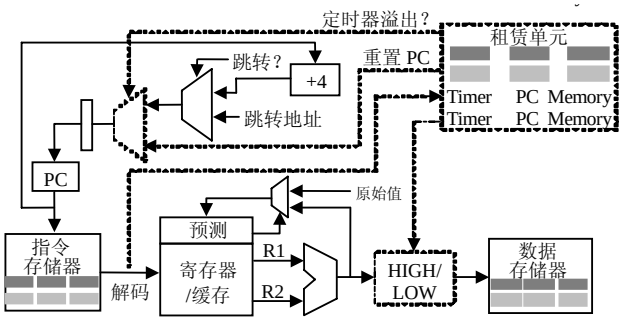

图1 执行租赁体系架构

在上述租赁架构下，当定时器溢出时，执行租赁单元将取得控制权，从而限定了进程执行的时间边界，防止恶意进程长时间占用处理器资源。若进程每次启动时采用随机或固定长度的定时器值，则可消除程序执行状态所引发的有害时间信息流(timing flow)，如不同条件分支执行时间差异所导致的信息泄露。进程在访问存储器时将受到严格的安全类型和边界检查，保证其只能访问同一或更低安全级别的数据，并禁止其越界访问共享资源。因此，不可信进程的影响边界将被严格限制在该进程的时间片和存储器资源范围之内，从而可防止由缓存等共享部件引发的信息泄露。

2.2 安全体系架构的测试与验证方法

本文采用门级信息流分析方法对执行租赁架构的安全性进行测试与验证。图2给出了安全体系架构测试与验证方法的基本原理。

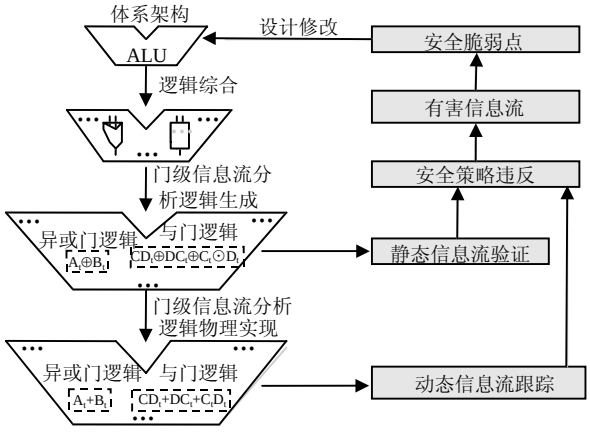

图2 安全体系架构的测试与验证方法

如图2所示，给定采用硬件设计语言(VHDL或Verilog)描述的租赁架构，首先需采用逻辑综合工具(如Synopsys Design Compiler)将设计转化为门级网表，然后，可采用文献[10-11]中所提出的算法为设计生成相应的门级信息流分析逻辑。门级信息流分析逻辑具有良好的数学形式，可在其基础上对设计中的全部逻辑信息流进行准确地度量，从而检测信息流安全策略是否被违反。若信息流安全策略被违反，则门级信息流分析逻辑将捕捉到相应的有害信息流动，通过分析有害信息流的传播路径，即可检测到设计中的安全漏洞，从而为设计修改提供指导。门级信息流分析方法能够充分利用底层硬件实现的细节信息，捕捉包括硬件相关时间隐通道在内的全部逻辑信息流，因此，上述设计与验证方法能够有效检测和消除硬件架构中潜在的安全漏洞。

此外，还可采用布尔逻辑对租赁架构的门级信息流分析逻辑进行描述，从而使得体系架构的信息流分析逻辑可随原始设计后端物理实现，在系统运行中实时地捕捉系统中的有害信息流动。当检测到有害信息流时，即可触发中断和异常处理，从而防止敏感信息泄露或关键数据被非法篡改。

3 软硬件联合安全验证方法

在上述安全架构的基础上，可进一步构建基于门级信息流分析的软硬件联合安全验证方法。如图3所示，该方法从逻辑门级抽象层次开始构建硬件的信息流模型，进而构造功能单元、控制逻辑、存储器的信息流模型，并进一步建立指令集架构的信息流模型。指令集架构的信息流模型可对软件中每条指令所包含的信息流进行准确描述，从而可为上层软件提供一种信息流度量能力。在上述验证方法中，信息流度量能力从逻辑门级抽象层次向上传递至上

层软件，而系统的安全属性(如保密性和完整性)则向下传递至硬件层面得到验证。

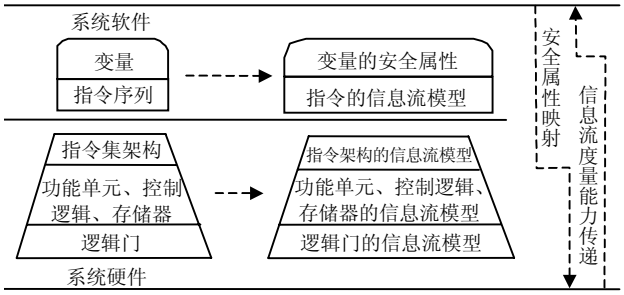

图3　基于门级信息流分析的软硬件联合安全验证方法

4　实验结果

4.1　仿真实验

本文以Trsut-Hub测试基准AES为例来验证门级信息流分析方法在检测硬件安全漏洞方面的有效性。如图4所示，当木马激活信号Tj_Trig触发后，木马程序会通过Antenna信号泄露密钥。此时，仅通过观测Antenna信号的值无法发现其隐含了密钥信息，但门级信息流分析逻辑信号Antenna_t置位(Antenna_t=1)区域则显示密钥流向了Antenna，即准确捕捉到了密钥泄露。

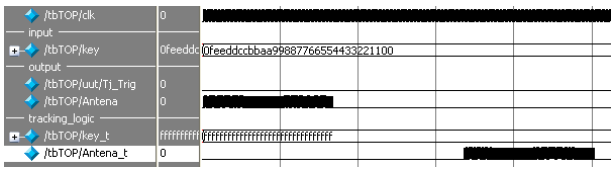

图4　门级信息流分析方法用于硬件木马分析

本文进一步以OpenCores测试基准RSA为例来验证所提出的安全架构在消除硬件相关隐通道方面的有效性。RSA密码算法的流程受密钥控制。密钥当前位分别为逻辑‘1’和逻辑‘0’时，算法所需执行的操作不同，完成操作所需的时间也相应存在差异，这将导致一个硬件相关的时间隐通道，并可引发密钥泄露[15]。本文通过状态机控制RSA密码算法核的执行，并引入计时器timer1和timer2分别对引入安全架构前后的算法执行时间进行测量，以模拟时序分析攻击，同时为RSA密码算法核产生门级信息流分析逻辑，通过仿真实验分析密钥泄露情况。仿真实验结果如图5所示。

由图5可知，密钥key的信息流分析逻辑key_t = FFFFFFFF，表征密钥属于敏感信息。未引入租赁架构时，timer1测量到的算法执行时间依赖于密钥，因此，仿真结果中timer1的信息流分析逻辑timer1_t = FFFFFFFF。在本文所提出的租赁架构下，算法分多个时间片执行，每个时间片的长度是随机的，与密钥无关，因此，仿真结果中timer2的信息流分析逻辑timer2_t = 00000000。仿真实验表明：未引入租赁架构时，算法执行上的延迟差异会造成时间隐通道，导致密钥泄露，而本文所提出的安全架构可消除该时间隐通道。

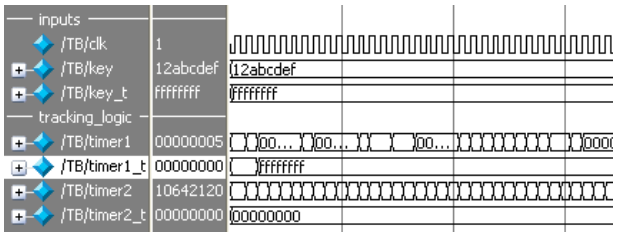

图5　引入租赁架构前后的密钥泄露情况分析

4.2　设计复杂度分析

为对门级信息流分析方法用于动态信息流跟踪时的设计复杂度进行评估，本文选用IWLS测试基准alu2，alu4和DES以及Trust-Hub测试基准PIC16F84，MC8051和AES对信息流分析逻辑的面积和延迟进行了分析。由归一化平均值可见，信息流分析逻辑(不包含原始设计)平均会引入2.69倍的额外面积开销和0.93倍的额外延迟开销。上述结果反映了将体系架构的门级信息流分析逻辑用于动态信息流安全监控时的面积和性能开销。

表2　门级信息流分析逻辑复杂度分析

测试向量	原始设计		信息流分析逻辑	
	面积/um^2	延迟/ns	面积/um^2	延迟/ns
alu2	1 591	1.00	5 357	1.71
alu4	3 060	1.08	11 585	2.17
DES	14 188	0.95	47 614	1.95
PIC16F84	19 416	0.29	64 282	0.50
MC8051	69 457	5.57	245 011	11.4
AES	1 928 226	0.33	9 257 379	0.68
归一化平均	1.00	1.00	3.69	1.93

由表2可知，门级信息流分析逻辑用于动态信息流安全监控时将带来较高的面积和性能开销。在实际应用中，可对设计进行安全划分，且仅需为安全关键模块附加门级信息流分析逻辑。此外，门级信息流分析方法还可用于静态信息流安全验证，当验证完成后，即可将额外的信息流分析逻辑移除，从而避免额外的设计开销。

5　结 束 语

门级信息流分析方法能够从硬件电路层面上捕捉系统中全部的逻辑信息流，特别是硬件相关时间隐通道所导致的敏感信息泄露。本文构建了一种基于执行租赁机制的安全体系架构，并给出了一种基

于门级信息流分析的安全体系架构验证方法。该方法能够自硬件底层为系统搭建一个可靠的安全基础，并可向上层软件提供一种信息流度量与验证能力，从而实现软硬件联合安全验证。

参 考 文 献

[1] DOMNITSER L, ABU-GHAZALEH N, PONOMAREV D. A predictive model for cache-based side channels in multicore and multithreaded microprocessors[C]//The 5th International Conference on Mathematical Methods, Models and Architectures for Computer Network Security (MMM-ACNS'10). Berlin, Heidelberg: Springer-Verlag, 2010: 70-85.

[2] BERNSTEIN D J. Cache-timing attacks on aes[R]. Chicago, USA: University of Illinois at Chicago, 2005.

[3] JEAN-PIERRE O A, SEIFERT J P, KOC C K. Predicting secret keys via branch prediction[C]//The Cryptographers Track at the RSA Conference. Berlin, Heidelberg: Springer-Verlag, 2007: 225-242.

[4]DENNING D E. Cryptography and data security[M]. Boston, MA, USA: Addison-Wesley Longman Publishing Co Inc, 1982.

[5] SABELFELD A, MYERS A. Language-based information-flow security[J]. IEEE Journal on Selected Areas in Communications, 2003, 21(1): 5-19.

[6] KROHN M, YIP A, BRODSKY M, et al. Information flow control for standard os abstractions[C]//The 21st ACM SIGOPS Symposium on Operating Systems Principles (SOSP'07). New York, USA: ACM, 2007: 321-334.

[7] SUH G E, LEE J W, ZHANG D, et al. Secure program execution via dynamic information flow tracking[C]//The 11th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-XI). New York, USA: ACM, 2004: 85-96.

[8] NEWSOME J, SONG D. Dynamic taint analysis for automatic detection, analysis, and signature generation of exploits on commodity software[C]//The 12th Annual Network and Distributed System Security Symposium (NDSS'05). San Diego, CA, USA: [s.n.], 2005.

[9] TIWARI M, WASSEL H W, MAZLOOM B, et al. Complete information flow tracking from the gates up[C]//The 14th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'09). New York, USA: ACM, 2009: 109-120.

[10] HU W, OBERG J, IRTURK A, et al. Theoretical fundamentals of gate level information flow tracking[J]. IEEE Trans on CAD, 2011, 30(8): 1128-1140.

[11] HU W, OBERG J, IRTURK A, et al. On the complexity of generating gate level information flow tracking logic[J]. IEEE Trans on IFS, 2012, 7(3): 1067-1080.

[12] OBERG J, HU W, IRTURK A, et al. Information flow isolation in i2c and usb[C]//The 48th ACM/EDAC/IEEE Design Automation Conference (DAC). San Diego, CA, USA: IEEE, 2011: 254-259.

[13] OBERG J, MEIKLEJOHN S, SHERWOOD T, et al. A practical testing framework for isolating hardware timing channels[C]//Design Automation and Test in Europe (DATE). San Jose, CA, USA: ACM, 2013: 1281-1284.

[14] OBERG J, SHERWOOD T, KASTNER R. Eliminating timing information flows in a mix-trusted system-on-chip[J]. IEEE Design and Test of Computers, 2013, 30(2): 55-62.

[15] KOCHER P C. Timing attacks on implementations of Diffie-Hellman, RSSA, DSS, and other systems[C]// Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO'96). Santa Barbara, CA, USA: Springer-Verlag, 1996: 104-113.

编 辑 蒋 晓