基于输入点复杂度的交互固有安全性度量

汤永新，余达太

(北京科技大学自动化学院 北京 海淀区 100083)

【摘要】为了弥补交互安全性度量的传统方法不反映交互基本属性所固有的安全性等不足，提出了一种交互固有安全性度量方法。通过分析软件交互基本属性中存在的固有安全性问题，提出了输入点复杂度、输入点集中度等概念，给出了输入点复杂度分级的依据和方法，以及基于此的人机交互固有安全性度量模型和步骤，并结合实践对3个软件样本进行了人机交互固有安全性度量。结果表明，该方法降低了测试技术、人员、时间等外在因素变化对交互安全性度量结果的干扰，降低了度量的技术难度，提高了度量的完整性、通用性和可信性。

关 键 词 固有安全性； 输入点复杂度； 输入点集中度； 交互安全性

中图分类号 TP3 **文献标识码** A doi:10.3969/j.issn.1001-0548.2012.05.027

Metrics of Interaction Inherent Security Based on Input Point Complexity

TANG Yong-xin and YU Da-tai

(School of Automation and Electrical Engineering, University of Science and Technology Beijing Haidian Beijing 100083)

Abstract Traditional interaction security measurement method can not reflect the inherent security of software interaction basic attributes. In order to remedy this deficiency, this paper presents a method for the metrics of interaction inherent security. Through analyzing intrinsic security problems existed in the basic attributes of interaction, we puts forward the concepts of input-point complexity and input-point concentration ratio etc., proposes a metric model of the inherent security of interaction of software based on input-point concentration ratio, and measures the inherent security of interaction of software through three examples. The measurement results show that it is effective by using input-point concentration ratio to measure inherent security of interaction. This method greatly reduces the disturbance of the changing of external factors on the metric results and, therefore, improves the credibility, integrity, and generality of the results.

Key words inherent security; input-point complexity; input-point concentration ratio; interaction security

在开放互联时代，缘于对软件人性化和网络化的追求，开发者通过在软件中提供更加强大和灵活的交互，赋予了软件使用者对软件愈加丰富的操控能力，这种人性化、网络化的交互在提高软件易用性带给人们更大方便的同时，也给软件本身引入了更多的安全性隐患，乃至于当今，由软件安全问题引发的软件安全性事件，已经成为软件使用者和开发者在整个软件生命周期都无法摆脱的困扰。为了提高软件的安全性，人们进行了各种技术和理论上的努力。在历来的漏洞利用攻击事件中，攻击者不通过与被攻击软件之间的交互攻击将无法完成，交互不仅起到了媒介的作用，还是攻击的切入点，这使得交互成为软件安全链条中至关重要的一环。如何认识交互安全性的规律已经成为软件安全研究领域一项重要内容。然而，从文献研究来看，交互安全性的度量仍然是该类研究的一个薄弱部分，本文将交互固有安全性度量方法作为研究内容，希望对此能够形成一个有益的补充。

1 相关研究及问题

对传统的软件交互安全性度量研究，主要有3种途径。

1) 认为软件的安全性问题来源于恶意交互，通过研究恶意交互者发动攻击的时间随机性变化规律来判定与交互相关的安全性。如文献[1]中将软件攻击者发动攻击所需要的一定时间看作随机变量，指

收稿日期：2011－09－06； 修回日期：2012－03－10
基金项目：国家自然科学基金重大研究计划重点项目(90818025)；国家自然科学基金面上项目(60773127)
作者简介：汤永新(1968－)，男，博士生，主要从事可信软件、人工智能方面的研究.

出这个时间变量是一个指数分布函数，并据此提出了平均安全故障时间的概念和入侵容忍系统模型以及该模型中系统状态的半马尔科夫链分析法和基于该模型的平均安全故障时间分析法。

2) 研究由交互引发的软件脆弱性的变化规律。文献[2]在研究自动定位输入相关的安全故障时，提出了软件输入覆盖(input coverage)的概念。文献[3]将输入引起的软件故障进一步细分，提出了软件输入漏洞的概念，并采用与传统软件可靠性模型同样的假设，通过对软件故障的NHPP模型[4]中平均值函数的重新构造，给出了一个软件输入漏洞发现的周期性概率模型。类似方法提出的软件输入脆弱性模型还有AT模型[5]、AML模型[6-7]、RL模型和RE模型[8]以及LP模型[9]，文献[10]随后的研究也发现，软件输入漏洞密度通常具有一定的稳定性，并随着软件安全性和质量的改进或维护有降低趋势，软件输入漏洞的衰减遵循指数分布或weibull分布。

3) 研究恶意交互对软件正常运行行为模式的影响及软件运行的安全行为模式。在对常见恶意交互方式及防御方法总结的基础上，人们提出了软件运行的安全行为模式，这类模式包括：patterns for authentication and authorization[12]、patterns for Web applications[13]、patterns for mobile Java code[14]、patterns for cryptographic software[15]、patterns for agent systems[16]以及secure GET parameters[17]等。恶意交互的重要特征是使软件运行的行为模式偏离预期，通过软件运行的实际行为模式和安全行为模式对比的差异情况便可判定软件交互的安全性。

途径1)、2)提出的模型实际上是对软件可靠性模型的沿用和改进，因此，都具有软件可靠性模型同样的局限性。文献[11]指出，AML模型适用于WINDOWS95系统，RL模型适用于WINDOWSXP系统，从另一个侧面说明了这些模型的局限性。途径3)的研究方法实质上是认为采用正确的防御方法可以减少偏离预期的运行行为，增强安全性。然而，历来的软件攻击事件表明，攻击不但可以发生在正常的运行行为下，而且没有攻不破的防御方法。因此，以交互中采取防御方法的防御能力或运行行为符合预期的程度来评价交互的安全性，受时间、攻击者能力、技术普及速度等因素差异的影响较大。

人机交互的安全性除了来自于软件在交互中所采取的技术防范措施，还来自于交互基本属性所固有的抵抗攻击的能力，以上3种方法皆不能反映软件交互本身所固有的抵抗攻击的能力(安全性)。

2 人机交互的固有安全性分析

交互的固有安全性是指软件处于运行态时，交互属性本身固有的防范恶意交互的基本能力。对交互固有安全性产生影响的交互属性包括交互的数量、交互的方式和交互的信息类型，如图1所示。交互的固有安全性反映了交互内在的和基础的安全性，是软件安全性的关键要素之一。

每一个交互对应一个或多个输入点，交互的每一个输入点都是软件向外界开放的一个窗口，每一个输入点都有可能成为被攻击者利用的攻击入口，随着软件输入点数量的增加，必然会为攻击者提供更多的机会，因而降低交互的固有安全性。

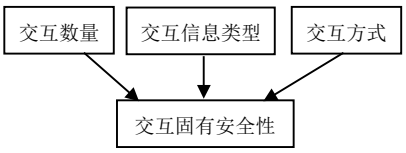

图1 交互固有安全性影响因素

交互的信息类型包括数值型数据、文本型数据、图形数据以及音视频数据。常见的溢出攻击正是攻击者利用交互信息的超预期性对软件发起攻击。交互信息的数据越复杂，软件就越难以控制交互的可预期性，输入点也就越容易被攻击者利用，从而降低交互的固有安全性。

交互方式包括主动输入式交互和被动选择式交互。在主动输入式交互中，交互信息由交互者输入，具有一定的不可预知性(从软件的角度看)。虽然，增加对输入信息的边界限制，可以提高交互的预知性，但是，无法产生根本上的改观，而对于图片、图像等类型数据的边界限制常常无法实现。这种不可预知性直接降低了软件对交互的可控性，为恶意交互提供了便利条件，降低了交互固有的安全性。在被动选择式交互中，交互输入的信息是由开发者预设，交互者只有选择权，没有干预权，交互是可预知的、可控的，这就尽可能地避免了恶意交互的发生，提高了交互固有的安全性。

3 人机交互固有安全性度量

为了建立人机交互固有安全性与交互方式、交互信息类型及交互数量之间的量化关系，需要借助以下概念。

3.1 相关概念

定义 1 输入点复杂度：交互输入点表现形式的复杂程度，包括输入点交互的信息类型和交互方式两个要素。

设U为输入点交互信息类型的论域，U={v,I,t,n}，A为输入点交互方式的论域，A={a,p}，其中，v为音视频或动态数据；i为静态图形类数据；t为文字类数据；n为数字数据(整型、浮点型)；a为主动式交互；p为被动式交互。

规则 1　$\forall u_1,u_2$，且$u_1\in U$，$u_2\in U$，若u_1能够表达u_2所表达的内容，u_2不能够表达u_1所表达的内容，称$u_1>u_2$；反之称$u_1<u_2$；若u_1与u_2信息类型相同，称$u_1=u_2$。

规则 2　$\forall x_1,x_2$，且$x_1\in A$，$x_2\in A$，若x_1方式的输入预知程度大于x_2方式的输入预知程度，称$x_1>x_2$；反之称$x_1<x_2$；若x_1与x_2方式相同，称$x_1=x_2$。

设y为输入点信息类型的复杂度，z为输入点交互方式的复杂度，则y和z分别可以看成输入点信息类型的函数和交互方式的函数，即$y=f(u)$，$u\in U$，$z=g(x)$，$x\in A$。基于以上认识，函数y和函数z分别具有以下性质：

性质 1　$\forall u_1,u_2$，且$u_1\in U$，$u_2\in U$，若$u_1>u_2$，则$f(u_1)>f(u_2)$；反之，则$f(u_1)<f(u_2)$；若$u_1=u_2$，则$f(u_1)=f(u_2)$。

性质 2　$\forall x_1,x_2$，且$x_1\in A$，$x_2\in A$，若$x_1>x_2$，则$g(x_1)>g(x_2)$；反之，则$g(x_1)<g(x_2)$；若$x_1=x_2$，则$g(x_1)=g(x_2)$。

根据规则1、2及性质1、2可以得出以下结论：

1) $f(v)>f(i)>f(t)>f(n)$；2) $g(p)>g(a)$。

规则 3　输入方式是输入点的起始激发因素，在判定输入点复杂度时，z的优先级高于y的优先级。

根据以上规则和性质，交互输入点复杂度的判定可以分为5个等级，如表1所示。

表1　交互输入点复杂度分级表

输入点复杂度等级	交互方式	交互的信息类型	特点
5	主动输入	各类音视频数据(AVI、RM、SWF、MP3等)或多媒体数据	可控性差、复杂性最强
4	主动输入	各类图形数据(JPG、GIF、BMP等)	可控性差、复杂性强
3	主动输入	文本类数据	可控性差、复杂性一般
2	主动输入	数字型数据(整型、浮点型)	可控性差、复杂性弱
1	被动选择	预设内容的各类型数据	可控性强

定义 2　输入点集中度：指软件输入点复杂度为某级以上(含某级)的输入点个数占软件输入点总数的百分比，以CR_n表示，其中，n为输入点复杂度等级，称CR_n为输入点的n级集中度。

由于交互和输入点之间存在的隶属关系，使得输入点集中度(CR_n)成为交互固有安全性影响因素(交互数量、交互方式、交互信息类型)的集中体现。

从历来的软件安全性事件来看，1、2级复杂度的输入点很少能够成为攻击事件的激发点，软件攻击多发生在3级以上复杂度的交互中，所以，CR_3对交互的安全性具有较好的指示作用，适宜用作交互安全性的指向性指标。

3.2　交互固有安全性度量方法

1) 依据表1采用黑盒方式测试统计各级复杂度的输入点数量，并计算软件输入点的总数IPS。

设某软件中的交互总数为n，其中第i个交互中含IN_i个输入点，则：

$$IPS=\sum_{i=1}^{n}IN_i \tag{1}$$

2) 计算各级复杂度的输入点总数占软件输入点总数的百分比PIP_k，k为输入点复杂度等级。

设软件中的交互总数为n，第i个交互中的输入点总数为IN_i，含有k级输入点复杂度的交互总数为m_k，其第j个交互中k级输入点总数为IN_{jk}，则：

$$PIP_K=(\sum_{j=1}^{mk}IN_{jk}/\sum_{i=1}^{n}IN_i)\times 100\%\quad 1\leqslant k\leqslant 5 \tag{2}$$

3) 计算CR_3，有：

$$CR_3=\sum_{k=3}^{5}PIP_k \tag{3}$$

4) 通过CR_3指标判断软件交互的固有安全性。

基于输入点复杂度越高，交互的固有安全性越低的认识，将人机交互的固有安全性进行分级，如表2所示。

表2　交互固有安全性分级表

CR_3		语义安全级
$CR_3\leqslant 0\%$	$PIP_2\leqslant PIP_1$	高$^{++}$
	$PIP_2>PIP_1$	高$^{+}$
$10\%<CR_3<20\%$	$PIP_2\leqslant PIP_1$	高
	$PIP_2>PIP_1$	高$^{-}$
$20\%\leqslant CR_3<25\%$		一般
$25\%\leqslant CR_3<30\%$		一般$^{-}$
$30\%\leqslant CR_3<40\%$		低
$40\%\leqslant CR_3<50\%$		低$^{-}$
$50\%\leqslant CR_3<60\%$		较低
$60\%\leqslant CR_3<70\%$		较低$^{-}$
$70\%\leqslant CR_3<80\%$		最低$^{+}$
$80\%\leqslant CR_3<90\%$		最低
$CR_3\geqslant 90\%$		极低

在实际中，由交互引发的安全隐患绝大多数主

要来自于3级复杂度以上的输入点，当CR_3小于20%时，其对交互的固有安全性的区分作用大幅减弱，改由PIP_2和PIP_1直接比较。

3.3 实 例

运用上节提出的方法，对某3个软件进行了实测统计，数据如表3～表5所示。

表3 软件1输入点统计

输入点复杂度等级	交互方式	交互信息类型	输入点数量
5	主动输入	音视频数据	40
4	主动输入	图形数据	1
3	主动输入	文字数据	297
2	主动输入	数字数据	143
1	被动选择	预设信息	433
总计		914	

表4 软件2输入点统计

输入点复杂度等级	交互方式	交互信息类型	输入点数量
5	主动输入	音视频数据	2
4	主动输入	图形数据	1
3	主动输入	文字数据	63
2	主动输入	数字数据	148
1	被动选择	预设信息内容	135
总计		349	

表5 软件3输入点统计

输入点复杂度等级	交互方式	交互信息类型	输入点数量
5	主动输入	音视频数据	0
4	主动输入	图形数据	0
3	主动输入	文字数据	7
2	主动输入	数字数据	1 152
1	被动选择	预设信息内容	423
总计		1 582	

软件交互固有安全性判别结果如表6所示。从表6来看，软件1的PIP_1尽管达到了47%以上较高的水平，但是CR_3在30%~40%之间，说明软件1中仍有较多的3级以上的输入点对软件交互的固有安全性起着主导作用，也可以说软件1的交互设计给恶意交互提供了更多的攻击点，使软件1的交互固有安全性大大降低；软件2的CR_3小于20%，说明软件2中2级以下的输入点占了绝大多数，3级以上输入点对软件交互的固有安全性影响大量减少，相应地使得软件2的交互固有安全性大为提高；软件3的CR_3不足0.5%，表明软件3的交互设计给攻击者留下的攻击点非常少，因此，软件3的交互固有安全性非常高。

表6 软件交互固有安全性判别结果

软件名称	CR_3/%	PIP_2/%	PIP_1/%	交互安全等级
软件1	36.98	15.65	47.37	低
软件2	18.91	42.41	38.68	高$^-$
软件3	0.44	72.82	26.74	高$^+$

3.4 比较与讨论

表7 交互安全性模型比较

比较项	方法		
	输入漏洞发现周期概率模型	运行行为模式安全性判别法	交互固有安全性度量法
判断准则	发现输入漏洞的周期越长，越安全	运行行为模式越符合预期，越安全	输入复杂度越低，越安全
适用性	针对性强，适用范围较窄	通用性较强，适用范围较广	通用性强，适用范围广
运用难度	需要大量历史试验数据和针对输入漏洞的挖掘技术，技术实现难度较大，对测试人员技术素养要求高。	需要专家经验和针对性的实时监测技术，技术实现难度较大，对测试人员技术要求较高。	以专家经验为基础，测试技术简单，对测试人员技术要求较低。
特点	1) 无法反映交互安全问题的来源范围。2) 结果准确性受样本多少、漏洞发掘工具的性能以及测试人员、测试方法等影响大。3) 度量所需时间较长。	1) 不能反映交互安全性问题的来源范围，无法反映利用正常交互行为进行攻击时软件面临的风险。2) 度量结果受监测技术影响大。3) 度量所需时间较短。	1) 可以在一定程度上反映交互安全性问题的存在范围。2) 度量结果不受测试工具、测试方法的影响。3) 度量所需时间短。

在进行交互安全性度量时，软件输入漏洞发现周期性概率类模型建立在软件的修正完善不引入新漏洞的前提条件下，而该条件受诸多主客观等因素的影响，显然难以被满足，使得该类模型针对性都较强，适用范围具有明显的局限性；运行行为模式安全性判别类方法，是基于安全行为模式的总结和预期，忽略了攻击者主观策略对软件安全性的作用，符合预期的行为模式同样可以被用来进行攻击。在安全观上以行为符合预期等于安全为准侧，显然具有明显的疏漏；交互固有安全性度量法反映了交互自身固有的抵御攻击的能力，其从交互内部属性的角度出发，减少了外在因素对度量结果的不利影响，具有更强的适用性，是对前两种方法的有益补充。

交互安全性模型比较如表7所示。通过表7的比较可以发现，输入漏洞发现周期概率模型在实际度量中受到的影响因素最多、最难以把握，行为模式安全性判别法则适中，交互固有安全性度量法最简单易行、高效。3类方法分别从不同的角度反映了交互的安全性，输入漏洞发现周期概率模型反映了交互安全性在时间上的特征，行为模式安全性判别反映了交互在防御方法和运行行为上的安全性，交互固有安全性度量反映了交互自身固有的安全性，将三者结合运用可以更加完整、科学地认识软件交互的安全性。

4 结 论

交互是攻击软件的必经之路，交互的安全性是软件安全性的重要组成部分。交互的安全性度量是认识交互安全性规律的关键和基础，作为对软件交互安全性度量传统模型和方法不足的有益补充。本文提出了一种交互固有安全性度量方法，并结合实例进行了实际的度量。总结以上研究，可以得到结论：1) 软件交互基本属性固有的安全性是认识交互安全性的一种重要方式，在不改变软件交互设计的情况下，交互固有安全性度量法所得结果更具稳定性，且该方法兼具简单易行、干扰因素少、通用性强等特点。2) 交互的输入点是恶意交互的激发点，输入点的复杂度越高，交互的固有安全性越低。为了提高软件交互的安全性，在不影响软件使用效率和功能的情况下，应该尽量减少软件的交互数量，或尽量采用被动选择式交互方式，尤其要控制不安全交互方式的使用。

输入点的CR_3指标对交互的固有安全性具有较好的指示作用，当$CR_3 \geqslant 20\%$时，CR_3对交互固有安全性的区分起主导作用，当$CR_3 < 20\%$时，区分度减弱，可改由PIP_2和PIP_1进行区分。

参 考 文 献

[1] MADAN B, GOSŠEVA-POPSTOJANOVA K, VAIDYANATHAN K, et al. Modeling and quantification of security attributes of software systems[C]//International Conference on Dependable Systems and Networks. Washington, D C, USA: IEEE Press, 2002: 505-514.

[2] NAGY C, MANCORIDIS S. Static security analysis based on input-related software faults[C]//Proc 13th European Conference on Software Maintenance and Reengineering, CSMR 2009. Kaiserslautern, Germany: IEEE Press, 2009: 37-46.

[3] KIMURA M. Software vulnerability: Definition, modelling, and practical evaluation, fore-mailtransfer software[J]. International Journal of Pressure Vessels and Piping, 2006, 83: 256-261.

[4] HUANG Chin-yu, LYU M R, KUO Sy-yen. A unified scheme of some nonhomogeneous Poisson process models for software reliability estimation[J]. IEEE Trans on Software Eng, 2003, 29(3): 261-269.

[5] ANDERSON R. Security in opens versus closed systems-the dance of boltzmann, coase and moore[EB/OL]. (2006-07-25)[2010-10-11]. http://www.cl.cam.ac.uk/~ rja14/Papers/toulouse.pdf.

[6] ALHAZMI O H, MALAIYA Y K. Quantitative vulnerability assessment of systems software[C]//Proc Annual Reliability and Maintainability Symposium. Alexandria, Virginia, USA: IEEE Press, 2005: 615-620.

[7] ALHAZMI O, MALAIYA Y, RAY I. Security vulnerabilities in software systems: a quantitative perspective[C]//Proc 19th Annual IFIP WG 11.3 Working Conference on Data and Applications Security. Storrs, CT, USA: Springer Berlin / Heidelberg Press, 2005: 281-294.

[8] RESCOLA E. Is finding security holes a good idea?[J]. Security and Privacy, 2005, 3(1): 14-19.

[9] MUSA J D, OKUMOTO K. A logarithmic Poisson execution time model for software reliability measurement[C]//Proc 7th International Conference on Software Engineering. Orlando, FL, USA: IEEE Press, 1984: 230-238.

[10] PENTA M D, CERULO L, AVERSANO L. The evolution and decay of statically detected source code vulnerabilities [C]//2008 Eighth IEEE International Working Conference on Source Code Analysis and Manipulation. Beijng, China: IEEE Press, 2008: 101-110.

[11] ALHAZMI O H, MALAIYA Y K, RAY I. Measuring, analyzing and predicting security vulnerabilities in software systems[J]. Computers&Security, 2007, 26: 219-228.

[12] BROWN F L, DIVIETRI J, VILLEGAS G D de, et al. The authenticator pattern[C]//Proc Sixth Conf Pattern Languages of Programming. Monticello, Illinois, USA: Hillside Group, Inc, 1999: 1-8.

[13] BONURA D, CULMONE R, MERELLI E. Patterns for Web applications[C]//Proc of the 14th International Conf on Software Engineering and Kowledge Engineering,SEKE'02. Ischia, Italy: ACM New York, NY, USA, 2002: 739-746.

[14] MAHMOUD Q H. Security policy: a design pattern for mobile Java code[C]//Proc Seventh Conf Pattern Languagesof Programming. Monticello, Illinois, USA: Hillside Group, Inc, 2000: 1-8.

[15] BRAGA A M, RUBIRA C M F, DAHAB R. Tropyc: a pattern language for cryptographic software[C]//Proc Fifth Conf Pattern Languages of Programming. Monticello, Illinois, USA: Hillside Group, Inc, 1998: 2-27.

[16] MOURATIDIS H, GIORGINI P, SCHUMACHER M. Security patterns for agent systems[C]//Eighth European Conference on Pattern Languages of Programming. Irsee, Germany: Hillside Group, Inc, 2003: C4,1-16.

[17] HALKIDIS S T, TSANTALIS N, CHATZIGEORGIOU A, et al. Architectural risk analysis of software systems based on security patterns[J]. IEEE Transactions on Dependable and Secure Computing, 2008, 5(3): 129-142.

编 辑 漆 蓉