基于时态逻辑的可信平台信任链建模*)

李 莉 曾国荪 陈 波
(同济大学计算机科学与技术系 上海 201804)
(国家高性能计算机工程技术中心同济分中心 上海 201804)

摘 要 针对可信计算中信任链理论缺乏深入分析验证的现状,分析了信任链理论中,可信与信任的内涵及其隐含的动态特性,借助时态逻辑的时间推理能力,建立了可信平台中的信任链的形式化模型,依据该模型,观察信任传递的过程,从而从理论上验证了信任在信任链上的可传递性,并得出信任在信任链上传递的充分条件。

关键词 可信,信任,信任链,模态逻辑

Temporal-logic-based Model for Chain of Trust of Trusted Platform

LI Li ZENG Guo-Sun CHEN Bo
(Department of Computer Science and Technology, Tongji University, Shanghai 201804)
(Tongji Branch, National Engineering and Technology Center of High Performance Computer, Shanghai 201804)

Abstract According to the current situation of the absence of the formally study to the chain of trust in trusted computing, the notions and the dynamic properties of trusted and trust are analyzed. By virtue of the temporal reasoning abilities of the temporal logic, a formal model of the chain of trust is built. Based on this model, the transfer of the trust on the chain of trust can be inspected, thus the transferability of the trust on the chain of trust is proved theoretically, finally the conditions for the transferability of trust is educed.

Keywords Trusted, Trust, Chain of trust, Temporal logic

1 引言

信息时代,信息成为一种至关重要的资产。而这些资产暴露在越来越多的威胁中,所以保护信息资产安全,建立一个可信任的计算环境已经成为信息产业发展的优先需求之一。当前网络安全措施不适于信息安全威胁主要源自内部的情况。要解决内部的安全威胁,就必须以终端可信任为基础,才能从源头上解决人与程序、人与机器间的信任关系。这样,一种基于信任链的可信终端技术被提出,用以解决用户与计算平台间信任建立问题。

早期研究者将安全操作系统作为可信的基础,这要求操作系统的支撑软、硬件必须可信[1]。这些研究者通常假设可信操作系统的支撑软、硬组件包含在可信计算机内[2],从而回避了支撑组件是否可信这一问题。1997 年 Arbaugh, W. A 提出一种安全启动过程:AEGIS[3],它通过构建一条完整性检查链,验证启动过程中涉及代码模块的完整性来保证上述支撑软、硬组件的可信性。此后 Arbaugh, W. A 和一些研究者对 AEGIS 进行了改进,增强其柔韧性[4,5]。1999 年,TCPA 成立对可信计算技术的研究起到了极大推动作用。该组织后更名为 TCG,在 TCG 制定的多项规范中[6,7]涉及信任链的实现等内容,并将信任链拓展到了网络中。

上述文献都从技术实现角度对信任链进行了研究,但缺乏对信任链的深入分析,没有从可信与信任概念的角度分析信任链上节点、研究信任传递的条件,以及考虑信任链上信任的动态性。本文在分析可信及信任内涵基础上,分析信任链上层次节点的特性,并考虑到信任的动态特性,基于时态逻辑对可信平台中的信任链形式化建模,从而得出信任传递的条件,并验证信任链上信任传递的可达性。

2 基本概念

2.1 信任链

信任链理论[6]的基本思想为:在可信计算体系中建立可信,需先拥有可信根(Root of Trust, RT),然后建立一条信任链(Chain of Trust),再将信任传递到系统的各个组件,之后就能建立整个系统的可信。由此,任何硬件和软件模块运行之前,需先建立对这些模块代码的信任。这种信任是通过在执行权转移之前对代码进行度量来确认的。可信 PC 系统中,信任链上各层次节点可包括:

RT→OSLoader→OSCode→Applation code。

2.2 时态逻辑

时态逻辑是模态逻辑的一种,适于推理随时间而更改的特性,定义如下:

定义 1 时态逻辑的语义模型是 Kripke 三元组(S ,δ, I) ,其中:

$S=\{S_0, S_1, \cdots\}$ 是非空状态集;$\delta=\{\delta_0,\delta_1,\cdots\}$ 是非空路径(状态序列) 集,$\delta_i=S_iS_{i+1}\cdots$称为一条路径;$I: F*S\rightarrow\{\text{True, False}\}$是谓词集合与 S 的笛卡儿集到{True, False}的映射。即:$\forall P\in F, S, \in S, I(p, S_i)=\text{True}\vee\text{False}$,且仅具其一。若 $I(p, S_i)=\text{True}$,记为$(I, S_i)\dashv p$。

定义 2 设 p 是一个时态逻辑公式,则 p 在状态序列 S_i

*)本课题得到国家自然科学基金(60673157)、教育部科研重点项目(105071)资助。李 莉 博士生,主要研究领域为:信息安全、可信计算;曾国荪 博士,教授,主要研究领域为异构计算、并发理论;陈 波 博士生,副教授,主要研究领域:可信计算、信任管理。

上的语义解释为：

$(I,S_i)\vdash puq \Leftrightarrow \exists k \geqslant i$，$(I,S_k)\vdash q$ 并且 $\forall j, i \leqslant j < k$，$(I,S_j)\vdash p$；

$(I, S_i)\vdash \Box p \Leftrightarrow \forall k \geqslant i$，$(I,S_k)\vdash p$；

2.3 密码学

非对称加密机制中，实体 X 所拥有的公私钥对为二元组 (PKx, SKx)。可定义数字签名机制为一个三元组 $(GenKey(), Sign(), Verify())$，它们依次代表了密钥生成、签名和验证。$\sigma \leftarrow Sign(SKx, m)$ 表示使用私钥 SKx 对信息 m 签名后生成 σ。公式 $ind \leftarrow Verify(PKx; m, \sigma)$ 表示使用公钥 PKx 对签名进行验证，函数的返回值 $ind \in$ (True, false)，指验证的结果为真或假。可信平台中的组件拥有确认(Validation)证书[6]，该证书由可信第三方(The Third Party, TTP)发给组件厂商，该证书包括以下内容：组件 Id、组件特征度量值的散列值(state)等。若 TTP 的公私钥对表示为 $(PKttp, SKttp)$，那么 TTP 对组件信息进行签名，$CINFO \leftarrow sign(SKttp; Id + state)$，则组件确认证书可描述为 $cert(SKttp, CINFO)$。

3 信任链节点可信与对节点的信任

对可信内涵还存在不同的认识，TCG 强调安全性，认为一个实体可信，如果它的行为符合预期。容错流派强调可靠性。IEEE 汇刊则认为可信包含可靠和安全。在本文的讨论中，可信内涵分析是针对信任链上的组件节点而言，其内涵实质包含上述两个方面。

3.1 可信内涵与可信组件定义

可信反映的是实体的状态特性，可信的实体必须具有以下特性[7]：第一、实体身份明确；第二、实体行为明确。当实体性质未被改变时，可以确定实体的行为方式是符合设定目的的；第三、任意时刻，实体的状态能够被判断和真实地反映。

组件指平台中能被唯一标识的、功能上相对独立、可替换的构成成分。组件可被唯一标识。组件的名称、模块号、版本等信息可被用来表征组件的唯一身份。组件的完整性(Integrity)可表示组件有无被非法篡改。组件出厂状态是完整的，如果经过升级或者加补丁的方式被厂商更改，组件的完整性状态仍然为真。组件的功能明确且相对独立，其功能是由设计组件的目的决定。

根据“可信”内涵分析与组件描述，可给出一个组件可信的条件及定义可信谓词。

条件 I：组件身份明确：组件的身份可验证。

条件 E：组件的行为可预期：一个组件总是以厂商设计的行为方式执行，除非该组件被非法篡改。即组件的完整性状态为真时，其确定的行为可以期望。

条件 M：组件完整性状态可被真实度量：组件的完整性状态，在任何运行时刻都能被度量和真实反映。

定义 3(可信谓词) $Trusted(C)$，表示一个组件 C 是可信的。一个组件的“可信性”是动态变化的，一个被判定可信的组件在运行过程中如果被篡改，它将不满足可信条件 E。

3.2 信任

信任与可信相关却不同。信任是实体间的二元关系，是一个主体对另一个客体所具备能力的信心以及对其良好行为的预期[8]。许多的文献[8~9]都对信任进行了定义、描述。本文根据信任链中信任关系的特性，将信任的属性纳入信任的定义中，认为信任是：依据某种准则，在客体 B 满足主体 A 定义的条件下，主体 A 对客体 B 达到它所期望的某个目的的肯定。这种定义强调信任建立的条件，一旦信任的条件不满足，之前建立的信任关系将失效。也可以说信任关系是动态变化的，信任关系建立后并非一成不变。

定义 4(信任谓词) $Trust(A,B)$，表示主体 A 信任客体 B

条件满足谓词：$Meet(B, conditions)$ 表示客体 B 满足条件 $conditions$。

在可信平台内，满足条件 I、E、M 的组件为可信组件，即公式(1)。用户对组件的信任以是否满足可信的条件为判定依据，一个可信的组件能被终端用户信任。这种信任关系将一直维持除非可信组件的完整性别破坏，即不满足条件 E。由时态逻辑中定义 2 有公式(2)：

$$Meet(C, \mathrm{I}) \wedge Meet(C,\mathrm{E}) \wedge Meet(C,\mathrm{M}) \rightarrow Trusted(C) \quad (1)$$

$$Trusted(C) \rightarrow Trust(user, C)\ u \rightarrow Meet(C,\mathrm{E}) \quad (2)$$

3.3 信任链节点的可信验证

验证信任链上的一个组件是否可信，由该组件在信任链上的前驱节点来执行。验证包括对组件进行身份认证，计算被验证组件特征值的摘要，通过与期望值比较，获得该组件的完整性状态。假设组件 $C1$ 满足：$Trusted(C1)$，则 $C1$ 验证 $C2$ 的过程可描述如下：

Procedure Measure($C1$, $C2$)

I. 获取组件 $C2$ 的身份 Id 和当前特征值 $METRIC$

II. 通过组件 $C2$ 的确认证书 $cert(SKttp, CINFO)$ 来验证

i. $True? = Verify(PKttp; Id, CINFO)$

ii. $True? = Verify(PKttp; hash(METRIC), CINFO)$

如果 i 式为真，则组件 $C2$ 身份可确认；如果 ii 式结果为真，组件 $C2$ 的完整性状态为真。则 $C2$ 满足条件 I、E，又根据假设 $C1$ 可信，则其验证结果的真实性可保证。由此判定 $C2$ 是可信的。根据该验证过程和表达式(1)(2)，有以下公式成立：

$$Meet(C2,\mathrm{I}) \wedge Meet(C2, \mathrm{E}) \wedge Trusted(C1) \rightarrow Trusted(C2) \quad (3)$$

$$Trusted(C2) \rightarrow Trusted(C2)u \rightarrow Meet(C2, \mathrm{E}) \quad (4)$$

4 可信平台的信任链建模

基于前文对信任、信任链节点可信的分析，本节考虑信任的动态特性，采用时态逻辑对 PC 平台中的信任链模型，分析基于信任链的信任传递过程和条件。

4.1 谓词集合

谓词 $Trust()$；谓词 $Meet()$；谓词 $Trusted()$。

4.2 状态集 S 与路径集 δ

定义状态集：$S = \{S_0, S_1, S_2\}$，其中 S_0 表示可信根 RT 对 OSLoader 验证结束，控制权传递的时间点，S_1 表示 OSLoader 对 OSCode 验证结束，控制权传递的时间点，S_2 表示 OSCode 对 AppCode 验证结束的时间点。

$\delta = \{\delta_0, \delta_1, \delta_2\} = \{S_0\ S_1 S_2, S_1 S_2 S_0, S_2\}$ 是路径集。

4.3 信任链动态模型及分析

基于时态逻辑的时间推理能力，分析信任在信任链节点间的动态传递过程，可获得信任传递的条件并验证可达性。为描述清晰，我们假设：假设 1：每个被验证的组件，其身份和完整性状态都通过认证，即满足可信的条件 I，E。假设 2：在 OSCode 对 AppCode 验证时，不失一般性，可假设验证了一个

应用软件组件。

可信平台启动过程是信任链的动态建立过程，信任将沿信任链传递。平台启动时第一个运行的组件是信任传递的原点，即可信根(RT)。一般认为可信根是可信、功能正确而且不需要外界保护的，它可以由专家来评估和确定是否符合可信的标准。则 RT 满足可信组件的三个条件 I、E、M。即：

$$Meet(RT, I) \wedge Meet(RT, E) \wedge Meet(RT, M) \rightarrow Trusted(RT) \quad (5)$$

$$(I, S_0) \vdash \Box Trusted(RT) \quad (6)$$

S_0 状态：平台启动后信任从可信根开始，沿信任链传递。RT 验证 OSLoade 结束

由式(6)得：$(I, S_0) \vdash Trusted(RT)$ (7)

由假设 1 得：

$$(I, S_0) \vdash Meet(OSLoader, I) \wedge Meet(OSLoader, E) \quad (8)$$

由式(3)(4)(7)(8)得：

$$(I, S_0) \vdash Trusted(OSLoader)u \rightarrow Meet(OSLoader, E) \quad (9)$$

由式(2)(9)得：

$$(I, S_0) \vdash Trust(user, OSLoader)u \rightarrow Meet(OSLoader, E) \quad (10)$$

(10)式表示在 S_0 状态点，用户信任成功传递到 OSLoader

S_1 状态：OSLoader 验证 OSCode 后

由假设 1 得：

$$(I, S_1) \vdash Meet(OSCode, I) \wedge Meet(OSCode, E) \quad (11)$$

在状态 S_0 到 S_1 之间，即 OSLoader 被验证可信到它运行过程中，设 OSLoader 没有被篡改，即：

$$(I, S_1) \vdash Meet(OSLoader, E) \quad (12)$$

由公式(9)(12)得：

$$(I, S_1) \vdash Trusted(OSLoader) \quad (13)$$

由公式(3)(4)(11)(13)得：

$$(I, \delta_1) \vdash Trusted(OSCode)u \rightarrow Meet(OSCode, E) \quad (14)$$

由式(2)(14)得：

$$(I, S_1) \vdash Trust(user, OSCode)u \rightarrow Meet(OSCode, E) \quad (15)$$

(15)式表示在 S_1 状态点，用户信任成功传递到 OSCode

S_2 状态：OSCode 验证 AppCode 结束

由假设 1 得：

$$(I, S_2) \vdash Meet(AppCode, I) \wedge Meet(AppCode, E) \quad (16)$$

在状态 S_1 到 S_2 之间，即 OSCode 被验证可信直到它运行，然后验证 AppCode 的时间段内，设 OSCode 没有被篡改，即：

$$(I, S_2) \vdash Meet(OSCode, E) \quad (17)$$

由式(14)(17)得：

$$(I, S_2) \vdash Trusted(OSCode) \quad (18)$$

由式(3)(4)(16)(18)得：

$$(I, S_2) \vdash Trusted(AppCode)u \rightarrow Meet(AppCode, E) \quad (19)$$

最后由式(2)(19)得结论：

$$(I, S_2) \vdash \rightarrow Trust(user, AppCode) \rightarrow Meet(AppCode, E) \quad (20)$$

由(20)式可知，用户的信任由可信根开始，传递到了应用组件。则有结论：在假设 1 和 2 基础上，由可信根开始，信任可以传递到平台的各个组件。但是传递的充分条件是：被验证可信的组件在运行时未被篡改。正如 S_1 状态中假设 OSLoader 没有被篡改，即式(12)：$(I, S_1) \vdash Meet(OSLoader, E)$；以及 S_2 状态中所作的假设(16)：$(I, S_2) \vdash Meet(OSCode, E)$。

总结与进一步研究 本文分析了可信和信任的内涵及信任的动态特性，并借助时态逻辑的时间推理能力，对可信平台中信任链形式化建模，分析信任链上信任传递的过程以及传递的充分条件。基于该形式化的分析过程可知，在可信根的基础上，信任沿信任链传递的充分条件是：被验证可信的组件，在运行过程中能够保持其完整性。所以，基于二进制的组件完整性的验证，并不能保证其运行时的动态完整性。尤其是一些易变软件组件。所以组件的动态完整性的度量和保障是保证信任成功传递的充分条件。今后我们需要关注软件的动态完整性度量与保护机制。

参 考 文 献

1 Schroeder M. Engineering a Security Kernel for Multics. In: 5th Symposium on Operating Systems Principles. TX USA, 1975. 125～132

2 Latham D C. Trusted computer system evaluation criteria. Dod 5200. 28-STD. Department of Defense, USA, 1985

3 Arbaugh W A, Farber D J, Smith J M. A secure and reliable bootstrap architecture. In: Proceedings of the 1997 IEEE Symposium on Security and Privacy table of contents, 1997. 65

4 Arbaugh W A. Chaining layered Integrity Checks. University of Pennsylvania; jean Gallier, 1999

5 Arbaugh I W A, Pollack S J, Reeves D. M. Personal secure booting. In: Proceedings of the 6th ACISP. Sydney, Australia, 2001. 130～144

6 TCG Specification Architecture Overview Specification Revision 1. 2. http://www. trustedcomputing. org, April 2004

7 TCG. TCG Infrastructure Working Group Architecture Part II - Integrity Management Specification Version 1. 0. Revision 1. 0, http://www. trustedcomputing. org, November 2006

8 Xiu Daoxi, Liu Zhaoyu. A Formal Definition for Trust in Distributed Systems. LNCS, 3650, ISC. Heidelberg: Springer, 2005. 482～489

9 Chopra K, Wallace W A. Trust in Electronic Environments. In: Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03), Hawaii, 2003