The CSP Protocol Model Based on Data Independence

Zheng Jun-Jie[1] *Kong Yi*[1] *Liu Zhi-Hua*[1] *Zhu Xiao-Lin*[2] *Ye Song*[1] *Ruan Kun*[1]

（1. Institute of meteorology,Liberation Army University of Science and Technology, Nanjing,211101, China；2. PLA GSH,Beijing,100000）

e-mail:oldwolf0411@sina.com

Abstract：In the field of security protocol formal analysis,model check tools such as FDR,generally been limited into a given small instance restricted by the finiteness of some resources set such as keys and nonces,so the verification result is incomplete.Based on data independence, the CSP protocol model is extended by using new process,through the translation function ,it is possible to create protocol models in which node can call upon an infinite supply of nonces、keys,etc and do not have to stop after a limit number of runs.

Key words：security protocol ；CSP；model checking；data independence

基于数据独立技术的 CSP 协议模型

郑君杰[1] 孔毅[1] 刘志华[1] 朱晓林[2] 叶松[1] 阮鲲[1]

（1.解放军理工大学气象学院，江苏，南京，211101；2.解放军总参谋部，北京，100000）

e-mail:oldwolf0411@sina.com

【摘要】在安全协议形式化分析领域，以 FDR 为代表的模型检测工具由于受到密钥、随机数等资源数量的限制，其使用通常被限定在一个给定的小系统里，验证结果缺乏完备性。基于数据独立技术，引入新的进程扩展 CSP(通信顺序进程)协议模型，通过转换函数使创建的协议模型节点可以调用无限个资源而无需停止协议的运行，确保了验证结果的完备性。

【关键词】安全协议；CSP；模型检测；数据独立

1. 引言

安全协议的安全性通常依赖协议中的密钥、随机数等数据对象的独一无二性或不可猜测性。如某些安全协议，在每次运行时其中某个主体产生一个随机数参与协议的运行，必须要确保这个随机数是没有使用过的，即它是“新鲜”（fresh）的，因为攻击者可能会使用其曾经截获过的随机数进行重放等攻击。基于模型检测技术的安全协议验证已经取得了很大的成功,但是在使用模型检测工具如 FDR[1][2] 、SMV、SPIN[3] 等进行安全协议的验证时，一些协议中的对象如随机数、密钥等的大小通常被限制在一个比较小的范围内，如仅仅被设定成单一的数字。但是在协议的实际执行中所需的数据值可能比这大许多。另外含有随机数的协议在验证时随机数保存在缓存中，每次运行时从中选用一个参与协议的验证，被使用过的随机数被剔除出去不再使用。如果缓存中没有随机数留下，模型检测工具就停止运行，以上状况导致了以下两个问题：

1.尽管模型检测工具被认为是一个极为有效的安全缺陷发现工具，但它仅仅能证明节点在执行有限活动的情况下没有发现攻击，不能给出一个完全的证明；

2.目前没有一种有效的机制来确保验证过程中数据对象的独一无二性。

基于文献[4] [5] [6]提出的数据独立技术，引入新的进程扩展 CSP 协议模型，通过转换函数使节点可以调用无限个新鲜的随机数值以保证检测实例的无限次运行，证明了模型检测工具验证安全协议结果的完备性。

2. 数据独立技术

2.1 基本概念

定义 1:如果一个进程 P 对于数据类型 T 没有任何限制，则 P 对于 T 类型是数据独立的，此时 T 可以被视为 P 的参数。

安全协议中的很多对象都可以被视为数据独立实体，如常见的 Key，nonce 等，因为它们不依赖于特定的协议参与者。

2.2 Collapsing 函数

显然一般目标的数据独立结果不适用于安全协议分析，因此需要对这些结果进行相应的推论。

对一般数据独立分析结果进行推论所基于的基本原则也是证明数据独立最重要的方法，即对进程 P 的

资助信息：解放军理工大学气象学院基础理论研究基金
解放军理工大学联合作战理论基金

978-1-935068-01-3 © 2009 SciRes.

参数类型T应用 Collapsing 函数ϕ建立映射关系，证明映射前$P(T)$的行为经过函数转换，是$P(\phi(T))$行为的子集。

如果 Collapsing 函数ϕ是单映射的(T 的所有成员被映射为不同值)，并应用于进程P的参数类型T上(P对于T数据独立)，则因为T的所有成员被映射为不同值，所以映射前的行为等价于映射后的行为：

$$traces(P(\Phi(T))) = \{\Phi(t) \mid t \in traces(P(T))\} \quad (1)$$

如果使用的 Collapsing 函数ϕ为非单映射函数，则有可能改变等价测试结果，产生不等式(2)：

$$traces(P(\Phi(T))) \subseteq \{\Phi(t) \mid t \in traces(P(T))\} \quad (2)$$

该不等式只适用于变量，对于程序中的常量，情况有所不同。

定义 2：如果进程P中的每一次等价测试失败都会引发 STOP 进程，则称进程P满足 PosConjEqT 条件。

PosConjEqT(Positive Conjunctions of Equality Tests)条件可以使变量使用非单映射函数的进程再度满足等式(1)。

对于常量虽然不能重新满足等式(1)，但可满足不等式（3）：

$$\{\Phi(t) \mid t \in traces(P(T))\} \subseteq traces(P(\Phi(T))) \quad (3)$$

协议的某些方面可能并不满足 PosConjEqT 条件，如代理进程可能需要执行拥有常量(例如名字)的不等式测试。因此，针对常量集 C 定义 P osConjEqT′c 条件如下：

定义 3：如果进程P对于常量集 C 满足下列条件，则称进程P满足 PosConjEqT′c 条件:满足 PosConjEqT 条件，但与 PosConjEqT 条件的不同之处是对于至少包含常量集 C 的一个成员的等价测试，P可能拥有 non-STOP 结果。

协议的某些方面可能并不满足 PosConjEqT 条件，如代理进程可能需要执行拥有常量(例如名字)的不等式测试。因此，针对常量集 C 定义 P osConjEqT′c 条件如下：

$$\forall x \in T, c \in C \bullet \phi(c) \Leftrightarrow x = c$$

其中T是可以应用ϕ函数的且数据独立的数据类型。

从上述定义可以确定这样的性质：如果在明确推理系统中建立攻击者，并且攻击者的初始知识集不包含对于类型T成员(除了常数 C)的非等价测试，那么该进程满足 PosConjEqT′c 条件。

可以看出系统成分是否满足上述性质对研究至关重要，只有满足这些条件才能够在协议分析的 CSP 模型中构造更为复杂的事件。

3. CSP 协议模型的扩展

Roscoe 的数据独立技术提供了这样的理论基础[6]：若系统的所有成分满足 PosConjEqT′c 条件，则大协议系统中转换前的全部行为(就系统迹而言)可以用经过映射的相应的有限系统描述。这一技术保证了可以在大协议中应用非单映射转换，将问题简化为相应的有限系统。

为了解决验证的局限性，需要系统代理能够在实际类型保持有限的情况下，调用无限的不同的新鲜值。沿用 Roscoe 的方法，引入 Manager 进程扩展 CSP 协议模型。以 Yahalom 协议和 NS 公钥认证协议为例，扩展后的模型分别如图 1 和 2 所示。

与以前的协议模型相比，新的模型引入了 Manager 进程。此进程与 Intruder 进程相互配合以实现新鲜值的循环利用，因此可视为新鲜值的循环机制。上述模型是对 Roscoe 文献的一个扩展和细化，可以证明其满足 PosConjEqT′c 条件：

(1)进程中的数据类型随机数N_a、N_b和密钥K_{ab}均为数据独立类型；

(2)可信代理进程为标准类型进程，满足 PosConjEqT′c 条件；

(3)攻击者进程在明确推理系统中建立，并且其初始知识集不包含对类型T成员(除了常数 C)的非等价测试，满足 PosConjEqT′c 条件。

因此该系统中转换前全部行为可以用经过映射的、相应的有限系统描述。也正基于此，可以在无限新鲜值的产生过程中应用非单映射转换，从而将其简化为相应的有限过程，以形成完整的形式化描述。

对于每一个新鲜值引入对应的 Manager 进程，取消可信代理分发新鲜值的能力，只在单次运行期间存储新鲜值，并要求其在完成协议一次运行时“遗忘”所有存储的新鲜值。当新鲜值 v 不再被可信参与者所知(存储)时，称 v 被“遗忘”。攻击者能够存储在网络中所见的所有新信息。为了能够产生无限的新鲜值，可以对攻击者存储的新鲜值应用 collasping 函数，从而启动所提出的新鲜值循环机制。即当新鲜值 v 在网络中被“遗忘”时可以被循环再利用。一旦该值被循环利用，相应的 Manager 进程可以在网络中再次将其作为新鲜值使用。

3.1 可信进程

FDR(Failures-Divergence Refinement)是由牛津大学欧洲形式化系统有限公司研制的模型检测工具[7]。

该工具基于著名计算机科学家 C.A.R.Hoare 为解决并发问题而提出的 CSP(Communicating Sequential Processes)理论[8]，用于测试 CSP 程序是否满足描述。CSP 中的精化理论可以提出许多正确性条件，例如死锁、活锁、安全性等，这些性质在 CSP 中都可以被方便地编程，这使得 FDR 善于验证系统的抽象特性，其重要性已经在许多应用中体现出来。

CSP 模型由可信主体与入侵者的进程组成，以 NSPK 协议为例，主体 A，B 在 Casper 中被描述成如下形式：

$Initiator(A,N_a)$； $\mathrm{Re}\,sponder(B,N_b)$。

其中 N_a， N_b 代表 A，B 各自使用的随机数。

文献 [4] 中主体 A 的行为被描述为如下进程：

$$
\begin{aligned}
&Initiator(A,N_a)=\\
&\Box B:Agent.env.A.B\rightarrow\\
&\quad send.A.B.\{A,N_a\}_{PK(B)}\rightarrow\\
&\quad\quad \Box N_b:Nonce\bullet\\
&\quad receive.B.A.\{B,N_a,N_b\}_{PK(A)}\rightarrow\\
&\quad send.A.B.\{N_b\}_{PK(B)}\rightarrow SKIP
\end{aligned}
$$

在新的模型中，随机数由随机数进程管理，主体可以执行任意的协议序列。随机数进程知道主体何时需要随机数并马上提供，因此在新的模型中上述描述变成如下形式：

$$
\begin{aligned}
&Initiator(A)=\\
&\Box B:Agent.env.A.B\rightarrow\\
&\Box N_a:Nonce_F\bullet send.A.B.\{N_a,A\}_{PK(B)}\rightarrow\\
&\quad\quad N_b:Nonce\bullet\\
&receive.B.A.\{B,N_a,N_b\}_{PK(A)}\rightarrow\\
&send.A.B.\{N_b\}_{PK(B)}\rightarrow\\
&close.A.Initiator\rightarrow Initiator(A)
\end{aligned}
$$

扩展后的描述主要有三处变动：

①进程被定义为递归进程，表示 Initiator 可以执行无限数量的序列运行；而在之前的模型描述中，进程在 SKIP 终止；

②新鲜 Nonce N_a 不再是参数，而是来自集合 $Nonce_F$（通过定义，进程可以接收该集合中的任意值）；

③添加了 $close$ 事件表示进程重新开始执行初始状态，该事件标志着进程一次运行的结束，在新鲜值的循环机制中发挥重要作用。

3.2 Manager 进程

Manger 进程负责利用有限资源向网络提供无限的新鲜值，需要为每一种数据独立类型分别定义一个 Manager 进程。

在上述的 NS 公钥认证协议和 Yahalom 协议中都定义了一个管理 Nonce 类型值的 Manager 进程——Nonce Manager，Yahalom 协议还需要一个管理会话密钥 SessionKey 类型值的 Manager 进程——SessionKeyManager。

将协议中的每一种数据独立类型 T 所拥有的值分为两类集合。第一类集合称为 Foreground 值，这些值被网络视为新鲜值；第二类集合由 Background 值组成，表示类型 T 旧的或静态的值。当循环利用 Intruder 存储的新鲜值时，将使用这一集合进行映射。

Manager 进程是建模过程中的人造产物，并不代表实际对象而只代表了对于新鲜值的某种操作，主要完成触发“遗忘”值的循环和分发新鲜值的功能。为了对 Manager 进程进行形式化描述，定义两个新的事件：

定义 4：ifclose.(v)：表示最后一个存储 v 的进程是否已经“遗忘”了 v，如果“遗忘”则为 true，否则为 false；

定义 5：replace.(v,b)：表示对 intruder 缓存中含有 v 的所有信息进行操作，将 v 的所有实例替换为 Background 值 b，即完成 Collapsing 函数的非单映射。在相对意义上，v 又会被视为新鲜，即实现了有限值产生无限新鲜值。

同时，使用下述策略确定循环值映射为哪个 Background 值：

对于每一种数据独立类型 T，定义两个不同的 Background 值 T_{BK} 和 T_{BU}。将所有 intruder 所知的 Foreground 值映射为 T_{BK}，不知的映射为 T_{BU}。

通过上述定义和策略研究，描述 Manager 进程如下：

$$
\begin{aligned}
&Manager_T(T_F,T_{BK},T_{BU})\\
&if\ \ ifclose(v:T_F)\ \ then\ \ knownit?v\rightarrow replace.(v,b:T_{BK})\\
&\quad \Box unknownit?v\rightarrow replace.(v,b:T_{BU})
\end{aligned}
$$

其中，T 表示数据类型，T_F 代表该数据类型的 Foreground 集合，T_{BK} 和 T_{BU} 分别代表不同的 Background 值 T_{BK} 和 T_{BU}。

因为对每一个新鲜值的控制都是独立的，为了编译阶段的效率，将其分解为并行结构。在 Yahalom 协议中，假设定义新鲜 Nonce 集为 $\{N_1,N_2\}$，新鲜 SessionKey 集为 $\{K_1\}$，则可建模为下面的并行结构：

$$
\begin{aligned}
Manager=&Manager_N(N_1,N_{BK},N_{BU})\\
&\|\ Manager_N(N_2,N_{Bk},N_{BU})\\
&\|\ Manager_K(K_1,K_{BK},K_{BU})
\end{aligned}
$$

3.3 Intruder 进程

扩展 Intruder 进程，使其与 Manager 进程一起形成数据独立类型的新鲜值循环机制。当启动新鲜值 v 的循环机制时，对缓存中含有 v 的所有信息进行操作，

978-1-935068-01-3 © 2009 SciRes.

将 v 的所有实例替换为 Background 值 b。沿用在 Manager 进程中的定义和研究，Intruder 进程描述如下：

$$\Box say?m:x\cap messages\rightarrow Intruder(x)$$
$$\Box replace.(v,b)\rightarrow$$
$$Intruder(x\,|\,components(x)\neq v,components(x)=b)$$

4. 小结

本文研究的目的是确保使用模型检测工具验证安全协议时协议中每个代理都可以执行无限数量的运行序列，解决模型检测工具的有限检测问题。但是，并行运行的代理实体数量的无限问题还没有得到解决；另外如果在模型中没有发现攻击，不能够证明在更高的并行度不存在攻击，这也是今后的一个研究方向。同时也可以考虑和其它模型检测工具如有色 Petri 网[9]相结合提高验证结果的完备性。

数据独立技术可以在一定的协议范围内有效使用，但是不可以直接运用在包含时戳的协议中，因为此类协议执行的典型操作超出了数据独立范围，如使用对比算子 $X>Y$ 决定时戳 y 是否比时戳 x 旧。这也是未来的研究方向之一。

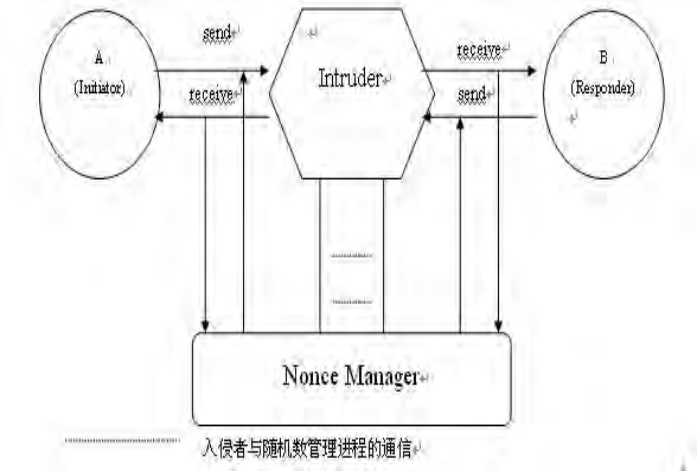

图 1　Yahalom 协议扩展后的模型

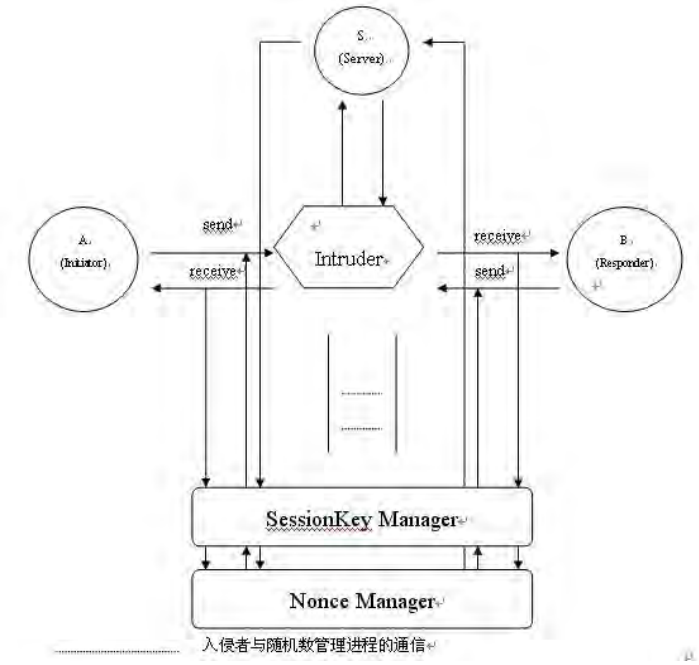

图 2　NS 公钥认证协议扩展后的模型

致谢

感谢解放军理工大学通信工程学院肖军模教授对本文提供的大力帮助。

References (参考文献)

[1]Gavin Lowe. Toward a completeness result for model checking of security protocols[R].In Proceedings of 11th IEEE Computer Security Foundations Workshop, Rockport Massachusetts, June 1998.

[2]G. Lowe. Breaking and fixing the needham-schroder public-key protocol using FDR[R].In Proceedings of TACAS(Tools and Algorithms for the Construction and Analysis of System),volume 1055 pages147-166,Spring Verlag,1996.

[3]Audun, J." Security protocol verification using SPIN" [R].Proceedings of the First SPIN Workshop. INRS-Telecommunications, Montreal, Canada, 1995.

[4]R. S. Lazic, A semantic study of data-independence with application to the mechanical verification of concurrent systems [D], Oxford D. Phil thesis, 1998.

[5]T. C. Newcomb.Model Checking Data-Independence systems with arrays [D]. PhD thesis, Oxford University Computing Laboratory, 2003.

[6]R. S. Lazic T. C. Newcomb, and A. W. Roscoe. On model checking data-independent systems with arrays without reset [J].Theory and Practice of Logic Programming, 4(5-6):659-693,2004.

[7]http://www.fsel.com/.

[8]C. A. R. Hoare. Communicating sequential processes [J].Prentice-Hall International, Englewood Cliffs, NJ, 1985.

Zheng Jun-Jie, Xiao Jun-Mo, Security simulation to security protocol based on coloured Petri net,2006，18（11）：P3294-3296。

[9]郑君杰，肖军模，基于 Petri 网的安全协议安全性验证[J]，系统仿真学报，2006，18（11）：P3294-3296。