A Secure Formal Analysis of E-commerce Protocol Based on the Semi-regular Entities

Yang Jie[1], Wu Tao[2], Chen Guan-qiao[3]

1. 2.3. School of software, South China University of technology, Guangzhou, Guangdong Province, China

Dept. name of organization, name of organization, acronyms acceptable, City, Country

1. yjclear@scut.edu.cn, 2. ssewt@scut.edu.cn, 3. yjclear@gmail.com

Abstract: We use semi-regular entities to denote the entities different from penetrates that participate in the protocol on behalf of themselves such as customers in e-commerce. A method in strand space model is also presented. After descriptions of hash and Diffie-Hellman exchange, semi-regular strand and semi-regular node are added into the model, finally is an attack on the Internet Key Exchange (IKE) protocol as a case.

Keywords: semi-regular entity; strand space; TLS; IKE

基于半规则实体的一种电子商务协议安全分析方法

杨捷[1]，吴涛[2]，陈冠桥[3]

1. 2. 3. 华南理工大学软件学院，广州，中国，510006

1. yjclear@scut.edu.cn, 2. ssewt@scut.edu.cn, 3. yjclear@gmail.com

【摘要】本文用半规则实体代表参与协议并为其利益驱动协议的实体，基于此设想给出一种在串空间模型中的形式化方法。引入 hash 和 Diffie-Hellman exchange 的案例，在模型中加入半规则串和半规则节点。最后基于此方法找到了一种对因特网密钥交换（IKE）协议的攻击，说明该方法合理有效。

【关键词】半规则实体；串空间；TLS; IKE

1 引言

形式化方法可用于验证电子商务协议。所有这些方法都假设：除了侵入者外所有实体都严格遵守分析协议（诚实假设）。我们在没有诚实假设（即使有也很弱）的前提下基于串空间模型分析基础协议。该方法核心是新的概念：半诚实实体（半规则串），代表与侵入者不同的一种不诚实的参与者。改变协议空间后，作为例证分析两个著名的协议：TLS 和 IKE。最后使用新方法找到一种对 IKE 协议的攻击。

2 新的 Hash 函数和半规则串

原始的串空间模型没有包含任何关于 hash 函数的说明。把 hash 作为一个一元操作函数加入到模型中：hash: A → A。这里的 A 是一组术语项，包括了所有可能由协议产生的信息。一个新的侵入者描述也被加入到原来的侵入者描述组中：H: <-g, +hash(g)>。单向 hash 函数一般是公开的。所以任何人都可以计算任何已知原文的 hash 值。类似于关于算子“encr” 和 “join”的公理，有一个关于 hash 的公理。

公理： $\forall$ m1, m2, m3$\in$A, k$\in$K

(1) hash(m1)=hash(m2) $\Rightarrow$ m1=m2

(2) hash(m1)$\neq$ join(m1,m2)

(3) hash(m1)$\neq$ encr(k,m2)

(4) hash(m1)$\notin$ Kp$\cup$T

这里的 K，Kp，T 分别表示一组密钥，一组侵入者已知的密钥，和一组原子信息。(1)说明 hash 是一个单射操作。(2)和(3)表明这三个操作的结果集是两两不相交的。(4)表明了 hash 的任何结果都不是原子信息。考虑到 hash 函数产生密钥的可能结果，我们没有给定 hash(m1)$\notin$K$\cup$T。接着给出两个关于 hash 的定义。

定义 2.1 S$\subseteq$A, W[S] 是满足下面两个条件的最小集：

(1) $\forall$ g$\in$S, then g$\in$W[S] (2) $\forall$ g$\in$W[S], h$\in$A, then gh, hg$\in$W[S]

上述定义表明 hash 函数的定义域包括了 S 集中的任何元素。

定义 2.2 S$\subseteq$A。H[S]是满足下述条件的最小集：$\forall$ g$\in$W[S], 则 hash(g)$\in$H[S]。H[S]是定义域为 W[S]的 hash 函数值域。由本定义知 hash 是满射。因此 hash 是 W[S]与 H[S]之间的双射。在本文的其余部分，我们分别把 H[{g}]和 W[{g}]表示为 W[g]和 H[g]。最后我们证明一个与 hash 函数相关的入侵者的结论。

定理 2.1 假设 S⊆A, x∈H[S], K 是所有密钥的集合。只要入侵者节点 p 是 IK[x]（表示为 I[x]）的入口点，则 p 属于入侵者串 H。换言之，除了 H 节点，x 不能源于其他任何侵入者节点。

证明：假设 m 是一个入侵者节点且为 I[x]的入口节点。考虑 H 串外的所有类型的入侵者串，所有串的类型具体请见参考文献【1】。因为入口点必须为正，所以 m 不可能在 F 类型的串中。因此考虑余下类型：

K，M．根据公理(4)，m 不属于这两种类型的任何串；

C，E．根据公理(2)和(3)，m 也不属于这两种类型的任何串；

T，S，D．根据定义 I[x]，只要 m 属于这三种类型的任何串，必然会有另一个节点 m'也落入 I[x]中并且先于 m。这与 m 的最小性矛盾。Hash 值不能由 hash 外的任何其他操作产生。因此若一个入侵者节点产生一个形如 hash(g)（其中 g∈A）的信息，则其入口点一定是在 H 类型的串上。□

1976 年 Diffie 和 Hellman 提出了 Diffie-Hellman 交换(D-H 交换)。它能应用于公共通信通道上，即使在网络上通信的每个信息都被窃听，仍能保持两个实体会话密钥交换的机密性。其次它为会话密钥提供了完美的前向保密（PFS），这意味着解开单个密钥只能访问到被那个密钥所保护的数据，其他的数据仍然处于保密状态。基于这些优点，D-H 交换被广泛应用于密钥协定协议的设计。我们应该在串空间模型中加入对 D-H 交换的描述。

定义 2.3 在一次 D-H 交换中，若两个实体之间交换的数据为 g 和 h，则交换的结果表示为 dh(g，h)。g 和 h 是公共的，且他们之间的顺序并不重要，所以可得公理：dh(g，h)=dh(h，g)。我们可以根据 dh(g，h)考虑谁可以知道结果。

定义 2.4 若实体 X 知道信息 m，则我们认为 m∈KNOWNX 。

命题 2.1 已知一个实体 X，一个 D-H 交换，交换数据为 g 和 h，若 g，h∈KNOWNX ，且 g 或 h 中至少有一个由 X 产生，则 dh(g，h)∈KNOWNX 。若 I[m]的入口点是在与 X 相关联的串中，则认为信息 m 由实体 X 产生，或认为实体 X 发起信息 m。D-H 交换的目的是确保交换双方保持一致的机密，所以给出定义 2.4 用于描述实体的知识集。

命题 2.1 说明了实体怎样可以知道 D-H 交换的结果（机密）。根据 D-H 交换的定义可得命题 2.1 显然成立。这个推理给出了一些实体想要知道 dh(g，h)所需的充分条件，但并非必要的。例如一个实体可以通过委托或者协作的方法使其结果（dh(g，h)）被另一个实体知道。考虑加密协议（尤其是电子商务协议）与实体应该遵循的条件之间的关系。串空间模型中，所有的串被划分成规则串和侵入者串，相应地所有节点被划分成规则节点和侵入者节点。规则串都是由某些协议的规范严格产生的，对于侵入者串并无任何限制，一些协议的特殊结果直接由规则串导致。因此我们有基本假设：协议中的每个实体都想要获取最大的利益。基于此假设，除了规则串和侵入者串，引入一种新的串：半规则串。

定义 2.5 任意一个带有某些条件约束的串定义为半规则串。对应于半规则串的节点为半规则节点。若半规则串的约束达到了某些协议的规范，则半规则串就代表规则串。若半规则串没有任何约束，则半规则串代表侵入者串。半规则串代表会以自己的利益为活动目的的非侵入者实体。半规则串的概念并不受限于串空间模型。我们利用半规则实体进行协议分析，步骤如下：

(1) 构造分析协议的渗透串空间。通常包含带有某些约束的半规则串。

(2) 在构造的渗透串空间中验证协议的设计者所声明的某些属性。

(3) 验证半规则串上约束的合理性。

对于一个带有两个实体的协议，我们有四种可能的入侵串空间。PROTOCOL_NAMEI,R=Init∪Resp∪PT；PROTOCOL_NAMEI',R=Init'∪Resp∪PT；PROTOCOL_NAMEI,R'=Init∪Resp'∪PT；PROTOCOL_NAMEI',R'=Init'∪Resp'∪PT。其中 PT 包含所有入侵者串，Init (Resp)是规则发起者（回应者）串的集合，Init' (Resp')是半规则发起者（回应者）串的集合。每个可能的串空间的证明都与原始模型的证明相似。在必需的第三步中的原理是上面步骤的基础假设。若有任何约束与实体的利益相矛盾，则基于此约束的结果不稳定。现在给出半规则串的第一个约束。

定义 2.6 假设 s 是一个串，若∀x∈LS ，则 s 是正常的，s 中没有任何一个节点是 I[x]的入口点。

3 TLS 和 IKE 应用实例

传输层安全（TLS）协议是安全套接层（SSL）协议的一个版本。客户计算机和服务器通过他们交换临时和计算会话密钥。

我们的 TLS 抽取版本【4】省去了所有的证明（假设每个实体都知道对方的公钥）并把一些信息整合到一起。由于没有考虑重用某些会话，客户端 hello 消息的“会话标识符”也被省去。对 TLS 协议的描述如下：

I → R: I, Ni, Pi

R → I: Nr, Sid, Pr

I → R: {PMS}kr, {hash(Nr, R, PMS)}ki-1, {Finished_I}ksi

R → I: {Finished_R}ksr

其中，Ni 和 Nr 是现时，Sid 是由回应者产生的用于识别会话的唯一性的会话标识符，Pi 和 Pr 包括了为 I 和 R 加密和压缩协议所需的参数。PMS 是用于产生会话密钥的 pre-master-secret。在这里没有危及安全性

978-1-935068-01-3 © 2009 SciRes.

的情况下，我们省去了 master-secret 并直接使用 PMS。Ksi，ksr，Finished_I 和 Finished_R 的详细说明如下：
ksi = hash(PMS, Ni, Nr, 0)
ksr = hash(PMS, Ni, Nr, 1)
Finished_I = hash(PMS, Sid, Ni, Pi, I, Nr, Pr, R, 0)
Finished_R = hash(PMS, Sid, Ni, Pi, I, Nr, Pr, R, 1)

定义 3.1 假设 M1=INiPi，M2= NrSidPr，M3={PMS}kr {hash(Nr，R，PMS)}ki-1{Finished_I}ksi，M4={Finished_R}ksr

（1）Init [I, R, Sid, Ni, Nr, Pi, Pr, PMS] 是轨迹为(+M1, -M2, +M3, -M4)的串。

（2）Resp [I, R, Sid, Ni, Nr, Pi, Pr, PMS] 是轨迹为(-M1, +M2, -M3, +M4)的串。

（3）Init' [I, R, Sid, Ni, Nr, Pi, Pr, PMS] 是轨迹为(…, +M1, …, -M2, …, +M3, …, -M4, …)的串。

（4）Resp' [I, R, Sid, Ni, Nr, Pi, Pr, PMS] 是轨迹为(…, -M1, …, +M2, …, -M3, …, +M4, …)的串。

其中对 $\forall s \in$ Init'， x 为任何一个 s 所发送消息，都不能使 $x \in$ I[PMS]，只能得到 $x \in$ {PMS}kr。对于 $\forall s \in$ Resp'， x 为任何一个 s 所发送消息，都不能得到 $x \in$ I[PMS]。显然可以得到 Init $\subseteq$ Init' 和 Resp $\subseteq$ Resp'。在此有两个关于半规则串的约束。首先由 Init' (Resp')的定义可知，发起者（回应者）循序渐进地执行 TLS 协议。只有前一步执行完下一步才开始。这样能保护他们不会被其他实体误用。其次由于 PMS 是密钥，为防止被滥用，除非是协议的要求（在 Init' 中的串发送{PMS}kr），发起者（回应者）不会发送 I[PMS]中的任何消息。这些约束显然合理。接着可以得到四个串空间：TLSI,R，TLSI',R，TLSI,R'，TLSI',R'。并可以直接得到以下命题。

命题 3.1 若 $s \in$ Init [I, R, Sid, Ni, Nr, Pi, Pr, PMS]，则 Ni 起源于<s,1>，PMS 起源于<s,3>。

命题 3.2 若 $s \in$ Resp [I, R, Sid, Ni, Nr, Pi, Pr, PMS]，则 Nr 起源于<s, 2>。

在此省略对这两个命题的证明。只有当 s 属于 Init 或 Resp 时这两个命题才成立。若 s 属于 Init'或 Resp' 则他们就不再成立。即若 s 属于 Init'，就不能得出 Ni 源于<s,1>或 PMS 源于<s,3>。若 s 属于 Resp'，则不能得到 Nr 源于<s, 2>。有些在规则串中可以直接得到的结论在半规则串中却不成立。

定理 3.1 假设 C 是 TLSI,R'中的一个束，束的定义详见 TLS 抽取版本【4】（串空间 TLSI,R'），只有 R 知道 kr-1。PMS 是唯一的发起源。在 Resp'中的任何串都是正常的。若 $s \in$ Init [I, R, Sid, Ni, Nr, Pi, Pr, PMS]有 C-Height 4，则 Finished_R 是由 R 在串 $t \in$ Resp' [I, R, Sid, Ni, Nr, Pi, Pr, PMS]中产生。

证明：TLSI,R'是由 Init，Resp' 和 PT 所组成的。Init 是一组规则串。可知在 Init 中没有规则节点是 I[Finished_R]的入口点。由于条件的限制，不会有半规则节点在 W[PMS]的入口点中。由于协议的定义，没有规则节点是 W[PMS]的入口点。此外，因为只有 R 知道 kr-1，任何在 Resp'中的串都是正常的，并且 PMS 是唯一的发起源，所以没有侵入者节点是 W[PMS]的入口点。因此，在 C 中没有任何 W[PMS]的入口节点。由定理 2.1 可知，没有侵入者节点是 I[Finished_R]的入口点。只有半规则节点才有可能是 I[Finished_R]的入口点。因为只有 I 或 R 知道 PMS，当 I 是规则的并且不能产生 Finished_R，所以 Finished_R 是由 R 所产生的。

□

根据 hash 的特性，我们可以从定理 4.1 中得到的结论是，I 和 R 在 Sid, Ni, Nr, Pi, Pr 和 PMS 上都是一致的。而且 $PMS \notin$ KNOWNX ($X \neq I,R$)，也就是机密性是被保护的。

定理 3.2 假设 C 是 TLSI',R 中的一个束，只有 I 知道 ki-1，只有 R 知道 kr-1。在 C 中 PMS 是唯一的发起源且源于 I。在 Init'中的任何串都是正常的。若 $s \in$ Resp [I, R, Sid, Ni, Nr, Pi, Pr, PMS] 有 C-Height 4，则 Finished_I 是由 I 在串 $t \in$ Init' [I, R, Sid, Ni, Nr, Pi, Pr, PMS]中产生。

证明过程同定理 3.1，此处略。

虽然在规则串中 Ni 或 Nr 是新鲜的，但在这两个定理中并没有要求这个特性。事实上他们是用以确保所产生的会话密钥在之前没有被使用过。在命题 4.2 中 PMS 必须由 I 产生。否则 PMS 可能被前面的一些实体所知道。那将会出现对协议的 man-in-the-middle 攻击（在后面举例说明）。I 认为其与 R 有两个共同的会话密钥 ksi 和 ksr，同时，R 认为其与 I 有两个共同的会话密钥 ksi'和 ksr'。我们分别地在 Finished_I，Finished_R,，ksi 和 ksr 中用 Ni'替代 Ni，从而得到 Finished_I'，Finished_R'，ksi' 和 ksr'。

IKE 协议的设计目的是在因特网中交换密钥信息。在第一阶段定义主要模型和主动模型，建立安全关联（SAs）和得到共享密钥。在第二阶段定义快速模型，把 SAs 转交给上层应用。具有加密公钥的主动模型（IKEAP）协议是 IKE 协议的一个子协议。

定义 3.2 假设 M1= CiSAiKEi{I}kr{Ni}kr， M2= CiCrSArKEr{R}ki{Nr}kiHASH_R，M3= CiCrHASH_I

(1) Init [I, R, Ci, Cr, SAi, SAr, KEi, KEr, Ni, Nr] 是轨迹(+M1, -M2, +M3)的串。

(2) Resp [I, R, Ci, Cr, SAi, SAr, KEi, KEr, Ni, Nr] 是轨迹(-M1, +M2, -M3)的串。

(3) Init' [I, R, Ci, Cr, SAi, SAr, KEi, KEr, Ni, Nr] 是轨迹(…, +M1, …, -M2, …, +M3, …)的串。

(4) Resp' [I, R, Ci, Cr, SAi, SAr, KEi, KEr, Ni, Nr] 是轨迹(…, -M1, …, +M2, …, -M3, …)的串。

除了侵入者没有其他实体 X 愿意发起 HASH_Y ($X \neq Y$)。$\forall s \in$ (Init' $\cup$ Resp')，s 不会发送任何信息 m，使得 $m \in$ I[dh(KEi, KEr)]。有四个 IKEAP 的串空间：IKEAPI,R, IKEAPI',R, IKEAPI,R', IKEAPI',R。这些约束条件是合理的。

命题 3.3 若 s∈Init [I, R, Ci, Cr, SAi, SAr, KEi, KEr, Ni, Nr]，则 Ni 和 KEi 起源于<s,1>。

命题 3.4 若 s∈Resp [I, R, Ci, Cr, SAi, SAr, KEi, KEr, Ni, Nr]，则 Nr 和 KEr 起源于<s,2>。

若 s 属于 Init'或 Resp'，则命题 3.3 或命题 3.4 将不再成立。

定理 3.3 设 C 是 IKEAPI,R 中的一个束，Ni∉KNOWNP。若 s∈Init [I, R, Ci, Cr, SAi, SAr, KEi, KEr, Ni, Nr] 有 C-Height 3，则 HASH_R 是由在串 t∈Resp' [I, R, Ci, Cr, SAi, *, KEi, KEr, Ni, Nr]中的 R 所产生。

定理 3.4 假设 C 是 IKEAPI',R 中的一个束，Nr∉KNOWNP。若 s∈Resp [I, R, Ci, Cr, SAi, SAr, KEi, KEr, Ni, Nr] 有 C-Height 3，则 HASH_I 是 I 在串 t∈Init' [I, R, Ci, Cr, SAi, *, KEi, KEr, Ni, Nr]中产生的。

假设侵入者不直接知道 Ni 和 Nr。定理 3.3 和定理 3.4 认为即使另一侧是半规则的，回应消息也必须由它自己产生。因此可得 I 和 R 中的 Ci, Cr, SAi, KEi, KEr, Ni 和 Nr 是一致的。但并不包括 SAr，这预示了协议中的一个缺陷。为了达到其目的，IKEAP 协议必须保证通信的两个实体取得一致的 dh(KEi, KEr)。由命题 3.1 可知，一个实体知道了 KEi 和 KEr 并不意味着其也知道 dh(KEi, KEr)。但令人遗憾的是，尽管在规则串中 Nr (Ni)是由 R (I)所产生的（据命题 3.2 和 3.3），但在定理 3.3（3.4）中并不能得到此结论。由此我们找到一个对 IKEAP 协议的攻击。

I → R: Ci, SAi, KEi, {I}kr, {Ni}kr;

R → V: Ci, SAi, KEi, {R}kv, {Ni}kv;

V → R: Ci, Cr, SAr, KEr, {V}kr, {Nr}kr, HASH_V;

R → I: Ci, Cr, SAr, KEr, {R}ki, {Nr}ki, HASH_R;

I → R: Ci, Cr, HASH_I;

R → V: Ci, Cr, HASH_R

4 结语

在因特网上没有任何方法可以保证其他实体是诚实的。传统分析中的诚实实体也可能没有严格遵循协议规范。目前已有一些关于第二类协议分析的研究，包括对恶意参与者的研究。本文提出一种新方法分析基础协议，为了描述 TLS 和 IKE 在模型中加入 hash 函数和 D-H 交换。把所有的实体分为三类：诚实实体，半诚实实体，入侵者。当证明发起者的保证时使用 TLSI,R'串和 IKEAPI,R'串。深入分析 TLS 并得到确保协议安全所需的精确条件。找到一种在原始串空间模型中无法发现的 IKE 协议攻击。

致谢

本文工作由国家自然科学基金项目（60873078）、华南理工大研究生重点课程建设项目（B07Y3080020）、校精品课程 x2rj-Y1080150、Y1080160 资助。

References（参考文献）

[1] J. Thayer, J. C. Herzog, and J. D. Guttman. Strand spaces: Proving security protocols correct, *Journal of Computer Security*, 1999. 15

[2] T. Dierks and C. Allen, The TLS Protocol Version 1.0, *RFC 2246*，January 1999.

[3] D. Harkins and D. Carrel, The internet key exchange, *RFC 2409*, 1998.

[4] L.C. Paulsen, Inductive Analysis of the Internet Protocol TLS, *Technical Report 440*, Computer Laboratory, University of Cambridge, England, December 1997.

[5] J. Zhou, Further Analysis of the Internet Key Exchange Protocol, *Computer Communications*, 23(17): 1606--1612, Elsevier, November 2000.