J. Lopez, S. Matamoros, J. J. Ortega, and J. M. Troya, "Observaciones sobre las técnicas de análisis formal de protocolos de seguridad", *VII Reunión Española sobre Criptología y Seguridad de la Información (VII RECSI)*, pp. 471-483, 2002.
NICS Lab. Publications: https://www.nics.uma.es/publications

OBSERVACIONES SOBRE LAS TÉCNICAS DE ANÁLISIS FORMAL DE PROTOCOLOS DE SEGURIDAD

Javier López, Sonia Matamoros, Juan J. Ortega, José M. Troya
Dpto. Lenguajes y Ciencias de la Computación
Universidad de Málaga
e-mail: {jlm, sonia, juanjose, troya}@lcc.uma.es

Keywords: Protocolos de seguridad, Análisis Formal, Confidencialidad, Autenticación, No-repudio, Anonimato

ABSTRACT

La aplicación de los métodos formales para el diseño y análisis de sistemas críticos está ampliamente aceptada en el desarrollo de estos sistemas. Los protocolos de seguridad abordan el objetivo de garantizar servicios y derechos como el de la confidencialidad de los datos personales o el de garantizar la identidad de acceso a un sistema. Por lo tanto, ya que un protocolo de seguridad es un sistema crítico, es necesario utilizar métodos formales para su diseño y análisis. Debido a las características especiales que presentan este tipo de protocolos, se deben utilizar métodos que no son los tradicionales utilizados para los protocolos de comunicaciones, sino que deben utilizarse otros específicos. En este artículo vamos a hacer un estudio de las principales propiedades de seguridad que poseen los protocolos criptográficos y de la manera de aplicar los métodos formales en su diseño y análisis.

1. INTRODUCCIÓN

En la ingeniería de protocolos han venido utilizándose unos determinados métodos para el desarrollo de protocolos de comunicaciones que han sido llamados métodos tradicionales. En este tipo de métodos existe una serie de etapas que se corresponden con el ciclo de vida del software. Una de ellas es la de depuración de la implementación para detectar posibles errores. Solventar un error encontrado en esta fase resulta un proceso muy costoso ya que se tendría que regresar a la etapa de donde procede. Por ejemplo, el error puede haber sido fruto de una mala especificación en cuyo caso habría que volver a especificar el sistema y esto implica cambios en el diseño y la implementación del mismo.

Uno de los objetivos, y quizás el más importante, que persigue la utilización de métodos formales [23] en el desarrollo de software es precisamente evitar este tipo de situaciones. Estos métodos se corresponden con la aplicación de las matemáticas al proceso de especificación y diseño de sistemas. Con su aplicación se pretende conseguir el mismo nivel de precisión y abstracción que con una metodología puramente matemática de forma que se obtengan especificaciones que sean consistentes, no ambiguas, completas y mínimas.

A partir de una especificación que cumpla las propiedades mencionadas se puede diseñar, validar [7] y documentar el sistema con un menor riesgo de error. La semántica de los formalismos proporciona reglas precisas de interpretación que evitan muchos problemas encontrados en una especificación en lenguaje natural. Es decir, un lenguaje con ricas facilidades de estructuración puede producir mejores especificaciones.

Además de producir una buena especificación, la aplicación de métodos formales proporciona la base para un procesamiento automático mediante el cual se da la posibilidad de generar una implementación del sistema a partir de su especificación.
En el desarrollo de protocolos de seguridad, además de encontrarnos con los problemas mencionados, existen otras consideraciones a tener en cuenta:

- Las propiedades que tienen que garantizar son extremadamente delicadas.
- Habitan en entornos complejos y hostiles. Aparece la figura del intruso que intentará "romper" la seguridad.
- Conocer las capacidades del intruso es muy complicado.
- Debido a su naturaleza, estos protocolos llevan asociados un alto grado de concurrencia.

Por otro lado, un protocolo de seguridad resulta ser un componente crítico en cualquier arquitectura de seguridad distribuida. Un mal diseño del mismo puede causar que presente vulnerabilidades ante ciertos ataques.
De este modo, por su naturaleza, los protocolos de seguridad resultan ser los candidatos ideales a los cuales aplicar técnicas rigurosas de especificación y diseño.
Durante muchos años se han estado utilizando este tipo de formalismos para propósitos generales pero sólo en la última década se han empezado a utilizar en los protocolos de seguridad.
Las técnicas de análisis de protocolos de seguridad [13][14] se pueden dividir en tres tipos. La primera utiliza lógica de primer orden diseñada específicamente para el análisis de protocolos de seguridad. La más significativa es la lógica BAN [3], aunque también se han definido extensiones como la lógica GYN [6]. Esta técnica está en desuso, sobre todo por su dificultad de aplicación. La segunda es la que utiliza los probadores de teoremas (*theorem-proving techniques*). Estos hacen uso de reglas de reescritura para el análisis de las propiedades de seguridad. Los más destacados son la traducción de CAPSL [4] a PVS y Maude, CASRUL [18][19] e ISABELLE. El último tipo se basa en técnicas de exploración del espacio de estado, sobre todo *model-checking* [11][1]. Es la que mejores resultados han producido hasta ahora, ya que aunque no puede garantizar en la mayoría de los casos que no existe un fallo en el protocolo debido a la no completitud del problema[20][10] , si que la exploración es lo suficientemente amplia como para deducirlo. En este tipo se enmarcan el NRL Protocol Analyzer [12], Casper [21], Athena, LOTOS [6] y SDL [17][9]

2. PROTOCOLOS DE SEGURIDAD

Como cualquier protocolo, un protocolo de seguridad [15] consiste en una secuencia de interacciones entre entidades para conseguir un resultado final. Pero, además, el objetivo fundamental que persigue este tipo de protocolos es proporcionar servicios de seguridad en un sistema distribuido [21]. Entre estos servicios se encuentran la autenticación de los agentes que intervienen, el establecimiento de claves de sesión y la aportación de confidencialidad, integridad, anonimato y no-repudio. Para conseguir estos servicios se hace necesario el intercambio de mensajes entre los agentes implicados y, en algunos casos, la participación de una tercera parte confiable.
El servicio de confidencialidad, que ofrecen muchos protocolos de seguridad consiste en hacer que el intruso sea incapaz de deducir nada acerca de la actividad de los usuarios legítimos. Este tipo de servicio se requiere en aquellas situaciones en las que la información que se están intercambiando dos agentes es sensible, es decir, datos de relevancia.

En el caso de la autenticación lo que se pretende es que los agentes que intervienen en una comunicación estén seguros de la identidad de la persona con la que se están comunicando.

Un agente, aún estando seguro de la identidad del agente con el que está manteniendo un diálogo, puede necesitar tener evidencias de que se producen determinadas situaciones durante dicha comunicación. En este caso se necesita que el protocolo utilizado ofrezca el servicio de no-repudio. Este servicio se confunde a veces con el de autenticación pero su misión es otra completamente distinta. Lo que pretende es que un agente no pueda alegar que no ha recibido o enviado algo cuando sí que ha sucedido dicha situación.

El servicio o propiedad de anonimato es necesario en aquellas situaciones en las que se desea que no conozca la identidad del agente que ha llevado a cabo una determinada acción. Puede ser muy útil en compras a través de la red donde no se hace necesario el conocimiento del usuario que está comprando. Es lo contrario del no-repudio.

Normalmente un protocolo proporcionará algunos de estos servicios y no necesariamente todos los mencionados. Por ejemplo, en el caso de los protocolos dedicados al intercambio de claves lo que se pretende es establecer una clave que será utilizada por dos agentes que desean comunicarse de forma segura. Para ello será necesario que dicha clave se envíe a la persona apropiada además de hacerlo de forma confidencial.

En cambio, en los protocolos de autenticación lo que se persigue es que los agentes implicados estén seguros de que la persona con la que se están comunicando sea quien dice ser.

El proporcionar estos servicios representa una ardua tarea ya que el protocolo se encuentra en un entorno hostil donde habita algún posible intruso que intentará realizar ataques a la seguridad del sistema. Existe mucha diversidad de ataques de los cuales nombraremos solamente algunos que pueden servir de ejemplo de las debilidades que pueden presentar los protocolos si no están diseñados correctamente.

El ataque del hombre en el medio (*man-in-the-middle*) consiste en que el intruso se interpone entre los dos agentes que se están comunicando para interceptar información de importancia. Para ello necesita "engañar" a algunos de los agentes honestos. Este tipo de ataque puede evitarse si los agentes se autentican previamente.

Uno de los agentes honestos puede ser inducido a realizar la ejecución de una determinada parte del protocolo lo que ayudaría al intruso a obtener algunos datos importantes. Este tipo de ataque es llamado ataque de oráculo (*oracle*).

En el caso de un ataque de repetición (*replay*) el intruso se encuentra atento a la ejecución de un protocolo y almacenar algunos de los mensajes que se envían. En alguna ejecución posterior del mismo protocolo podría enviar uno de esos mensajes llegando a confundir a los agentes implicados.

También existen ataques a los algoritmos algebraicos utilizados para el cifrado de mensajes pero estos no son evitables mediante métodos formales mientras que los expuestos, entre otros, sí que son salvables.

Las técnicas formales que analizan estas propiedades describen el protocolo de seguridad y lo evalúan considerando que existe un medio controlado por el intruso o atacante. Además considera que las operaciones criptográficas son perfectas, por lo que no considera ataques por criptoanálisis.

3. ANÁLISIS DE LA PROPIEDAD DE CONFIDENCIALIDAD

La propiedad de confidencialidad es la cualidad de prevenir que un intruso sea capaz de deducir el texto en claro de los mensajes que se intercambian entre agentes honestos. Si en algún estado del protocolo se produce que cierto dato aparece en la base de conocimiento del intruso, esto quiere decir que se ha vulnerado tal propiedad de dicho dato. La base de conocimiento del intruso la constituye los datos de los mensajes que se intercambian más los que el intruso puede deducir.
En principio, para analizar si un protocolo proporciona el servicio de confidencialidad de los datos, se va a considerar que se realiza un cifrado "perfecto" por lo que para deducir un dato cifrado con una determinada clave, se debe poseer dicha clave.
Se puede distinguir entre dos tipos de secreto; el primero sería el referente a elementos externos al protocolo pero que intervienen en el mismo (*strong secrecy*). Por ejemplo, una clave como entrada del protocolo que va a hacer uso de ella. Y el segundo tipo de secreto sería el concerniente a elementos creados en la ejecución del protocolo como, por ejemplo, una clave de sesión.
Para la evaluación de la propiedad [5] se comprueba que los datos cifrados no se pueden deducir mediante alguna regla de deducción. Las reglas de deducción principales son:

1. {{M}*sk*}*sk'* = M, *sk* es una clave simétrica y *sk'* indica que es descifrar
2. {{M}*pk*}*pk'* = {{M}*pk'*}*pk* = M., *pk* es la clave pública y pk' es la privada
3. { {M}*key1* }*key2* = { {M}*key2* }*key1*

Un ejemplo de ataque a la propiedad de secreto lo vemos en al análisis del protocolo RSA.

```
1. A → B: {M}Ka
2. B → A: {{M}Ka}Kb
3. A → B: {{{M}Ka}Kb}Ka'
```

El ataque consiste en averiguar el secreto que es le mensaje *M*. Para ello se va a simular un intruso que utilizando las reglas 2 y 3 va a deducirlo. La secuencia del ataque es la siguiente:

```
I es el intruso
Ki es la clave pública del intruso

    1. A → I: {M}Ka
    2. I → B: {Mi}Ki # se engaña a B
    3. B → I: {{Mi}Ki}Kb
    4. I → A: {{M}Ka}Ki
    5. A → I: { {{M}Ka}Ki}Ka' # Aplicamos regla 2 y 3
    6. I → B: { {{M1}Ki}Kb}Ki'
```

En el paso 5 se deduce *M* que es el secreto que se intenta intercambiar. La capacidad de deducción la debe tener el intruso.

4. ANÁLISIS DE LA AUTENTICACIÓN

La autenticación de entidades se basa en que seamos capaces de estar seguros de que todos los mensajes que recibimos son de un determinado agente (autenticación en origen). Si aceptamos mensajes que son originados por un agente deshonesto (normalmente el intruso) y finalizamos el protocolo, entonces se produce el fallo en la autenticación. A este tipo de fallo se le llama fallo de correspondencia (*correspondence*) o precedencia (*precedence*), ya que se modela haciendo que un agente termine el protocolo correctamente mientras que el otro agente ni siquiera lo empiece.
De este modo, un protocolo de autenticación asegura que un agente *A* ha establecido una comunicación con el agente *B*. Para asegurar la identidad se comparten algunos datos (claves, variables,...)
En la ejecución de un protocolo de autenticación se pueden producir dos situaciones:

1- El agente *A* ha completado la ejecución del protocolo con el agente *B*.
2- El agente *A* está ejecutando el protocolo aparentemente con el agente *B*.

En el proceso de autenticación hay que distinguir el punto en el que cada agente tiene total seguridad de que se está comunicando con el agente deseado.
La autenticación [24] se lleva a cabo mediante una secuencia ordenada de mensajes donde los agentes se intercambian una serie datos ordenados convenientemente. Si esa secuencia se ve alterada se puede pensar que el protocolo no se ha ejecutado correctamente. Por lo tanto, para verificar que un protocolo autentica a los agentes que se están comunicando, hay que comprobar que esa secuencia de mensajes no se altere.
El caso de una ejecución del protocolo donde aparecen múltiples agentes se debe modelar indicando un número finito de agentes ya que es imposible probar todas las posibilidades. Este es el problema de la incompletitud de los protocolos, existen varios trabajos relacionados [10].
Un ejemplo de protocolo de autenticación es el *Encrypted Key Exchange* (EKE) [2]. El objetivo de este protocolo es el de asegurar que *A* está comunicándose con *B* y que *B* lo está con *A*. Al final del intercambio de mensajes tanto *A* como *B* deben conocer *Na* y *Nb*, y por lo tanto, se demuestra la autenticación de ambos. La secuencia de intercambio de mensajes es la siguiente:

1. A → B: {Ka}P
2. B → A: {{Re}Ka}P
3. A → B: {Na}Re
4. B → A: {Na,Nb}Re
5. A → B: {Nb}Re

El ataque a este protocolo se produce cuando se ejecutan dos sesiones paralelas y el intruso envía los mensajes de una sesión a la otra. Al final, termina los agentes *A* de la primera sesión y *B* de la segunda sesión, en realidad *"b"* se conecta consigo mismo. La secuencia de ataque es la siguiente:

Primera sesión
A1=b
B1=a
Segunda sesión
A2=a
B2=b

Intercambio de mensajes:
1. A1 → Intruso: {Kb}P
2. A2 → Intruso: {Ka}P
3. Intruso → B2: {Kb}P # redirecciona a B2 todos los mensajes
4. Intruso → B2: {Ka}P
5. B2 → Intruso: {{Re}Kb}P
6. Intruso → A1: {{Re}Kb}P # redirecciona a A1 todos los mensajes
7. A1 → Intruso: {Na}Re
8. Intruso → B2: {Na}Re
9. B2 → Intruso: {Na, Nb}Re
10. Intruso → A1: {Na, Nb}Re
11. A1 → Intruso: {Nb}Re
12. Intruso → B2: {Nb}Re

5. ANÁLISIS DEL NO-REPUDIO Y EL ANONIMATO

En protocolos más avanzados como son los de comercio electrónico, además de las propiedades vistas hasta ahora, existen otras que se hacen necesarias para que las transacciones que se producen en la red sean totalmente seguras. Este es el caso de las propiedades de no-repudio y anonimato.
La propiedad de no-repudio persigue que ninguna de las dos partes involucradas en una transacción se desdiga acerca de alguna acción que se haya llevado a cabo durante dicha comunicación. Es muy utilizada para el caso de transacciones comerciales en Internet como son las compras en tiendas virtuales. En este tipo de transacciones normalmente interviene alguna tercera parte confiable (*TTP*) que será la que supervise todo el proceso y muestre evidencias de los pasos llevados a cabo.
Existen distintas clases de no-repudio las cuales las describimos a continuación y para ello utilizaremos un pequeño ejemplo en el que un usuario *A* desea realizar una compra a través de Internet en una tienda *B*.

- **No-repudio de origen**. Es la prueba producida por el originario de un mensaje. De este modo, si *A* realiza una petición a la tienda *B* ésta tendrá una prueba de que ha sido *A* quien realmente le ha realizado una consulta.
- **No-repudio de recepción**. Es la prueba producida por el destinatario de un mensaje. Así, cuando *B* recibe una petición de un usuario *A* éste deberá poseer una prueba de que su petición ha sido recibida por la tienda.

Si para llevar a cabo la transacción interviene una *TTP* aparecen dos nuevos tipos de no-repudio.

- **No-repudio de envío**. Es la prueba producida por la *TTP* cuando recibe un mensaje de alguna de las dos partes involucradas. Esta prueba es entrega al originario del mensaje.

- **No-repudio de entrega**. Es la prueba producida por el destinatario del mensaje. En vez de ser entregada al originario del mismo, esta prueba se entrega a la *TTP*.

Bien es cierto que siempre se hará necesaria la intervención una tercera parte confiable ya que, por ejemplo, el no-repudio de origen mencionado se puede conseguir mediante una firma digital por parte del originario del mensaje. Para que esta firma tenga validez tendrá que estar respaldada por un certificado digital emitido por alguna autoridad de certificación que no deja de ser una *TTP*.

La propiedad de no-repudio [22][25] se analiza garantizando que algunos eventos sucedan antes que otros al igual que ocurría para la verificación de la autenticación. Los eventos que se analizan son aquellos que representan las evidencias necesarias para que ninguna de las partes se desdiga.

Estas evidencias suelen ser mensajes cifrados por claves las cuales sólo son conocidas por los agentes implicados. De este modo, si un agente *A* recibe un mensaje cifrado con la clave que comparte con otro agente *B* entonces puede demostrar que el mensaje ha sido enviado por *B* por lo que éste último no puede negar dicha evidencia.

A diferencia con el análisis de la autenticación, el sistema debe considerar que puede ser uno de los agentes que suponemos honestos el que tiene intenciones de engañar. Debido a esto, no se podría utilizar el esquema en el que un intruso externo es el que controla el medio de transmisión.

La propiedad de anonimato consiste en la protección de la identidad de los agentes con respecto a determinados eventos o mensajes pertenecientes a la ejecución de un protocolo. Se basa en que un dato puede ser originado por uno u otro agente indistintamente.

El anonimato se puede presentar desde dos puntos de vista. Por un lado, se puede hacer que uno de los agentes o ambos sean anónimos a un observador externo. Y por otro, se puede proporcionar anonimato a un agente con respecto al agente con el que se está comunicando.

Para proporcionar el anonimato de los agentes implicados en un protocolo existen varias técnicas como son la ocultación, el enmascarado o el renombrado. El protocolo debe estar diseñado de tal modo que, aún proporcionando el anonimato de los agentes, funcione de la misma forma.

El estudio del análisis de este tipo de propiedad se encuentra muy poco desarrollado con respecto del análisis de propiedades de confidencialidad o autenticación existiendo muy pocos trabajos al respecto.

6. CONCLUSIONES

Los protocolos de seguridad deben garantizar aquellas propiedades de seguridad para los que estén diseñados. Las propiedades básicas que hemos analizado en este trabajo son las de confidencialidad, autenticación, no-repudio y anonimato.

La mayoría de los trabajos de basan en el estudio del secreto y la autenticación ya que son las más usuales en sistemas tradicionales como el control de acceso o el intercambio de una clave de sesión.

El estudio de las demás propiedades es un problema abierto sobre el que están trabajando actualmente varios grupos de investigación. El problema principal con el que nos encontramos al estudiar estas propiedades es el de formalizar características tan complejas como es el no-repudio.

REFERENCIAS

1. Alur, R.; Henzinger, T.; Mang, F.; Qadeer, S.; Rajamani, S. and Tasiran, S. *"Mocha: modularity in model checking"*. In A. Hu and M. Vardi, editors, CAV 98: Computer-aided Verification, Lecture Notes in Computer Science 1427, pages 521-525. Springer-Verlag, 1998.
2. Bellovin, S.M. and Merrit, M. *"Encrypted key exchange: Password-based protocols secure against dictionary attacks"*. In Proceedings of IEEE Symposium on Reseach in Security and Privacy, pages 72-84, 1992.
3. Burrows, M.; Abadi, M. and Needham, R. *"A logic of authentication"*. In Proceedings of the Royal Society, Series A, 426(1871):233-271, 1989.
4. Denker, G. and Millen, J. *"Capsl intermediate language"*. In Formal Methods and Security Protocols,1999. FLOC '99 Workshop.
5. Dolev, D. and Yao, A. *"On the security of public key protocols"*. IEEE Transactions on Information Theory, IT-29:198{208, 1983. Also STAN-CS-81-854, May 1981, Stanford U.
6. Gong, L.; Needham, R. and Yahalom, R. *"Reasoing about belief in cryptographic protocols"*. IEEE Symposium on Research in Security and Privacy, Oakland, California, 1990.
7. Holzmann, G. *"Design and Validation of Computer Protocols"*. Prentice-Hall, Englewood Cliffs, 1991.
8. Leduc, G. and Germeau, F. *"Verification of Security Protocols using LOTOS-method and apliaction"*. Computer Communications 23. 2000.
9. López, J.; Ortega, J.J; Troya, J.M. *"Verification of authentication protocols using SDL-Method."* Workshop of Information Security. Abril 2002 (en prensa).
10. Lowe, G. *"Towards a Completeness Result for Model Checking of Security Protocols"*. In 11th IEEE Computer Security Foundations Workshop, pages 96-105. IEEE Computer Society, 1998.
11. Marrero, W.; Clarke, E. and Jha, S. *"Model checking for security protocols"*. DIMACS Workshop on Design and Formal Verification of Security Protocols, 1997.
12. Meadows, C. *"A model of computation for NRL protocol analyser"*. Computer Security Foundation Workshop. 1994.
13. Meadows, C. *"Formal verification of cryptographic protocols: A survey"*. In Advances in Cryptology-ASIACRYPT '94, number 917 in Lecture Notes in Computer Science, pages 133-150. Springer-Verlag, Berlin, 1995.
14. Meadows, C. *"Open issues in formal methods for cryptographic protocol analysis"*. In Proceedings of DISCEX 2000,pages 237--250. IEEE Comp. Society Press, 2000.
15. Menezes, A.;. van Oorschot, P.C.; Vanstone, S. *"Handbook of Applied Cryptography"*. CRC Press, (1997).
16. Mitchell, J.C.; Mitchell, M. and Stern, U. *"Automated analysis of cryptographic protocols using Murphi"*. In Proceedings of IEEE Symposium on Security and Privacy, pages 141-151. IEEE Computer Society Press, 1997.
17. Ortega, J.J. *"Técnicas de Descripción Formal para el estudio de la Vulnerabilidad de Protocolos"*. V Reunión Española de Criptología y Seguridad, pages 201-213, Málaga 1998.
18. Rusinowich, M. *CASRUL*. http://www.loria.fr/equipes/protheo/SOFTWARES/CASRUL/
19. Rusinowich, M.; Jacquemard, F.; Vigneron, L. *"Compiling and Verifying Security Prot*ocols *Logic for Programming and Automated Reasonin"*. Reunion Island, November 2000. LNCS 1955.
20. Rusinowich, M. and Turuani, M. *"Protocol Insecurity with Finite Number of Sessions is NP-complete"*. 14th IEEE Computer Security Foundations Workshop June 11-13, 2001 Cape Breton, Nova Scotia, Canada.
21. Ryan, P. and Scheneider, S. *"Modelling and Analysis of Security Protocols:the CSP Approach"*. Addison-Wesley, 2001, ISBN 0 201 67471 8.
22. Schneider, S. *"Formal analysis of a non-repudiation protocol"*. In Proceedings of the 11th IEEE Computer Security Foundations Workshop (CSFW '98), pages 54-65, June 1998.
23. Turner, J.K. *"Using Formal Description Techniques. An introduction to Estelle, LOTOS and SDL"*.
24. Woo T.Y.C. and Lam, S.S. *"A Semantic model for authentication protocols"*. IEEE Symposium on Research in Security and Privacity, 1993.
25. Zhou, J. and Gollmann, D. *"Towards verification of non-repudiation protocols"*. In Proceedings of 1998 International Refinement Workshop and Formal Methods Pacific, pages 370-380, Canberra, Australia, Sept. 1998. Springer.