Eine interaktive und kollaborative Selbstlernumgebung für den Bereich der Sicherheitsprotokolle[1]

C. Burger, M. Papesch

Institut für Parallele und Verteilte Systeme, Universität Stuttgart
Universitätsstraße 38
70569 Stuttgart
caburger@informatik.uni-stuttgart.de, mpapesch@gmx.de

Abstract: Im Bereich Verteilter Systeme und Rechnernetze spielen Sicherheitsprotokolle zum Schutz von Identität und anderweitigen Ressourcen einzelner Personen eine wichtige Rolle. Um die Verständlichkeit dieser komplexen Materie in der Lehre zu erhöhen und für dieses Gebiet ein von Ort und Zeit unabhängiges Lernen zu ermöglichen, entstand eine speziell darauf ausgelegte Selbstlernumgebung. Sie verbindet vielfältige Funktionen für ein exploratives Vorgehen sowie automatisiert auswertbare Selbsttests zur Lernkontrolle mit der Möglichkeit der gemeinsamen Erarbeitung von Inhalten im Lernteam.

1. Einleitung

Eigenständiges, selbstbestimmtes Lernen spielt eine zunehmend wichtige Rolle. Aufgrund des universitären Selbstverständnisses gilt dies bereits während des Studiums, aber in gleicher Weise in der Weiterbildung mit ihrer stärkeren zeitlichen Beanspruchung durch anderweitige Tätigkeiten. Eine Unterstützung durch Material, das in elektronischer Form veröffentlicht wird und damit unabhängig von Ort und Zeit zur Verfügung steht, ist daher sehr hilfreich. Was die Verwendung dieses Materials angeht, so wird es meist vor der Bearbeitung in gedruckte Form gebracht, soweit dies möglich ist. Das Lernen am Bildschirm selbst stößt dagegen bislang auf recht geringe Akzeptanz. Dies liegt unter anderem daran, dass das im Vergleich zur Papierversion ganz andere Potential von elektronischem Material noch nicht vollständig ausgelotet ist.

Am Beispiel des Lehrgebiets der Sicherheitsprotokolle behandelt der vorliegende Beitrag die folgenden drei Möglichkeiten, die durch das digitale Medium neu entstehen, und verknüpft sie zu einer interaktiven und kollaborativen Selbstlernumgebung:

[1]Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 08NM081A gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

1. Umfangreiche Explorationsarten und anschauliche Präsentation auf der Basis von Simulation und Visualisierung.

2. Selbsttest zur Lernkontrolle mit automatisierter Auswertung durch Einsatz formaler Validation.

3. Gemeinsame Erarbeitung von Inhalten im Lernteam, um sowohl der drohenden Isolation zu begegnen als auch den Lernerfolg selbst durch Rückmeldungen aus dem Team zu verbessern. Dies geschieht unter Verwendung der Konzepte zur gemeinsamen Nutzung von Anwendungen.

Vor allem die ersten beiden dieser Aspekte sind stark inhaltsbezogen und nur dann umsetzbar, wenn sich der Lerngegenstand formal spezifizieren und auf dieser Basis analysieren lässt. Im Bereich der Kommunikationsprotokolle für Verteilte Systeme und Rechnernetze sind dafür Verfahren verfügbar, aber auch hier treten aufgrund von vollkommen unterschiedlichen Einsatzzwecken und Korrektheitsanforderungen noch Besonderheiten auf. Daher begann die Untersuchung der technischen Realisierbarkeit zunächst mit dem Teilbereich der Sicherheitsprotokolle.

Diese speziellen Protokolle dienen dazu, Menschen beim Handeln in der elektronischen Welt zu schützen, also ihre Privatsphäre, Identität, Autonomie und Ressourcen vor unerlaubtem Zugriff oder Störung durch Dritte zu bewahren. Die Art solcher Angriffe ist schlecht vorauszusehen, was das Verständnis sehr erschwert. Eine Unterstützung durch die oben genannten drei Möglichkeiten ist also wünschenswert.

Nach einem Überblick über relevante Lernszenarien für Sicherheitsprotokolle in Abschnitt 2 skizzieren die Abschnitte 3 und 4 die Architektur einer geeigneten Selbstlernumgebung sowie die aktuelle Realisierung. Abschnitt 5 gibt erste Erfahrungen mit dem Prototypen wieder, Abschnitt 6 verweist auf verwandte Arbeiten.

2. Lernszenarien

Bei Sicherheitsprotokollen sind üblicherweise vier verschiedene Instanzen involviert. In der Literatur finden sich dafür häufig die Bezeichnungen Alice, Bob, Eve und Authentifizierungsserver, kurz AS (vgl. [Ta96]). Alice und Bob wollen in der Regel mit Hilfe von AS eine sichere Kommunikationsverbindung untereinander aufbauen, während Eve genau dies verhindern oder in sonstiger Weise störend einwirken möchte. Ein Sicherheitsprotokoll erfüllt seinen Zweck, wenn es derartige Angriffe erfolgreich abwehren kann. Dies ist immer dann möglich, wenn sich die Angreiferin durch das Senden eines nicht protokollkonformen Nachrichtentyps oder –parameters verrät. Weist das Protokoll dagegen wie das in Tabelle 1 dargestellte Beispiel Schwächen auf, so kann sich Eve für den Dialogpartner Bob ausgeben, ohne dass Alice dies erkennen kann.

Richtung	Nachricht
Alice → AS	REQUEST_KEY (Alice, Bob, Zufallszahl_1)
AS → Alice	SESSION_KEY ({Zufallszahl_1, Alice, Sitzungsschlüssel, { Sitzungsschlüssel, Alice}$^{\text{Geheimnis B – AS}}$}$^{\text{Geheimnis A – AS}}$)
Alice → Bob	REQUEST_SESSION ({ Sitzungsschlüssel, Alice}$^{\text{Geheimnis B – AS}}$)
Bob → Alice	CHALLENGE ({ Zufallszahl_2, Bob}$^{\text{Sitzungsschlüssel}}$)
Alice → Bob	CONFIRM (Zufallszahl_2 – 1)

Tabelle 1: Beispiel eines fehlerhaften Sicherheitsprotokolls

Wie Lernen allgemein erfolgt auch das Erlernen von Sicherheitsprotokollen schrittweise. Am Anfang steht häufig die Einführung in ein bestimmtes Protokoll anhand vordefinierter Fälle, beispielsweise der erfolgreichen Abwehr oder des Versagens gegenüber einem Angriff. Dies lässt sich gleichermaßen während einer Präsenzveranstaltung oder im Rahmen des Selbststudiums durchführen.

In einem zweiten Schritt ist den Lernenden die Gelegenheit zu geben, das Protokoll selbstständig zu erkunden und auf Schwächen hin zu untersuchen, indem sie verschiedene Angriffe definieren und durchspielen, sowohl im Alleingang als auch im Team. Bei mehreren Beteiligten sieht eine mögliche Aufgabenteilung derart aus, dass ein bis zwei Personen die Rollen von Alice und Bob übernehmen und eine weitere die der Angreiferin Eve.

Wurden auf diese Weise Fälle gefunden, in denen das Protokoll versagt, besteht der dritte Schritt darin, eine Lösung für das Schließen der Sicherheitslücken zu finden und das vorgegebene Protokoll entsprechend zu korrigieren. Prinzipiell lässt sich die verbesserte Protokollvariante erneut mit den vorher gefundenen Angriffen untersuchen. Dies ist jedoch zeitaufwändig und bietet keinerlei Garantie hinsichtlich der Korrektheit der von den Lernenden erarbeiteten Lösung. Zu beachten ist außerdem, dass alle Lernschritte auch ohne tutorielle Betreuung durchführbar sein müssen, um die oben genannte Unabhängigkeit von Zeit und Ort zu erreichen. Eine automatisierte Entscheidung über die Korrektheit der Lösung ist daher vorzuziehen.

3. Gesamtarchitektur und beteiligte Komponenten

Die im letzten Abschnitt beschriebenen Lernszenarien erfordern eine Architektur, die es einzelnen Personen und einem Team aus mehreren Lernern ermöglicht, gemeinsam und möglicherweise in verschiedenen Rollen ein gegebenes Protokoll sowohl anhand vordefinierter als auch selbst entworfener Fälle zu untersuchen, eine Korrektur zu erarbeiten und die Korrektheit dieser Lösung zu erfragen. Dementsprechend besteht die Selbstlernumgebung zunächst aus einem Server zur Simulation und Validation von Sicherheitsprotokollen sowie aus Clients zu deren Visualisierung (linke Seite in Abbildung 1). Prinzipiell ist damit schon Teamarbeit möglich, für weitergehende Funktionalitäten wie z. B. eine rollenabhängige Eingabesteuerung ist jedoch die Einbindung in ein allgemeines Framework zur gemeinsamen Anwendungsnutzung sinnvoll (rechte Seite in Abbildung 1). Die folgenden Teilabschnitte behandeln die genannten Komponenten etwas detaillierter. Für die ausführliche Beschreibung sei auf [Pa03] verwiesen.

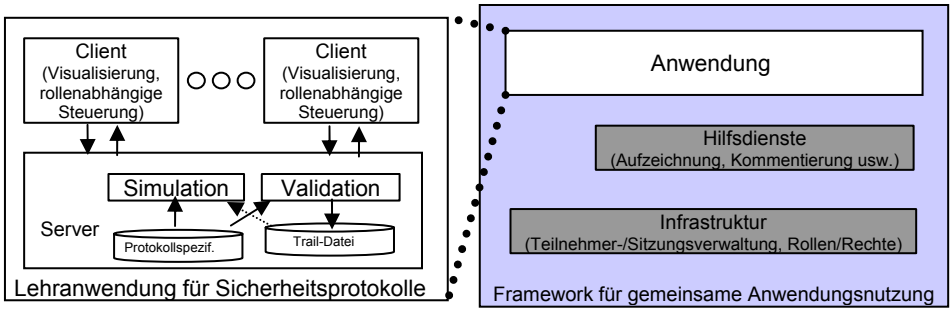

Abbildung 1: Gemeinsame Nutzung der Lehranwendung für Sicherheitsprotokolle

3.1 Simulation und Validation, Visualisierung

Untersuchungsgegenstand bei Protokollen können prinzipiell sowohl Leistungs- als auch funktionale Eigenschaften sein. Bei Sicherheitsprotokollen stehen allerdings letztere im Vordergrund. Es bietet sich daher an, die Verfahren des Model Checking heranzuziehen, die Untersuchungen mit Hilfe von Simulationen und Validationen ermöglichen (vgl. z. B. [Ho97]). Sie arbeiten auf der Basis von formalen Spezifikationen, bei denen sowohl das Protokoll als auch die von ihm zu erfüllenden Korrektheitskriterien mit Hilfe formaler Methoden wie beispielsweise der algebraischen beschrieben werden.

Eine formale Spezifikation enthält die beteiligten Instanzen (Alice, Bob, Eve, AS), die Verbindungen zwischen ihnen, die möglichen Nachrichtentypen sowie den Protokollablauf. Letzterer regelt eindeutig die für einen Austausch zugelassene Abfolge von Nachrichten. Während für Alice und Bob jeweils nur das protokollkonforme Verhalten in Frage kommt, sind für Eve alle möglichen Angriffsvarianten zu definieren. Hierbei kann auf die Ergebnisse von [MS02] zurückgegriffen werden, die ein Verfahren zur Ableitung eines universellen Angreifers für ein gegebenes Protokoll entwickelt haben.

Es basiert auf einer statischen Analyse des Nachrichtenaustauschs zwischen den beteiligten Instanzen. Darüber hinaus enthält die abgeleitete Angreiferspezifikation einen Mechanismus zur Erkennung von Erfolg oder Misserfolg, je nachdem, ob sie Zugriff auf entsprechende Nachrichteninhalte hat oder nicht, und vermerkt dies entsprechend in einer globalen Variablen.

Sowohl die Spezifikation des Protokolls an sich als auch die der zugehörigen Korrektheitskriterien sind durch den Dozenten im voraus zu erstellen und den Lernenden zur Verfügung zu stellen. Diese laden die entsprechende Datei in die Selbstlernumgebung und starten die Simulation. Für den ersten der drei Lernschritte können Lernende die Simulation im Zufallsmodus ablaufen lassen oder vordefinierte Fälle durchspielen. Letzeres erfolgt über sogenannte Trail-Dateien, die sich mit Hilfe der Validation automatisch erzeugen lassen und in die Simulation eingespeist werden können. Für den in Abbildung 2 gezeigten zweiten Lernschritt können sich die Lernenden aus der Menge aller möglichen Ablaufpfade durch das Protokoll sukzessiv jeweils einen zusammenbauen. Dabei besteht auch die Möglichkeit, auf diesem Pfad zurück zu wandern und eventuell einen anderen Weg einzuschlagen.

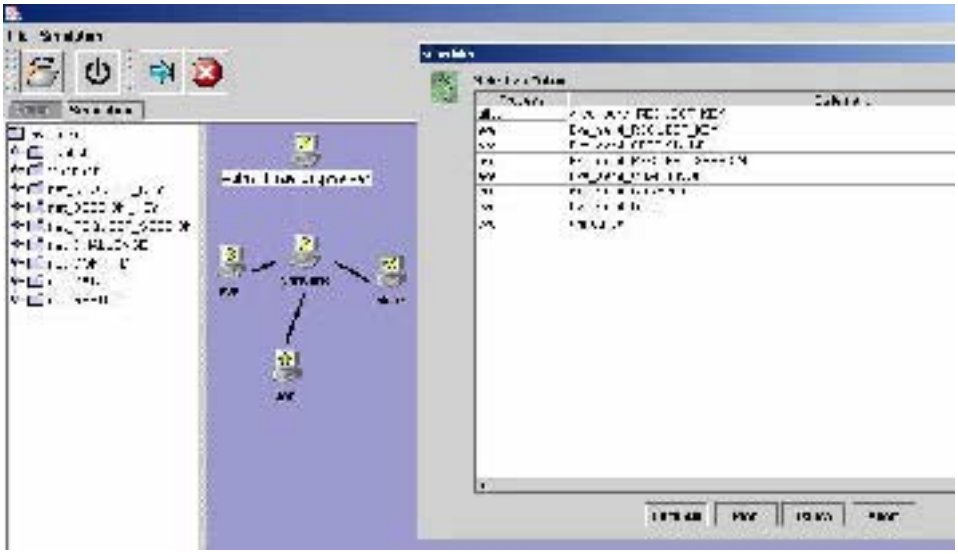

Abbildung 2: Visualisierung und Steuerung der Protokollsimulation (Lernschritt 2)

Im dritten Lernschritt schließlich können die Lernenden das vorgegebene Protokoll einerseits dadurch modifizieren, dass sie während der Simulation anstelle der Protokollinstanzen agieren und interaktiv sowohl einzelne Nachrichten als auch ganze Folgen von Nachrichten selbst definieren. Das dadurch spezifizierte Verhalten einer oder mehrerer Protokollinstanzen wird aufgezeichnet und kann nach einem Simulationslauf unter Verwendung eines geeigneten Gerüsts als neue Spezifikation abgespeichert werden.

Andererseits können diejenigen, die sich dies zutrauen, auch direkt die gegebene Protokollspezifikation in die von ihnen anvisierte Richtung abändern. In beiden Fällen eröffnet sich dadurch für die Lernenden die Möglichkeit, ihre Lösung durch Einsatz der Validation zu überprüfen, was durch Abgleich mit den zuvor vom Dozenten erstellten Korrektheitskriterien erfolgt.

Die Visualisierung der simulierten Daten dient der Veranschaulichung und unterstützt die Erkundung des Protokollverhaltens. Aus diesem Grund enthält sie verschiedene Sichten, eine zustandsorientierte Sicht, die punktuell den aktuellen Zustand von Instanzen, Verbindungen und Nachrichten einschließlich aller Variablen- und Parameterwerte wiedergibt sowie eine topologische Sicht, ein Zeit-Ablauf-Diagramm und die Inhalte der ausgetauschten Nachrichten im Verlauf der Zeit.

3.2 Framework für gemeinsame Anwendungsnutzung

Ein Framework zur gemeinsamen Nutzung verschiedener Anwendungen wie z. B. der im letzten Abschnitt beschriebenen bietet eine Reihe von Diensten, die in diesem Zusammenhang allgemein benötigt werden. Dies beginnt bereits mit der Auffindbarkeit. Nach der Authentifikation haben die Beteiligten also die Möglichkeit, unter den vom Framework verwalteten Anwendungen auszuwählen und diese dann jeweils gemeinsam zu steuern, wobei der Zugriff rollenabhängig eingeschränkt sein kann. Dazu sind mit Hilfe des Frameworks über Regeln und Strategien geeignete Rollen zu definieren und den einzelnen Teamteilnehmern zuzuweisen. Bei jeder Eingabe durch einen Benutzer erfolgt dann durch das Framework eine Überprüfung, ob die entsprechende Aktion mit seiner Rolle verträglich ist, und im negativen Fall eine Zurückweisung. Dieses Konzept dient in der Selbstlernumgebung unter anderem dazu, die Teilnehmer im zweiten Lernschritt auf die Steuerung einzelner Instanzen einschränken und sie dadurch entweder eine protokollkonforme Rolle oder die der Angreiferin spielen lassen zu können.

Außer der Benutzer- und Rechteverwaltung können verschiedene Hilfsdienste wie beispielsweise einer zur Aufzeichnung von Sitzungen des Lernteams oder zur Verankerung von Fragen und Kommentaren an bestimmten Zuständen eingebunden sein. Jede Anwendung im Framework kann diese zusätzlichen Dienste nutzen. Auf dieser Basis lässt sich unter anderem auch die Möglichkeit des Rückwärts- und Vorwärtsnavigierens innerhalb eines Protokollablaufs realisieren.

4. Einige Implementierungsaspekte

Die Realisierung der Selbstlernumgebung erfolgte in JAVA. Als Spezifikationsbasis wurde aufgrund seiner großen Verbreitung und vielfältigen Möglichkeiten die Sprache PROMELA (Protocol Meta Language) mit der Simulations- und Validationsumgebung SPIN (Simple Promela Interpreter, vgl. [Ho97]) gewählt, die in C verfasst ist. Da PROMELA und SPIN in erster Linie auf die Designphase von Protokollen und nicht auf Lehranwendungen abzielen, waren Erweiterungen an der Spezifikationssprache selbst wie beispielsweise Makros für die Ver- und Entschlüsselung von Daten erforderlich. Zur

Kommunikation mit einem oder mehreren Visualisierungs-Clients und insbesondere der Möglichkeit, Parameterwerte während einer laufenden Simulation ändern zu können, mussten der PROMELA-Interpreter und der SPIN-Scheduler in JAVA nachimplementiert werden. Bei Ausführung eines PROMELA-Statements schreibt der Scheduler dementsprechend eine eindeutige Kennung in den zur Kommunikation verwendeten JavaSpace und stößt damit die entsprechende Visualisierung im Client an, wozu JAVA Swing und die in [BR01] beschriebenen Erweiterungen eingesetzt werden. Die Benutzer- und Rechteverwaltung sowie die Einbindung zusätzlicher Hilfsdienste erfolgt durch das in [Bo03] vorgestellte Framework.

5. Erste Erfahrungen

Nach einem ersten Usability-Test mit vier Studierenden und ermutigendem Ergebnis erfolgte ein großflächiger Einsatz des Prototypen mit dem in Tabelle 1 skizzierten Protokoll in den Übungen zur Vorlesung „Grundlagen Verteilter Systeme“ (ca. 100 Teilnehmer) und dem englischsprachigen Pendant „Introduction to Distributed Systems“ (ca. 50 Teilnehmer) im Sommersemester 2003. Bei der anschließenden Bewertung wurde die Frage gestellt, ob die multi- und telemedialen Anteile von Vorteil gewesen seien. Bei einer Rücklaufquote von 27 in der deutschsprachigen (d) und 26 in der englischsprachigen (e) Veranstaltung entschieden sich 28 (17d+11e) Personen für die Antwort „ja“, 20 (7d+13e) für „teils/teils“ und 4 (2d+2e) für „nein“, sprachen sich also mehrheitlich für den Einsatz des Systems aus.

6. Verwandte Arbeiten

Trotz seines Potentials für die Lehre von Kommunikationsprotokollen finden sich erst wenige Ansätze, in denen eine Kombination aus Simulation und Validation in diesem Bereich eingesetzt wird ([ELL01], [Re98]). Gegenstand dieser Arbeiten sind Protokolle zum gegenseitigen Ausschluss und den dinierenden Philosophen. Sie liegen damit fern von dem hier behandelten Thema der Sicherheitsprotokolle. Außerdem wurde die Möglichkeit einer Zusammenarbeit zwischen Lernern nicht betrachtet.

Speziell für das System SPIN sind bereits grafische Benutzungsoberflächen zur Visualisierung verfügbar. Dies gilt in erster Linie für XSPIN ([Ho97]), das zur Laufzeit unterschiedliche Anzeigeformen anbietet, wie ein Zeit-Ablauf- und ein Nachrichten-Diagramm sowie den Zugriff auf die einzelnen Parameterwerte. Die hier vorgestellte Selbstlernumgebung enthält zwei weitere Funktionalitäten, die speziell für Lehrzwecke benötigt werden: Eine Topologie-Ansicht sowie die Möglichkeit, sich auf den simulierten Pfaden rückwärts und vorwärts zu bewegen.

Wie bereits in Abschnitt 3.1 beschrieben, setzt die Selbstlernumgebung auf den Ergebnissen von [MS02] auf, um für ein gegebenes Protokoll die Spezifikation eines universellen Angreifers abzuleiten, erweitert sie aber, um den speziellen Anforderungen des Einsatzes in der Lehre gerecht zu werden.

7. Zusammenfassung und Ausblick

Der Beitrag beschreibt eine Kombination von Simulation, Visualisierung, Validation und Gemeinsamer Nutzung von Anwendungen sowie einigen Erweiterungen für eine interaktive und kollaborative Selbstlernumgebung, um die Lehre im Bereich der Sicherheitsprotokolle zu verbessern. Geplant sind eine Ausweitung auf andere Protokolltypen aus dem Bereich Verteilter Systeme und Rechnernetze sowie weitere Einsätze und Bewertungen in Lehrveranstaltungen.

Danksagung

Unser Dank gilt Prof. Dr. Kurt Rothermel für seine Unterstützung sowie den Studierenden Reinhard Kuntz, Daniel Tiebler, Jing Jing Wei und Yang Zhou für die Teilnahme am Usability-Test. Die zuerst genannte Autorin erhält ein Margarethe-von-Wrangell-Habilitationsstipendium des Landes Baden-Württemberg.

Literaturverzeichnis

[Bo03] Bosau, D. et al.: Distributed Computing in Classrooms: Towards More Interactivity during Lectures. Vortrag beim Microsoft Regional Crash Course for Central and Eastern Europe, München, Deutschland, 8. April 2003

[BR01] Burger, C.; Rothermel, K.: A framework to support teaching in distributed systems. Journal of Educational Resources in Computing (JERIC), 1(1es):3, 2001.

[ELL01] Edelkamp, S.; Lafuente, A. L.; Leue, S.: Protocol Verification with Heuristic Search. In AAAI-Spring Symposium on Model-based Validation of Intelligence, 2001.

[Ho97] Holzmann, G. J.: The Spin Model Checker. IEEE Transactions on Software Engineering, 23(5), 1997.

[MS02] Maggi, P.; Sisto, R.: Using SPIN to Verify Security Properties of Cryptographic Protocols. In Proceedings of the 9-th SPIN Workshop 2002, 2002.

[Pa03] Papesch, M.: A constructivist environment for selfstudies in the area of security protocols. Diplomarbeit, Universität Stuttgart, 2003.

[Re98] Regehr, J.: Using SPIN to Help Teach Concurrent Programming, 1998. URL: http://citeseer.nj.nec.com/regehr98using.html.

[Ta96] Tanenbaum, A. S.: Computer Networks. Prentice-Hall, 1996.