Universitatea din București
Facultatea de Matematică și Informatică
Securitate și Logică Aplicată

LUCRARE DE DISERTAȚIE

Coordonator științific:
Prof. Univ. Dr. Ioana LEUȘTEAN

Absolvent:
Bogdan MACOVEI

București
2021

Universitatea din București
Facultatea de Matematică și Informatică
Securitate și Logică Aplicată

LUCRARE DE DISERTAȚIE

Logică dinamică epistemică pentru modelarea protocoalelor de securitate

Coordonator științific:
Prof. Univ. Dr. Ioana LEUȘTEAN

Absolvent:
Bogdan MACOVEI

București
2021

Rezumat

Protocoalele de securitate sunt o componentă principală în analiza și implementarea securității sistemelor, ceea ce conduce la dorința de a dispune de mecanisme formale care să poată asigura o verificare cât mai riguroasă a respectării tuturor cerințelor impuse pentru a avea garanția schimbului sigur de mesaje între agenții participanți în cadrul sesiunilor de comunicare. În cadrul acestei lucrări, am dezvoltat un nou sistem logic - *DELP* - corect și complet, care să poată reprezenta, în particular, toate elementele necesare protocoalelor, să îmbine ideile din abordările recente și, în final, să demonstreze proprietățile de securitate ale unui protocol. Pentru implementare și verificare automată, a fost utilizat *Lean*, un demonstrator interactiv bazat pe teoria tipurilor dependente.

Cuprins

1 Preliminarii: modelarea protocoalelor de securitate **2**
1.1 Protocoale de securitate 2
1.1.1 Protocolul Needham-Schroeder 3
1.2 Logici pentru modelarea protocoalelor 5
1.2.1 Logica BAN 5
1.2.2 Abordarea logicienilor Halpern, van der Meyden și Pucella 8
1.2.3 Abordarea operațională 11
1.2.4 Protocoalele din *Hidden protocols* 12

2 DELP - Logică dinamică epistemică pentru modelarea protocoalelor **14**
2.1 Sisteme clasice 14
2.1.1 Logică modală 14
2.1.2 Logică epistemică 16
2.1.3 Logică dinamică 17
2.1.4 Logică dinamică epistemică 18
2.2 DELP - sistem logic dinamic epistemic 20
2.2.1 Sintaxa 20
2.2.2 Semantica 20
2.2.3 Sistemul deductiv 21
2.2.4 Completitudinea 23
2.3 Modelarea logicii BAN 26
2.3.1 Translatarea formulelor din logica BAN 26
2.3.2 Corectitudinea regulilor BAN 26

3 Implementare în Lean **30**
3.1 Prezentare generală 30
3.2 Implementarea DELP 31
3.2.1 Limbajul 31
3.2.2 Sistemul deductiv 32
3.3 Verificarea regulilor BAN 33

4 Implementarea protocolului Needham-Schroeder **38**
4.1 Protocolul Needham-Schroeder 38

4.1.1 Protocolul Needham-Schroeder și explicația schimbului de mesaje 38
4.1.2 Modelarea protocolului în logica BAN 39
4.1.3 Descrierea protocolului Needham-Schroeder în Lean 40
4.1.4 Verificarea proprietăților de securitate ale protocolului Needham-Schroeder în Lean 42

Introducere

În această lucrare am dezvoltat un nou sistem de analiză formală a protocoalelor de securitate și a proprietăților acestora, și anume sistemul *DELP*. Elementele principale care stau la baza lui sunt trei abordări recente de modelare ([5], [7] și [8]) și logica dinamică epistemică din [9]. Rezultatul final obținut este o logică pentru care avem demonstrată teorema de completitudine, și ale cărei rezultate teoretice le-am implementat și verificat în *Lean*, un demonstrator interactiv bazat pe teoria tipurilor dependente și, mai specific, pe teoria *calculus of constructions*.

Scopul dezvoltării unui nou sistem a fost dat de faptul că abordările studiate până în prezent sau nu au rezultatele teoretice complete ([5]), sau nu se pot încadra într-o logică ([7]).

În cazul articolului [5], logica propusă este foarte expresivă, cu multe elemente pentru modelarea tuturor cazurilor (logică temporală, probabilități, logică epistemică), dar doar o mică parte dintre acestea sunt utilizate pentru formalizarea singurului exemplu prezentat - și anume logica BAN - și, în plus, nu este demonstrat niciun rezultat teoretic.

În ceea ce privește articolul [7], sistemul dezvoltat este un sistem formal, dar nu poate fi încadrat într-o logică, este doar un sistem etichetat de tranziții, care poate să încorporeze schimbul de mesaje (acțiunile) din cadrul sesiunilor de comunicare.

DELP rezolvă aceste probleme, fiind un sistem dezvoltat peste o logică standard, cu rezultatele teoretice bine puse la punct, preluând și îmbinând ambele idei, cu ajutorul [8].

Lucrarea este structurată pe patru capitole, astfel: în primul capitol sunt prezentate abordările recente în analiza formală a protocoalelor, reprezentând punctul de pornire în construcția *DELP*, al doilea capitol prezintă logica dinamică epistemică și întreg sistemul *DELP*, bazat pe logicile standard și completat cu un set de trei axiome specifice, care păstrează toate rezultatele teoretice și care ajută la modelarea logicii BAN și la corectitudinea acestora. Al treilea capitol prezintă implementarea în *Lean* a sistemului și verificarea automată a regulilor de deducție BAN, iar al patrulea capitol prezintă analiza unui protocol de securitate, în acest caz fiind ales protocolul *Needham-Schroeder* cu cheie publică.

Capitolul 1

Preliminarii: modelarea protocoalelor de securitate

În acest capitol vom prezenta ce sunt protocoalele de securitate, și vom exemplifica folosind protocolul *Needham-Schroeder* cu cheie publică. În continuare, vom prezenta patru abordări formale în modelarea protocoalelor, constând într-o abordare istorică (logica *BAN*) și trei abordări recente.

1.1 Protocoale de securitate

Un protocol de securitate se definește ca fiind o mulțime de reguli și de convenții care determină un schimb de mesaje între doi sau mai mulți agenți, cu scopul de a implementa un serviciu de securitate.

Un protocol este neambiguu și descrie mai multe comportamente, numite roluri, iar fiecre agent execută un rol. Este presupus că toți agenții cunosc protocolul în avans și că nu au alt canal de comunicare în afara celui descris.

În această lucrare, scopul este de a dezvolta o logică ce poate specifica astfel de protocoale. O reprezentare vizuală este următoarea:

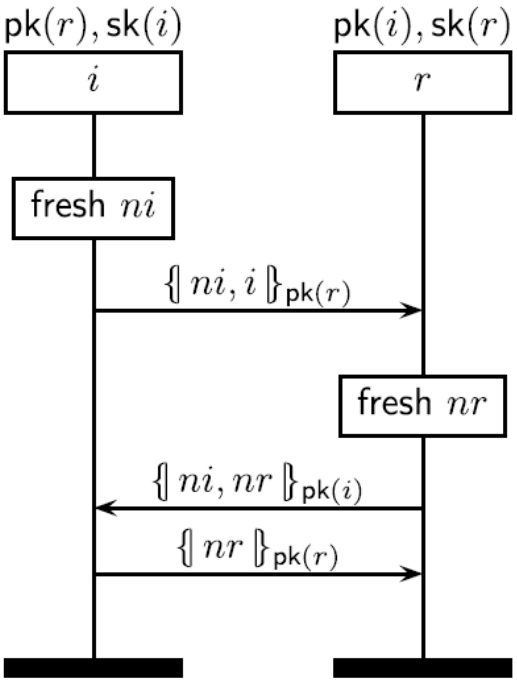

Figura 1.1: Exemplu de protocol de securitate [7]

1.1.1 Protocolul Needham-Schroeder

Protocolul *Needham-Schroeder*, propus de Roger Needham și Michael Schroeder, este un protocol care asigură autentificarea reciprocă a doi agenți participanți. În cadrul acestei secțiuni, vom prezenta, pe scurt, protocolul *Needham-Schroeder* cu cheie publică, schimb de mesaje care s-a dovedit nesigur și asupra căruia a fost descoperit un atac de tip *man-in-the-middle*.

Specificația protocolului este următoarea, pentru agenții participanți A, S și B (Alice, Server și Bob).

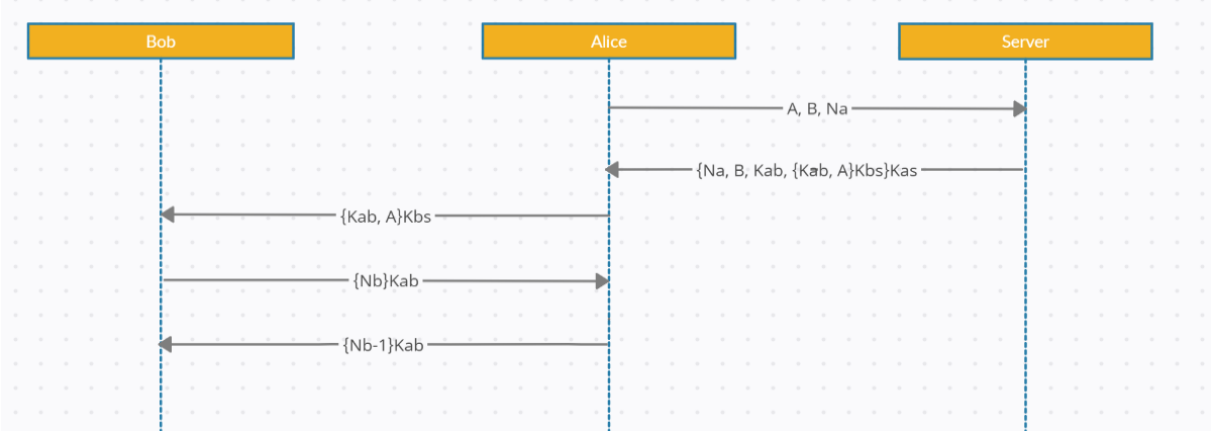

Figura 1.2: Protocolul Needham-Schroeder cu cheie publică

$$
\begin{aligned}
&A \to S : A, B, N_a \\
&S \to A : \{N_a, B, K_{ab}, \{K_{ab}, A\}_{K_{bs}}\}_{K_{as}} \\
&A \to B : \{K_{ab}, A\}_{K_{bs}} \\
&B \to A : \{N_b\}_{K_{ab}} \\
&A \to B : \{N_b - 1\}_{K_{ab}}
\end{aligned}
$$

O descriere a protocolului pe pașii prezentați este următoarea:

- Alice inițiază conexiunea cu Serverul, transmițând cine este, cu cine vrea să comunice și un *nonce*;
- Serverul îi transmite, criptat cu cheia comună dintre A și S, *nonce*-ul generat de A, identitatea lui B și cheia de comunicare dintre A și B, la care se adaugă un mesaj pe care îl poate decripta doar B (fiind criptat cu cheia publică dintre B și S), care conține cheia partajată de A și B și identitatea lui A. În acest mod, A nu poate să citească mesajul transmis de server către B, iar B află cu cine trebuie să comunice și cum poate comunica;
- A îi transmite lui B mesajul pe care nu-l putea decripta, primit de la Server;
- B decriptează mesajul, și îi transmite lui A un *nonce*, criptat cu cheia comună dintre A și B;
- A primește mesajul lui B, îl decriptează, și îl retransmite, aplicându-i o funcție simplă - în acest caz, îl decrementează. Această etapă este utilă în două situații: este o primă protecție pe un *reply attack* și, în plus, arată că agenții sunt încă activi în cadrul sesiunii - verifică un *claim* de *alive*.

1.2 Logici pentru modelarea protocoalelor

În această secțiune vom prezenta patru metode formale pentru analiza protocoalelor de securitate. Punctul de plecare este constituit de logica BAN [4], urmată de abordările [5], [7] și [8].

1.2.1 Logica BAN

Logica BAN este prezentată conform articolului în care este introdusă, și anume [4].

Suportul matematic al logicii BAN pleacă de la elementele principale pe care le avem într-un protocol de securitate: avem agenți participanți, o mulțime de chei, mesaje și formule. În general, agenții participanți sunt notați cu majusculele de la începutul alfabetului - *A, B, C, ...*, cheile cu $K_{a,b}$ (cheia simetrică de comunicare între A și B), K_a (cheia publică a lui A), K_a^{-1} (cheia secretă a lui A). Mesajele sunt notate cu majuscule de la finalul alfabetului, de exemplu X, iar un mesaj criptat are forma $\{X\}_K$.

Construcția formulelor BAN

Formulele din logica BAN sunt construite pe baza următorilor operatori speciali:

- $P \mid\equiv X$: agentul P crede că mesajul X este un mesaj valid în sesiunea curentă;
- $P \triangleleft X$: agentul P vede un mesaj X (nu vede doar X criptat, vede X în sine, ca șir de caractere);
- $P \mid\sim X$: P a spus în trecut X și a crezut X în acel moment în trecut;
- $P \Rightarrow X$: P are jurisdicție asupra lui X și X este un mesaj în care are încredere;
- X: X este un mesaj *fresh*, abia generat. Poate fi văzut ca un *nonce*;
- $P \overset{K}{\leftrightarrow} Q$: K este o cheie simetrică pentru comunicarea dintre P și Q, în care ambii agenți au încredere;
- $\overset{K}{\mapsto} P$: K este cheia publică a lui P;
- $P \overset{X}{\leftrightarrow} Q$: mesajul X este un secret comun între P și Q, în care ambii agenți au încredere;
- $\{X\}_K$: mesajul X criptat cu cheia K;
- $< X >_Y$: reprezintă concatenarea dintre X și Y, unde X este un mesaj, iar Y poate fi mesaj sau formulă.

Regulile de deducție ale logicii BAN

Conform precizării inițiale, logica BAN este formată doar dintr-un sistem deductiv, unde principalele reguli sunt cele care vor fi prezentate în continuare. O observație este aceea că regulile alese de autori sunt, mai de grabă, intuitive, ceea ce face dificilă determinarea unei semantici cu ajutorul căreia sistemul să devină complet și corect.

Regula message-meaning

$$\frac{P \mid\equiv Q \overset{K}{\leftrightarrow} P \quad P \triangleleft \{X\}_K}{P \mid\equiv Q \mid\sim X} \tag{1.1}$$

Modelarea nu suferă nicio schimbare dacă, în loc de cheia K comună, avem că ea este doar cheia publică a lui Q, iar mesajul este văzut cu cheia secretă a lui Q, adică:

$$\frac{P \mid\equiv \overset{K}{\mapsto} Q \quad P \triangleleft \{X\}_{K^{-1}}}{P \mid\equiv Q \mid\sim X} \tag{1.2}$$

sau dacă nu se partajează chei, ci secrete:

$$\frac{P \mid\equiv Q \overset{Y}{\leftrightarrow} P \quad P \triangleleft < X >_Y}{P \mid\equiv Q \mid\sim X} \tag{1.3}$$

Interpretarea este una simplă: dacă P crede că are o cheie sigură sau un secret sigur de comunicare cu agentul Q, iar P vede un mesaj criptat / concatenat cu acel secret, atunci P va crede că Q a transmis acel mesaj.

Regula nonce-verification

$$\frac{P \mid\equiv X \;\; P \mid\equiv Q \mid\sim X}{P \mid\equiv Q \mid\equiv X} \tag{1.4}$$

Intuiția este imediată: P crede că X este un mesaj proaspăt generat și mai crede că acest mesaj proaspăt generat este transmis de X - atunci, P va crede că Q crede (încă) mesajul X.

Regula jurisdicției

$$\frac{P \mid\equiv Q \Rightarrow X \quad P \mid\equiv Q \mid\equiv X}{P \mid\equiv X} \tag{1.5}$$

P crede că X este un mesaj controlat de Q (în care Q are completă încredere și asupra căruia Q are control absolut), iar P crede că Q crede că X este un mesaj actual și valid în sesiunea curentă de comunicare. Atunci, P crede X.

Regula încrederii pe componente

$$\frac{P \mid\equiv X \quad P \mid\equiv Y}{P \mid\equiv (X, Y)} \tag{1.6}$$

care mai poate avea următoarele două forme:

$$\frac{P \mid\equiv (X, Y)}{P \mid\equiv X} \tag{1.7}$$

$$\frac{P \mid\equiv Q \mid\equiv (X, Y)}{P \mid\equiv Q \mid\equiv X} \tag{1.8}$$

Dacă P crede un mesaj, atunci crede și părțile lui componente - iar dacă P crede mai multe părți componente, poate crede mesajul format din acestea. Se aplică și pentru interpunerea unui agent - dacă P crede că Q crede un mesaj, atunci P crede și că Q crede o anumită parte din acel mesaj.

Regula mesajelor nou generate

$$\frac{PX}{P \mid\equiv X, Y} \tag{1.9}$$

Dacă P crede că X este un mesaj proaspăt generat, atunci crede că orice mesaj care îl conține pe X este și el proaspăt generat.

Regula mesajelor recepționate

Există mai multe formulări dintre care următoarele sunt comun utilizate:

$$\frac{P \triangleleft (X, Y)}{P \triangleleft X} \tag{1.10}$$

$$\frac{P \triangleleft < X >_Y}{P \triangleleft X} \tag{1.11}$$

$$\frac{P \mid\equiv Q \overset{K}{\leftrightarrow} P \quad P \triangleleft \{X\}_K}{P \triangleleft X} \tag{1.12}$$

$$\frac{P \mid\equiv \overset{K}{\mapsto} P \quad P \triangleleft \{X\}_K}{P \triangleleft X} \tag{1.13}$$

$$\frac{P \mid\equiv \overset{K}{\mapsto} Q \quad P \triangleleft \{X\}_{K^{-1}}}{P \triangleleft X} \tag{1.14}$$

Toate aceste formule au o intuiție comună: dacă P vede un mesaj compus, vede și părți componente din mesaj, iar dacă P vede un mesaj criptat după o cheie pe care o cunoaște, atunci P vede și mesajul în sine. Cu toate că nu este cazul în logica BAN, diferite abordări moderne diferențiază între modul în care este recepționat un mesaj în cadrul unei sesiuni - dacă este un șir de caractere sau dacă este un mesaj cu sens. Un mesaj cu sens este un șir de caractere, dar nu orice șir de caractere recepționat este un mesaj cu sens, unde mesaj cu sens este un mesaj ce poate fi decriptat de agentul care l-a recepționat. În logica BAN, această distincție este clară prin notație - orice X mesaj este înțeles ca un mesaj decriptat, pe care agentul îl poate replica / retransmite, în timp ce $\{X\}_K$ este un mesaj criptat, pe care nu-l poate înțelege dacă nu se poate deduce că agetul crede cheia K.

Regulile de comutativitate în schimbul de chei

$$\frac{P \mid\equiv Q1 \overset{K}{\leftrightarrow} Q2}{P \mid\equiv Q2 \overset{K}{\leftrightarrow} Q1} \tag{1.15}$$

Această formulă se poate generaliza și pentru un agent P care crede că un alt agent crede în schimbul de chei, astfel că

$$\frac{P \mid\equiv Q \mid\equiv R1 \overset{K}{\leftrightarrow} R2}{P \mid\equiv Q \mid\equiv R2 \overset{K}{\leftrightarrow} R1} \tag{1.16}$$

O observație adusă de cei trei logicieni în articolul în care logica BAN este introdusă ține de cuantificarea universală a tuturor mesajelor (cu rol de variabile) utilizate în cadrul regulilor de deducție. Astfel, prin scrierea $P \mid\equiv X$, înțelegem, de fapt că, dacă mesajul X este format din variabilele V_1, V_2, ... V_n, atunci $P \mid\equiv \forall V_1 \forall V_2 ... \forall V_n X$.

1.2.2 Abordarea logicienilor Halpern, van der Meyden și Pucella

În acest subcapitol, vom prezenta abordarea logicienilor Joseph Y. Halpern, Ron van der Meyden și Riccardo Pucella în articolul *An Epistemic Foundation for Authentication Logics* [5]. Scopul acestui articol îl constituie crearea unei semantici expresive pentru modelarea, în general, a protocoalelor de securitate și, în particular, a logicii BAN. Pentru îndeplinirea acestui scop, sunt utilizate logicile epistemică și temporală, cărora li se adaugă și elemente probabilistice.

Sintaxa

Sunt definite mulțimile K (mulțimea cheilor), N (mulțimea *nonce*-urilor), T (mulțimea textelor clare) și Φ (mulțlimea formulelor atomice), mulțimi disjuncte. Este făcută precizarea că mulțimea cheilor conține atât chei simetrice, cât și chei asimetrice.

Specificația limbajului în forma BNF este următoarea:

$$\begin{aligned}
\mathbf{s} &::= s \mid x \\
\mathbf{m} &::= t \mid k \mid n \mid i \mid (m_1, m_2) \mid \{m\}_k \mid \varphi \\
\varphi &::= p \mid sent_i(s) \mid recv_i(s) \mid extract_i(m) \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid K_i\varphi \mid \bigcirc\varphi \\
&\quad \textcircled{-}\varphi \mid \Box\varphi \mid \boxed{-}\varphi \mid \exists x\varphi \mid [m] = s \mid s \sqsubseteq s' \mid Pr_i(\varphi) \geq \alpha
\end{aligned}$$

unde p reprezintă un element din mulțimea formulelor atomice, Φ, i un agent generic, m un mesaj generic, t un text clar, k o cheie de criptare, n un *nonce*, $\alpha \in [0, 1]$ o măsură de probabilitate, s un șir de caractere, x o variabilă peste mulțimea șirurilor de caractere și φ o formulă.

Este definită și notația

$$\textcircled{s}\varphi \sim \textcircled{-}\varphi \wedge \neg \textcircled{-}\bot$$

i.e. formula φ a fost adevarată la pasul anterior și, în plus, a existat un pas anterior.

Semantica

Sistemul utilizat este un sistem multiagent, format din n agenți și un mediu în care aceștia interacționează (*environment*). Fiecare agent este într-o stare locală într-un anumit moment de timp, iar starea globală este $(st_e, st_1, ..., st_n)$, unde st_i, pentru $i \in 1, ..., n$ este starea curentă a agentului i, iar st_e este starea *environment*-ului.

O rundă în protocol este descrisă ca fiind o funcție de la timp la starea globală. Un punct este o pereche (r, m) și este format dintr-o rundă r și un moment de timp $m \in \mathbb{N}$. Într-un punct (r, m), sistemul este în starea globală $r(m)$, iar dacă $r(m) = (st_e, st_1, ..., st_n)$, atunci $r_i(m)$ reprezintă starea locală a agentului i.

Sistemul logic este definit ca fiind format dintr-o mulțime R de runde. Sunt definite următoarele două formule logice în fiecare stare: $send(j, s)$ și $recv(s)$: formula $send(j, s)$ reprezintă transmiterea șirului de caractere s de către agentul j în cadrul unei runde, iar $recv(s)$ reprezintă recepționarea mesajului s într-o anumită rundă a protocolului.

Asumpții criptografice

Asumpțiile criptografice oferite pentru modelarea protocoalelor sunt următoarele:

- cheia k și cheia inversă k^{-1} sunt șiruri de caractere care se păstrează de-a lungul rundelor protocolului: $[k]_r = k$ si $[k^{-1}]_r = k^{-1}$ pentru orice k cheie și orice r rundă;
- există o funcție de concatenare: dacă s și s' sunt două șiruri de caractere, există atunci șirul (s, s'). Proprietatea se respectă și pentru mesaje;
- există o funcție de criptare dependentă de rundă: pentru un șir de caractere s și o cheie k, la o rundă r avem $[\{s\}_k]_r$ (criptarea lui s cu cheia k în runda r);
- se definește $[\{m\}_k]_r = [\{s\}_k]_r$ dacă $[m]_r = s$, adică agentul înțelege mesajul din șirul de caractere s;
- criptarea este unică: dacă $[\{m\}_k]_r = [\{m'\}_{k'}]_r$, atunci $[m]_r = [m']_r$ si $k = k'$.

Modelele și interpretarea formulelor

Modelele sunt $I = (R, \pi, \mathbf{C}, \{\mu_C\}_{C \in \mathbf{C}})$, unde R este un sistem de runde, π este o funcție de evaluare, $\mathbf{C}$ este o partiție a rundelor lui R, iar pentru fiecare $C \in \mathbf{C}$, mărimea μ_C este probabilitatea de distribuție a rundelor în $\mathbf{C}$.

Interpretarea formulelor este definită astfel:

$$
\begin{aligned}
&(I, r, m) \models p \iff \pi(r(m))(p) \textit{ este adevărat} \\
&(I, r, m) \models \neg\varphi \iff (I, r, m) \not\models \varphi \\
&(I, r, m) \models \varphi_1 \wedge \varphi_2 \iff (I, r, m) \models \varphi_1 \textit{ si } (I, r, m) \models \varphi_2 \\
&(I, r, m) \models K_i\varphi \iff \textit{pentru orice } (r', m') \sim_i (r, m), (I, r', m') \models \varphi \\
&(I, r, m) \models \bigcirc\varphi \iff (I, r, m+1) \models \varphi \\
&(I, r, m) \models \textcircled{-}\varphi \iff m = 0 \textit{ sau } (I, r, m-1) \models \varphi \\
&(I, r, m) \models \Box\varphi \iff \textit{pentru orice } m' \geq m, (I, r, m') \models \varphi \\
&(I, r, m) \models \boxed{-}\varphi \iff \textit{pentru orice } m' \leq m, (I, r, m') \models \varphi \\
&(I, r, m) \models Pr_i(\varphi) \geq \alpha \iff \\
&\qquad \mu_{r,m,i}(\{(r', m') \mid (I, r', m') \models \varphi\} \cap K_i(r, m) \cap (C)(r)) \geq \alpha \\
&(I, r, m) \models \exists x\varphi \iff \textit{există } s \textit{ text}, (I, r, m) \models \varphi[s/x]
\end{aligned}
$$

Interpretarea regulilor din logica BAN

În articol sunt propuse și următoarele traduceri ale formulelor din logica BAN:

$$i \mid\equiv m \Longleftrightarrow \neg K_i \neg \top \wedge K_i(\top \rightarrow m^T) \tag{1.17}$$

$$i \triangleleft m \Longleftrightarrow \exists x \exists y([m^M] = x \wedge recv_i(y) \wedge K_i(x \sqsubseteq y)) \tag{1.18}$$

$$i \overset{k}{\leftrightarrow} j \Longleftrightarrow extract_i(k) \wedge extract_j(k) \wedge \wedge_{i' \neq i,j} \neg extract_{i'}(k) \tag{1.19}$$

$$i \Rightarrow m \Longleftrightarrow K_i(\top \rightarrow m^T) \leftrightarrow m^T \tag{1.20}$$

$$i \mid\sim m \Longleftrightarrow \exists x \exists y([m^M] = x \wedge \Diamond\textcircled{\scriptstyle s}(\neg sent_i(y) \wedge \bigcirc sent_i(y) \wedge K_i(x \sqsubseteq y))) \tag{1.21}$$

$$m \Longleftrightarrow \exists x([m^M] = x \wedge \textcircled{\scriptstyle -}^{l} \bigwedge_i \boxed{-}(\neg \exists y(\neg sent_i(y) \wedge \bigcirc sent_i(y) \wedge x \sqsubseteq y))) \tag{1.22}$$

1.2.3 Abordarea operațională

Din semantica operațională, cele mai importante aspecte pe care le vom utiliza în această lucrare sunt cele care sunt legate de sistemul de deducție pe termeni. Avem următoarele:

1. termenii sunt rolurile, mesajele, cheile si *nonce*-urile;
2. variabilele sunt pe sorturile *Var*, *Fresh* și *Role*;
3. simbolurile de funcție sunt conținute într-o mulțime *Func*;
4. specificația protocolului este formată dintr-o mulțime de roluri;
5. execuțiile sunt un sistem etichetat de tranziție între stări (*LTS - labeled transition system*).

Sistemul de deducție pe termeni

Având Γ o mulțime de cunoștințe, atunci:

1. dacă $t \in \Gamma$, atunci $\Gamma \vdash t$;
2. $\Gamma \vdash t_1$ si $\Gamma \vdash t_2$ dacă și numai dacă $\Gamma \vdash (t_1, t_2)$;
3. dacă $\Gamma \vdash t$ si $\Gamma \vdash k$, atunci $\Gamma \vdash \{|t|\}_k$;
4. dacă $\Gamma \vdash \{|t|\}_k$ și $\Gamma \vdash k^{-1}$, atunci $\Gamma \vdash t$;
5. dacă $\Gamma \vdash t_i, 1 \leq i \leq n$, atunci $\Gamma \vdash f(t_1, t_2, ..., t_n)$, $f \in Func$, $ari\ f = n$.

Mulțimea tuturor informațiilor ce pot fi deduse din Γ reprezintă consecințele lui Γ și se definește prin:

$$Cons(\Gamma) := \{t \in RoleTerm \mid \Gamma \vdash t\}$$

1.2.4 Protocoalele din *Hidden protocols*

În această secțiune, vom prezenta principalele rezultate obținute în articolul *Hidden protocols* [8].

Sunt introduse două mulțimi, mulțimea I a agenților și mulțimea P a formulelor atomice. Pentru interpretarea formulelor, se definesc *modele Kripke* de forma $\mathcal{M} = (S, \sim, V)$, unde S este mulțimea stărilor, $\sim$ este relația de accesibilitate, fiind o relație binară peste S, iar V este funcția de evaluare, care asociază fiecărui atom propozițional o submulțime în S.

Este introdusă și o mulțime Σ de acțiuni, peste care se definește limbajul $\mathcal{L}_{obs}$ al observațiilor. Gramatica acestora este specificată în forma BNF:

$$\pi ::= \delta \mid \varepsilon \mid a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^* \tag{1.23}$$

unde δ denotă o mulțime vidă de observații, constanta ε reprezintă șirul de caractere gol, iar a este un element din alfabetul Σ.

Mulțimea de observații se notează cu $\mathcal{L}(\pi)$ și se construiește inductiv, astfel:

$$\mathcal{L}(\delta) = \emptyset \tag{1.24}$$
$$\mathcal{L}(\varepsilon) = \{\varepsilon\} \tag{1.25}$$
$$\mathcal{L}(a) = \{a\} \tag{1.26}$$
$$\mathcal{L}(\pi \cdot \pi') = \{wv | w \in \mathcal{L}(\pi) \text{ și } v \in \mathcal{L}(\pi')\} \tag{1.27}$$
$$\mathcal{L}(\pi + \pi') = \mathcal{L}(\pi) \cup \mathcal{L}(\pi') \tag{1.28}$$
$$\mathcal{L}(\pi^*) = \{\varepsilon\} \cup \bigcup_{n>0} (\mathcal{L}(\pi \cdot ... \cdot \pi)) \tag{1.29}$$

Un model epistemic construit cu astfel de observații se definește ca fiind un model epistemic cu așteptări, și este un tuplu $\mathcal{M} = (S, \sim, V, Exp)$, unde $Exp : S \to \mathcal{L}_{obs}$ este o funcție care asociază fiecărei stări o observație π pentru care $\mathcal{L}(\pi) \neq \emptyset$.

Formulele logicii sunt definie conform următoarei gramatici în forma BNF:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid K_i\varphi \mid [\pi]\varphi \tag{1.30}$$

unde $p \in P$, $i \in I$ și $\pi \in \mathcal{L}_{obs}$.

Rezultate importante

Articolul prezintă o serie de rezultate importante, pe care le vom utiliza ulterior pentru definirea unui nou sistem de analiză a protocoalelor.

Bisimilaritatea. O relație binară între două modele epistemice $\mathcal{M} = (S, \sim, V, Exp)$ și $\mathcal{N} = (S', \sim', V', Exp')$ se numește bisimilaritate dacă pentru orice

$s \in S$, $s' \in S'$, avem că dacă $(s, s') \in R$, atunci:

$$\textbf{Invarianță propozițională } V(s) = V'(s') \tag{1.31}$$
$$\textbf{Invarianță peste observații } \mathcal{L}(Exp(v)) = \mathcal{L}(Exp(v')) \tag{1.32}$$
$$\textbf{Zig } s \sim_i t \in \mathcal{M} \Longrightarrow \text{există } t' \in \mathcal{N} \text{ astfel încât } s' \sim'_i t' \text{ și } t \rho t' \tag{1.33}$$
$$\textbf{Zag } s' \sim'_i t' \in \mathcal{N} \Longrightarrow \text{există } t \in \mathcal{M} \text{ astfel încât } s \sim_i t \text{ și } t \rho t' \tag{1.34}$$

Invarianța la bisimilaritate. Pentru două stări $\mathcal{M}, s$ și $\mathcal{N}, s'$, următoarele două afirmații sunt echivalente:

$$i)\ \mathcal{M}, s \leftrightarrow \mathcal{N}, s' \tag{1.35}$$
$$ii)\ \text{pentru orice } \varphi\text{: } \mathcal{M}, s \models \varphi \Longleftrightarrow \mathcal{N}, s' \models \varphi \tag{1.36}$$

unde $\leftrightarrow$ este notația de bisimilaritate între stări.

Modelele restricționate. Fie w o observație peste Σ și $\mathcal{M} = (S, \sim, V, Exp)$ un model epistemic cu așteptări. Atunci, modelul restricționat se notează $\mathcal{M}|_w = (S', \sim', V', Exp')$, unde

1. $S' = \{s \mid \mathcal{L}(Exp(s) - w) \neq \emptyset\}$;
2. $\sim'_i = \sim_i |_{S' \times I \times S'}$;
3. $V' = V|_{S'}$;
4. $Exp'(s) = Exp(s) - w$.

unde $\pi - w = \{v \mid wv \in \mathcal{L}(\pi)\}$.

Modelele temporale. Fie $\mathcal{M} = (S, \sim_i, V, Exp)$ un model epistemic cu așteptări. Modelul temporal generat este notat cu $ET(\mathcal{M})$ și este definit ca $ET(\mathcal{M}) = (H, \rightarrow_a, \sim'_i, V')$, unde

1. $H = \{(s, w) \mid s \in S, w = \varepsilon \text{ sau } w \in \mathcal{L}(Exp(s))\}$;
2. $(s, w) \rightarrow_a (t, v) \Longleftrightarrow s = t$ și $v = wa, a \in \Sigma$;
3. $(s, w) \sim_i (t, v) \Longleftrightarrow s \sim_i t$ și $w = v$;
4. $p \in V'(s, w) \Longleftrightarrow p \in V(s)$.

Având acest model definit, se demonstrează că $\mathcal{M}, s \models \varphi \Longleftrightarrow ET(\mathcal{M}), (s, \varepsilon) \models_{EPDL} \varphi$, astfel că sistemul este complet, conform completitudinii logicii dinamice epistemice.

Capitolul 2

DELP - Logică dinamică epistemică pentru modelarea protocoalelor

În acest capitol vom introduce sistemul *DELP*, un sistem nou pentru analiza formală a protocoalelor de securitate, construit pe baza logicii dinamice epistemice. Vom prezenta, în prima parte, logicile clasice care stau la baza *DELP*, în a doua parte sintaxa, semantica, sistemul deductiv și rezultatele teoretice (teorema de completitudine) pentru acesta, iar în a treia parte vom demonstra corectitudinea regulilor *BAN* în sistemul dezboltat.

2.1 Sisteme clasice

Pentru a ajunge la varianta finală de logică dinamică epistemică, vom prezenta, pe rând, logica modală, logica epistemică, logica dinamică și logica dinamică epistemică.

2.1.1 Logică modală

În această secțiune vom prezenta, pe scurt, logica modală, așa cum este ea introdusă în cartea *Modal Logic*, scrisă de Patrick Blackburn, Maarten de Rijke și Yde Venema [3]. Logica modală poate fi văzută ca o logică între cea propozițională și logica de ordinul I, întrucât nu există formule cuantificate existențial sau universal în limbaj, dar există în metalimbaj, în momentul în care introducem semantica.

Sintaxa

Având o mulțime Φ de formule, limbajul logicii modale este caracterizat prin următoarea gramatică în forma BNF:

$$\varphi ::= p \mid \bot \mid \neg\varphi \mid \varphi \vee \psi \mid \Diamond\varphi \tag{2.1}$$

unde p este un atom propoziţional, $p \in \Phi$.

Se defineşte operatorul $\Box\varphi$, definit astfel:

$$\Box\varphi := \neg\Diamond\neg\varphi \tag{2.2}$$

iar conectorii logici care nu apar în specificaţia BNF, se definesc în funcţie de negaţie şi de conjuncţie,

$$\varphi \wedge \psi := \neg(\neg\varphi \vee \neg\psi) \tag{2.3}$$
$$\varphi \to \psi := \neg\varphi \vee \psi \tag{2.4}$$
$$\varphi \leftrightarrow \psi := (\varphi \to \psi) \wedge (\psi \to \varphi) \tag{2.5}$$

Semantica

Un cadru este o pereche $\mathcal{F} = (W, R)$ unde W este o mulţime finită de stări / lumi, iar R este o relaţie binară pe W, $R \subseteq W^2$.

Având cadrele definite, atunci modelele sunt perechi de forma $\mathcal{M} = (\mathcal{F}, V)$, sau de forma $\mathcal{M} = (W, R, V)$, unde V este o funcţie de evaluare, care asignează fiecărui element din mulţimea de formule o submulţime a lumilor, i.e. $V : \Phi \to \mathcal{P}(W)$.

Având modelele definite, interpretarea formulelor se defineşte, inductiv, astfel:

$$\mathcal{M}, w \models p \iff w \in V(p), \text{ unde } p \in \Phi \tag{2.6}$$
$$\mathcal{M}, w \models \neg\varphi \iff \mathcal{M}, v \not\models \varphi \tag{2.7}$$
$$\mathcal{M}, w \models \varphi \vee \psi \iff \mathcal{M}, w \models \varphi \text{ sau } \mathcal{M}, w \models \psi \tag{2.8}$$
$$\mathcal{M}, w \models \Diamond\varphi \iff \text{există } v \in W \text{ astfel încât } Rwv \text{ şi } \mathcal{M}, v \models \varphi \tag{2.9}$$

Definiţia pentru formula $\Box\varphi$ este următoarea:

$$\mathcal{M}, w \models \Box\varphi \iff \text{pentru orice } v \in W, \text{ dacă } Rwv, \text{ atunci } \mathcal{M}, v \models \varphi \tag{2.10}$$

Sistemul deductiv

Axiomele logicii modale sunt toate insanţele tautologiilor propoziţionale, cărora li se adaugă următoarele două axiome:

$$\text{(K) } \Box(p \to q) \to (\Box p \to \Box q) \tag{2.11}$$
$$\text{(Dual) } \Diamond p \leftrightarrow \neg\Box\neg p \tag{2.12}$$

Regulile de deducţie sunt:

1. *modus ponens*: din φ și $\varphi \to \psi$ demonstrăm ψ;
2. *substituția uniformă*: din φ se obține θ, unde θ este obținut prin înlocuirea în φ a propozițiilor atomice cu formule arbitrare;
3. *generalizarea*: din φ se demonstrează $\Box\varphi$.

Logica modală este un sistem logic corect și complet.

2.1.2 Logică epistemică

În cadrul acestei secțiuni, vom prezenta sistemul *S5*, din cartea *Dynamic Epistemic Logic*, scrisă de Hans van Ditmarsch, Wiebe van der Hoek și Barteld Kooi [9].

Sintaxa

Logica epistemică este o logică modală, unde operatorul $\Box$ este înlocuit de operatorul epistemic, K. Avem o mulțime de formule Φ și o mulțime A de agenți. Atunci, limbajul este definit prin următoarea gramatică în forma BNF:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid K_a\varphi \tag{2.13}$$

unde $p \in \Phi$ este o formulă atomică, iar $a \in A$ este un agent din mulțimea agenților. Restul formulelor *booleene* (disjuncția, implicația, echivalența) se definesc pe baza negației și conjuncției.

Formula $K_a\varphi$ se citește „agentul a știe că φ", iar dualul este notat $\hat{K}_a\varphi := \neg K_a \neg\varphi$ și se citește „agentul a consideră posibil φ".

Semantica

Cadrele și modelele sunt similare celor din logica modală, iar în cartea de referință [9], modelele sunt numite *modele Kripke*. Un model *Kripke* este prezentat ca un triplet $\mathcal{M} = (W, R, V)$, format din mulțimea de stări W, relația binară R peste W, înțeleasă ca relația de accesibilitate, și funcția de evaluare V, care duce atomii propoziționali într-o submulțime a lui W. În general, relația de accesibilitate epistemică se notează $\sim$. Având modelele definite, interpretarea formulelor va fi următoarea:

$$\mathcal{M}, v \models p \iff v \in V(p) \tag{2.14}$$
$$\mathcal{M}, v \models \varphi \wedge \psi \iff \mathcal{M}, v \models \varphi \text{ și } \mathcal{M}, v \models \psi \tag{2.15}$$
$$\mathcal{M}, v \models \neg\varphi \iff \mathcal{M}, v \not\models \varphi \tag{2.16}$$
$$\mathcal{M}, v \models K_a\varphi \iff \text{pentru orice } w \text{ astfel încât } v \sim_a w, \text{ avem } \mathcal{M}, w \models \varphi \tag{2.17}$$

Sistemul deductiv

Cel mai simplu model epistemic este **modelul K**, ale cărui axiome sunt toate instanțele tautologiilor propoziționale, la care se adaugă:

$$\text{(K) } K_a(\varphi \to \psi) \to (K_a\varphi \to K_a\psi) \tag{2.18}$$

$$\tag{2.19}$$

Ca reguli de deducție, avem *modus ponens* (dacă avem o demonstrație pentru φ și pentru $\varphi \to \psi$, atunci avem o demonstrație pentru ψ) și *necesitatea lui* K_a (dacă avem o demonstrație pentru φ, atunci putem infera $K_a\varphi$).

Putem extinde acest sistem cu următoarele trei axiome, pentru a obține sistemul epistemic *S5*. *S5* este un sistem logic corect și complet.

$$\text{(T) } K_a\varphi \to \varphi \tag{2.20}$$

$$\text{(Introspecția pozitivă) } K_a\varphi \to K_aK_a\varphi \tag{2.21}$$

$$\text{(Introspecția negativă) } \neg K_a\varphi \to K_a\neg K_a\varphi \tag{2.22}$$

2.1.3 Logică dinamică

În această secțiune, vom prezenta logica dinamică propozițională, așa cum este ea introdusă în cartea *Dynamic logic*, scrisă de David Harel, Dexter Hozen și Herzy Tiuryn [6].

Sintaxa

În cazul logicii dinamice, pe lângă mulțimea formulelor, Φ, avem și o mulțime a programelor, pe care autorii o notează Π. Pentru formulele, respectiv programele atomice, se introduc mulțimile Φ_0 și Π_0. Limbajul este specificat prin următoarea gramatică în forma BNF:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \to \psi \mid [\alpha]\varphi \tag{2.23}$$

unde $p \in \Phi$ este o formulă atomică, iar $\alpha \in \Pi$ este un program.

Formula $[\alpha]\varphi$ este construită similar formulei modale pe baza operatorului $\Box$, și se citește, în acest caz, că „după execuția lui α, atunci φ". Ea admite și un dual care, de regulă, se notează $< \alpha > \varphi$.

Semantica

Spre deosebire de modelele din logica modală și cea epistemică, în cazul logicii dinamice avem o echivalență pe stări atât ca apartenență la un program, cât și relația obișnuită de accesibilitate. Modelele sunt, de această dată, de forma $\mathcal{M} = (R, V)$, unde R este mulțimea stărilor, iar V este interpretat astfel: dacă îl

aplicăm peste o formulă $\varphi \in \Phi$, $V(\varphi) \subset R$, iar dacă îl aplicăm peste un program $\alpha \in \Pi$, atunci $V(\alpha) \subset R \times R$. Interpretarea este definită inductiv:

$$\mathcal{M}, w \models \varphi \to \psi \iff \mathcal{M}, w \models \varphi \text{ implică } \mathcal{M}, w \models \psi \tag{2.24}$$

$$\mathcal{M}, w \models \neg\varphi \iff \mathcal{M}, w \not\models \varphi \tag{2.25}$$

$$\mathcal{M}, w \models [\alpha]\varphi \iff \text{pentru orice } v \in R \text{ astfel încât } (v, w) \in V(\alpha), \text{ atunci } \mathcal{M}, v \models \varphi \tag{2.26}$$

Pentru programe avem, în plus, următorii operatori:

$$V(\alpha_1 \cup \alpha_2) = V(\alpha_1) \cup V(\alpha_2) \tag{2.27}$$

$$V(\alpha_1; \alpha_2) = V(\alpha_1) \circ V(\alpha_2) \tag{2.28}$$

$$V(\alpha^*) = \bigcup_{n \geq 0} V(\alpha)^n \tag{2.29}$$

Sistemul deductiv

Sistemul deductiv conține toate instanțele tautologiilor propoziționale, cărora li se adaugă următoarele:

$$[\alpha](\varphi \to \psi) \to ([\alpha]\varphi \to [\alpha]\psi) \tag{2.30}$$

$$[\alpha](\varphi \wedge \psi) \leftrightarrow [\alpha]\varphi \wedge [\alpha]\psi \tag{2.31}$$

$$[\alpha \cup \beta]\varphi \leftrightarrow [\alpha]\varphi \wedge [\alpha]\psi \tag{2.32}$$

$$[\alpha; \beta]\varphi \leftrightarrow [\alpha][\beta]\varphi \tag{2.33}$$

Regulile de deducție sunt *modus ponens* și *generalizarea programelor*:

$$(MP)\frac{\varphi \; \varphi \to \psi}{\psi}; \; (GEN)\frac{\varphi}{[\alpha]\varphi}$$

Logica dinamică propozițională este corectă și completă.

2.1.4 Logică dinamică epistemică

Prezentare generală

Logica dinamică epistemică este un sistem care îmbină ultimele două secțiuni prezentate, astfel că avem ca formule atât formula epistemică (a operatorului K), cât și formula din logica dinamică, $[\alpha]\varphi$, cu $\alpha \in \Pi$ și $\varphi \in \Phi$. Limbajul acestei logici este descris prin următoarea gramatică în forma BNF:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \to \varphi \mid K_i\varphi \mid [\alpha]\varphi \tag{2.34}$$

Modelele sunt triplete formate din cel specific logicii dinamice, căruia i se adaugă relația de accesibilitate epistemică, $\mathcal{M} = (R, \sim, V)$. Interpretarea formulelor este aceeași ca cea din logicile clasice.

Sistemul deductiv este format din toate instanțele tautologiilor propoziționale, cărora li se adaugă axiomele din sistemul epistemic *S5* și din logica dinamică:

$$K_a(\varphi \to \psi) \to (K_a\varphi \to K_a\psi) \tag{2.35}$$
$$K_a\varphi \to \varphi \tag{2.36}$$
$$K_a\varphi \to K_aK_a\varphi \tag{2.37}$$
$$\neg K_a\varphi \to K_a\neg K_a\varphi \tag{2.38}$$
$$[\alpha](\varphi \to \psi) \to ([\alpha]\varphi \to [\alpha]\psi) \tag{2.39}$$
$$[\alpha](\varphi \wedge \psi) \leftrightarrow [\alpha]\varphi \wedge [\alpha]\psi \tag{2.40}$$
$$[\alpha \cup \beta]\varphi \leftrightarrow [\alpha]\varphi \wedge [\alpha]\psi \tag{2.41}$$
$$[\alpha;\beta]\varphi \leftrightarrow [\alpha][\beta]\varphi \tag{2.42}$$

Acest sistem logic este corect și complet, teorema de completitudine fiind demonstrată în cartea *Dynamic Epistemic Logic*, [9, p. 187-188].

2.2 DELP - sistem logic dinamic epistemic

În cadrul acestei secțiuni va fi prezentat sistemul *DELP*: sintaxa, semantica, rezultatele teoretice obținute și modelarea logicii BAN prin intermediul acestui sistem general. Sistemul logic *DELP* este construit pe baza logicii dinamice epistemice [9], căreia i se adaugă o gramatică de mesaje, o mulțime suplimentară în model pentru sistemul de inferență al cunoștințelor, și un operator care interpretează un mesaj ca o formulă.

2.2.1 Sintaxa

Considerăm mulțimile Φ mulțimea formulelor și Π mulțimea programelor din logica dinamică. Avem Φ_0 mulțimea formulelor atomice.

În locul programelor atomice din logica dinamică epistemică, vom considera mulțimea Π_0 ca fiind formată din acțiunile *sent* și *recv* din cadrul rundelor protocoalelor, astfel că

$$\Pi_0 := \{sent_i, recv_i\}|_{i \in Agent} \tag{2.43}$$

Mesajele din sistem conțin texte clare, chei, *nonce*-uri, identități de agenți, iar operațiile posibile sunt concatenarea și criptarea. Gramatica mesajelor este următoarea:

$$\mu ::= t \mid k \mid n \mid i \mid (\mu, \mu) \mid \{\mu\}_\mu \tag{2.44}$$

Formulele sistemului *DELP* sunt cele din logica dinamică epistemică, peste care adăugăm operatorul @:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \to \varphi \mid K_i\varphi \mid [\alpha]\varphi \mid @\mu \tag{2.45}$$

Sistemul deductiv pe mesaje este similar celui de inferență pe cunoștințe din semantica operațională:

$$\frac{}{nonce(m)} \quad \frac{key_k(i,j)}{key_k(j,i)} \quad \frac{\mu_1 \quad \mu_2}{(\mu_1, \mu_2)} \quad \frac{t \quad k}{\{t\}_k} \quad \frac{\{t\}_k \quad k}{t} \quad \frac{t_1, t_2, ..., t_n}{f(t_1, t_2, ..., t_n)} \tag{2.46}$$

2.2.2 Semantica

Modelele utilizate pentru interpretarea formulelor sunt modelele Kripke utilizate pentru logica dinamică epistemică, $\mathcal{M} = (R, \sim, V)$, pe care le extindem cu mulțimea *Exp*, mulțimea de cunoștințe în urma acțiunilor din cadrul rundelor protocolului (*expectations*).

Definiția 1. *Fie $\mathcal{M} = (R, \sim, V, Exp)$ un model pentru DELP, unde*

1. *R reprezintă mulțimea lumilor, a rundelor protocolului;*
2. *$\sim := \bigcup_{i \in Agent} \sim_i$ reprezintă relațiile de echivalență din logica epistemică;*
3. *V este funcția de evaluare din logica dinamică, $V(\varphi) \subset R$ pentru orice $\varphi \in \Phi$, iar $V(\alpha) \subset R \times R$ pentru $\alpha \in \Pi$;*
4. *Exp reprezintă mulțimea cunoștințelor, astfel încât pentru orice stare s din R, $Exp(s)$ reprezintă mulțimea tuturor cunoștințelor inferate până la runda s a protocolului, inclusiv.*

Cu modelele definite, vom interpreta formula $@\mu$ astfel:

$$\mathcal{M}, s \models @\mu \iff \mu \in Exp(s) \tag{2.47}$$

Restul formulelor au interpretarea din logica dinamică epistemică:

$$\mathcal{M}, s \models p \iff v \in V(s) \tag{2.48}$$
$$\mathcal{M}, s \models \varphi \wedge \psi \iff \mathcal{M}, s \models \varphi \text{ și } \mathcal{M}, s \models \psi \tag{2.49}$$
$$\mathcal{M}, s \models \neg\varphi \iff \mathcal{M}, s \not\models \varphi \tag{2.50}$$
$$\mathcal{M}, s \models K_i\varphi \iff \text{pentru orice } t \text{ astfel încât } s \sim_i t, \text{ avem } \mathcal{M}, t \models \varphi \tag{2.51}$$
$$\mathcal{M}, s \models [\alpha]\varphi \iff \text{pentru orice } t \in R \text{ astfel încât } (s,t) \in V(\alpha), \text{ atunci } \mathcal{M}, t \models \varphi \tag{2.52}$$

2.2.3 Sistemul deductiv

Sistemul deductiv este format din toate tautologiile logicii propoziționale, cărora li se adaugă:

1. Sistemul deductiv din logica dinamică:

$$[\alpha](\varphi \to \psi) \to ([\alpha]\varphi \to [\alpha]\psi) \tag{2.53}$$
$$[\alpha](\varphi \wedge \psi) \leftrightarrow [\alpha]\varphi \wedge [\beta]\psi \tag{2.54}$$
$$[\alpha \cup \beta]\varphi \leftrightarrow [\alpha]\varphi \wedge [\alpha]\psi \tag{2.55}$$
$$[\alpha;\beta]\varphi \leftrightarrow [\alpha][\beta]\varphi \tag{2.56}$$

2. Distributivitatea operatorului epistemic asupra implicației:

$$K(\varphi \to \psi) \to (K\varphi \to K\psi) \tag{2.57}$$

3. Trei axiome specifice ale sistemului *DELP*:

$$@\{m\}_k \wedge @key_k(i,j) \to [send_i]@m \vee [send_j]@m \tag{2.58}$$
$$[send_i]@m \vee [recv_i]@m \to K_i@m \tag{2.59}$$
$$@key_k(i,j) \to K_i@k \vee K_j@k \tag{2.60}$$

În continuare, precizăm următoarele observații asupra axiomelor introduse pentru logica *DELP*, în afara celor din logica dinamică epistemică:

Observația 1. *Prima axiomă specifică a sistemului reprezintă o axiomă de onestitate a agenților participanți; necesitatea acesteia este evidențiată în modelarea logicii BAN: dacă există un mesaj criptat cu cheia de comunicare k, iar cheia de comunicare k este o cheie știută de agenții i și j, atunci mesajul este transmis doar de unul dintre cei doi.*

Observația 2. *A doua axiomă este necesară pentru a avea o corespondență între stări: dacă agentul i efectuează o acțiune în cadrul protocolului (sau transmite, sau recepționează un mesaj), atunci el știe acel mesaj. În cazul în care mesajul este unul criptat, chiar dacă agentul i recepționează, de exemplu, $\{m\}_k$, el va putea infera $K_i@\{m\}_k$, și nu $K_i@m$.*

Observația 3. *A treia axiomă este o axiomă pentru modelarea protocoalelor cu cheie simetrică: dacă cheia k este o cheie de comunicare între agenții i și j, atunci fiecare dintre cei doi o cunoaște.*

Corectitudinea sistemului este dată de corectitudinea logicii dinamice epistemice [9, p. 187-188], astfel că vom demonstra doar corectitudinea axiomelor (2.57)-(2.59).

Lema 1. $@\{m\}_k \wedge @key_k(i,j) \to [send_i]@m \vee [send_j]@m$.

Demonstrație. Fie $\mathcal{M} = (R, \sim, V, Exp)$ un model pentru *DELP* și $s \in R$ o stare arbitrară. Atunci, dacă

$$\mathcal{M}, s \models @\{m\}_k \wedge @key_k(i,j) \iff \{m\}_k \in Exp(s) \text{ și } key_k(i,j) \in Exp(s)$$

Din regulile de inferență din $Exp(s)$, avem că $m \in Exp(s)$. Dacă mesajul m există în starea s, înseamnă că există o stare t astfel încât sau $(s,t) \in V(send_i)$, sau $(s,t) \in V(send_j)$.

Lema 2. $[send_i]@m \vee [recv_i]@m \to K_i@m$.

Demonstrație. Fie $\mathcal{M} = (R, \sim, V, Exp)$ un model pentru *DELP* și $s \in R$ o stare arbitrară. Atunci,

$\mathcal{M}, s \models [send_i]@m \iff$ pentru orice t astfel încât $(s,t) \in V(send_i)$, $\mathcal{M}, t \models @m$

i.e. $m \in Exp(t)$. În cadrul rundelor, starea t este accesibilă din s prin intermediul uneia dintre acțiuni, astfel încât sau $m \in Exp(t)$, sau $m \in Exp(t')$, unde $(s,t') \in V(recv_i)$.

Lema 3. $@key_k(i,j) \to K_i@m \vee K_j@m$

Demonstrație. Fie $\mathcal{M} = (R, \sim, V, Exp)$ un model pentru *DELP* și $s \in R$ o stare arbitrară. Atunci,

$$\mathcal{M}, s \models @key_k(i,j) \iff key_k(i,j) \in Exp(s)$$

Din regulile de inferență, avem și că $key_k(j,i) \in Exp(s)$, iar concluzia este imediată.

2.2.4 Completitudinea

Pentru a demonstra completitudinea *DELP*, utilizăm rezultatele deja știute din logica dinamică epistemică, alături de ideile din articolul *Hidden protocols*, de van Ditmarsch, Ghosh, Verbrugge și Wang [8]. Definim următoarele:

Definiția 2. ***[Modelul restricționat]****. Fie μ un mesaj și $\mathcal{M} = (R, \sim, V, Exp)$ un model pentru DELP. Atunci, modelul* ***restricționat*** *(sau actualizat) este*

$$M|_\mu = (R', \sim', V', Exp')$$

unde $R' = \{s \mid Exp(s) - \mu \neq \emptyset\}$, $\sim'_i = \sim_i |_{R' \times R'}$, $V' = V|_{R'}$, iar $Exp'(s) = Exp(s) - \mu$.

Definiția 3. ***[Modele temporale]*** *Pentru un model $\mathcal{M} = (R, \sim, V, Exp)$ definim modelul*

$$ET(\mathcal{M}) = (H, \to, \sim', V')$$

unde:

- $H = \{(s,m) \mid s \in R,\ m \in Exp(s)\}$;
- $(s,m) \to (s',m')$ *dacă și numai dacă* $s = s'$ *și* $m \to m'$;
- $(s,m) \sim' (s',m')$ *dacă și numai dacă* $s \sim s'$ *și* $m \equiv m'$ *unde* $\equiv$ *este echivalența logică;*
- $p \in V'(s,m)$ *dacă și numai dacă* $p \in V(s)$

Având un astfel de model $\mathcal{N} := ET(\mathcal{M})$, definim următoarea semantică:

$$\mathcal{N}, w \models p \iff p \in V(w) \tag{2.61}$$
$$\mathcal{N}, w \models \neg\varphi \iff \mathcal{N}, w \not\models \varphi \tag{2.62}$$
$$\mathcal{N}, w \models \varphi \wedge \psi \iff \mathcal{N}, w \models \varphi \text{ și } \mathcal{N}, w \models \psi \tag{2.63}$$
$$\mathcal{N}, w \models_{EPDL} K_i\varphi \iff \text{pentru orice } v \in \mathcal{N},\ \text{dacă } w \sim_i v,\ \text{atunci } \mathcal{N}, v \models_{EPDL} \varphi \tag{2.64}$$
$$\mathcal{N}, w \models_{EPDL} [\alpha]\varphi \iff \text{pentru orice } \mu \in Exp(\alpha), w \to v \text{ implică } \mathcal{N}, w \models_{EPDL} \varphi \tag{2.65}$$

Bisimilaritatea. Conform articolului [8, Def. 11], avem că relația binară ρ definită pe $\mathcal{M} \times \mathcal{N}$, unde $\mathcal{M} = (R, \sim, V, Exp)$, respectiv $\mathcal{N} = (R', \sim', V', Exp')$ se numește bisimilaritate dacă pentru orice $v \in R$, $v' \in R'$ avem că, dacă $v \rho v''$,

atunci:

$$\textbf{Invarianță propozițională } V(v) = V'(v') \tag{2.66}$$
$$\textbf{Invarianță peste observații } Exp(v) = Exp(v') \tag{2.67}$$
$$\textbf{Zig } v \sim_i w \in \mathcal{M} \Longrightarrow \text{există } w' \in \mathcal{N} \text{ astfel încât } v' \sim'_i w' \text{ și } w\rho w' \tag{2.68}$$
$$\textbf{Zag } v' \sim'_i w' \in \mathcal{N} \Longrightarrow \text{există } w \in \mathcal{M} \text{ astfel încât } v \sim_i w \text{ și } w\rho w' \tag{2.69}$$

Teorema 1. ***[Invarianța la bisimilaritate]*** *Pentru două stări aparținând unor modele Kripke extinse $\mathcal{M}, v$ și $\mathcal{N}, v'$, următoarele două afirmații sunt echivalente:*

$$(i)\ \mathcal{M}, v \leftrightarrow \mathcal{N}, v' \tag{2.70}$$
$$(ii)\ \textit{pentru orice } \varphi\textit{:}\ \mathcal{M}, v \models \varphi \Longleftrightarrow \mathcal{N}, v' \models \varphi \tag{2.71}$$

unde $\leftrightarrow$ este notația de bisimilaritate între modele.

Demonstrația este oferită în articolul *Hidden protocols*, [8, Prop. 12].

Teorema 2. ***[completitudine]*** *Fie $\mathcal{M} = (R, \sim, V, Exp)$ un model Kripke extins, ε cunoștința inițială și φ o formulă. Atunci*

$$\mathcal{M}, v \models \varphi \Longleftrightarrow ET(\mathcal{M}), (s, \varepsilon) \models_{EPDL} \varphi \tag{2.72}$$

Demonstrație. Vom demonstra prin inducție după φ.

Cazul $\varphi = \neg\psi$.

$$\begin{aligned} ET(\mathcal{M}),\ (v, \varepsilon) &\models_{EPDL} \neg\psi \\ &\Longleftrightarrow ET(\mathcal{M}), (v, \varepsilon) \not\models \psi \\ &\Longleftrightarrow \mathcal{M}, v \not\models \psi \end{aligned}$$

din modul de construcție al modelului $ET(\mathcal{M})$ (din faptul că evaluarea formulelor în $ET(\mathcal{M})$ este aceeași din $\mathcal{M}$).

Cazul $\varphi = \psi_1 \wedge \psi_2$ este similar cazului anterior, fiind o demonstrație imediată din modul de construcție al modelului $ET(\mathcal{M})$.

Cazul epistemic, $\varphi = K_i\psi$.

$$\begin{aligned} &ET(\mathcal{M}),\ (v, \varepsilon) \models_{EPDL} K_i\psi \\ &\qquad \Longleftrightarrow \text{pentru orice } w \in H,\ \text{dacă } v \sim'_i w,\ \text{atunci } ET(\mathcal{M}), (w, \varepsilon) \models_{EPDL} \psi \\ &\qquad \Longleftrightarrow \text{pentru orice } w \in R,\ \text{dacă } v \sim_i w,\ \text{atunci } \mathcal{M}, w \models_{EPDL} \psi \\ &\qquad \Longleftrightarrow \mathcal{M}, v \models K_i\psi \end{aligned}$$

din construcția relației de echivalență epistemică, $(s, m) \sim' (s', m')$ dacă și numai dacă $s \sim s'$ și $m \equiv m'$ unde $\equiv$ este echivalența logică, utilizând faptul că ε, cunoștința inițială, este prin reflexitivtate echivalentă cu ea însăși.

Cazul $\varphi = [\alpha]\psi$. Presupunem, prin reducere la absurd, că $\mathcal{M}, v \models [\alpha]\psi$, dar $ET(\mathcal{M}), (v, \varepsilon) \not\models_{EPDL} [\alpha]\psi$.

Atunci, înseamnă că există $m \in Exp(v)$ astfel încât $ET(\mathcal{M}), (v, m) \not\models \psi$.

Din construcția $ET(\mathcal{M})$, cum mulțimea lumilor era $H = \{(s, m) \mid s \in R,\ m \in Exp(s)\}$, înseamnă că $m \in Exp(v)$.

Cum m este un mesaj, înseamnă că există modelul restricționat $\mathcal{M}|_m$. Din bisimilaritate, avem că $ET(\mathcal{M}|_m), (v, \varepsilon)$ este bisimilar cu $ET(\mathcal{M}), (v, m)$.

Atunci, avem că $ET(\mathcal{M}|_m), (v, \varepsilon) \models \neg\psi$. Din ipoteza de inducție, obținem că $\mathcal{M}, v \models \neg\psi$, ceea ce contrazice $\mathcal{M}, v \models [\alpha]\psi$.

Rezultat. Am obținut, astfel, că sistemul *DELP* este un sistem complet.

2.3 Modelarea logicii BAN

În această secțiune vom demonstra că regulile din logica BAN sunt corecte în *DELP*. Pentru aceasta, vom avea nevoie de o translatare, pe care o vom construi pe baza articolului [5], cu modificările necesare, conform [8].

2.3.1 Translatarea formulelor din logica BAN

1. Formula $i \mid\equiv m$ se traduce ca fiind $K_i@m$, și înseamnă că agentul i știe mesajul m în starea curentă.
2. Formula $i \triangleleft m$ înseamnă că agentul i vede mesajul m, ca fiind primit într-una dintre rundele protocolului - mesajul m nu este o cunoștință până nu este recepționat. Avem, astfel, translatarea $[recv_i]@m$.
3. Formula $i \mid\sim m$ înseamnă că agentul i a transmis mesajul m: $[sent_i]@m$.
4. Formula $i \Rightarrow m$ înseamnă că agentul i are jurisdicție asupra mesajului m, adică faptul că îl știe implică și faptul că mesajul este adevărat: $K_i@m \rightarrow @m$.
5. Formula $i \overset{k}{\leftrightarrow} j$ înseamnă că agenții i își j partajează cheia de comunicare k, deci este o cunoștință comună între cei doi: $@key_k(i,j)$.
6. Formula $\#(m)$ înseamnă că mesajul m este abia generat, $@nonce(m)$.

2.3.2 Corectitudinea regulilor BAN

În această secțiune, vom demonstra corectitudinea în sistemul *DELP* a tuturor regulilor din logica *BAN*.

Regula MM-SK (message meaning rules for shared key)

$$R_1.\frac{i \mid\equiv j \overset{k}{\leftrightarrow} i \quad i \triangleleft \{m\}_k}{i \mid\equiv j \mid\sim m}$$

Lema 4. *Regula MM-SK este corectă.*

Demonstrație. Fie $\mathcal{M} = (R, \sim, V, Exp)$ un model Kripke extins și o stare arbitrară $v \in R$. Avem de arătat că dacă $\mathcal{M}, v \models K_i(@key_k(i,j)) \wedge [recv_i]@\{m\}_k$ atunci $\mathcal{M}, v \models K_i([sent_j]@m)$. Atunci:

$\mathcal{M}, v \models K_i@key_k(i,j) \wedge [recv_i]@\{m\}_k \Longrightarrow \mathcal{M}, v \models K_i@key_k(i,j) \wedge K_i@\{m\}_k$ (cf. A_2)

Atunci, pentru orice lume $w \in R$, astfel încât $v \sim_i w$, avem

$$\mathcal{M}, w \models @key_k(i,j) \wedge @\{m\}_k \Longrightarrow \mathcal{M}, w \models [send_j]@m$$

Astfel, obtinem concluzia, și anume că $\mathcal{M}, v \models K_i[send_j]@m$.

Regula BC (belief and components)

$$R_2.\frac{i \mid\equiv j \mid\sim (m, m')}{i \mid\equiv j \mid\sim m}$$

Lema 5. *Regula BC este corectă.*

Demonstrație. Fie $\mathcal{M} = (R, \sim, V, Exp)$ un model Kripke extins și o stare arbitrară $v \in R$. Avem de arătat că dacă $\mathcal{M}, v \models K_i[send_j]@(m, m')$ atunci $\mathcal{M}, v \models K_i([sent_j]@m)$. Atunci:

$\mathcal{M}, v \models K_i[send_j]@(m, m')$

$\iff$ pentru orice $w \in R,\ v \sim_i w, \mathcal{M}, w \models [send_j]@(m, m')$

$\iff$ pentru orice $w \in R,\ v \sim_i w,$ orice $s,\ (w, s) \in V(send_j),\ \mathcal{M}, s \models @(m, m')$

Din regulile de inferență peste Exp, dacă $\mathcal{M}, s \models @(m, m')$, i.e. $(m, m') \in Exp(s)$, în particular $m \in Exp(s)$, deci $\mathcal{M}, s \models @m$, în concluzie $\mathcal{M}, v \models K_i[send_j]@m$.

Regula NC (nounce verification)

$$R_3.\frac{i \mid\equiv \#(m) \quad i \mid\equiv j \mid\sim m}{i \mid\equiv j \mid\equiv m}$$

Lema 6. *Regula NC este corectă.*

Demonstrație. Fie $\mathcal{M} = (R, \sim, V, Exp)$ un model Kripke extins și o stare arbitrară $v \in R$. Avem de arătat că dacă $\mathcal{M}, v \models K_i@nonce(m) \wedge K_i[send_j]@m$ atunci $\mathcal{M}, v \models K_iK_j@m$. Atunci, pentru orice lume $w \in R$ pentru care $v \sim_i w$ avem:

$$\mathcal{M}, w \models @nonce(m) \wedge [send_j]@m \Longrightarrow \mathcal{M}, w \models @nonce(m) \wedge K_j@m \ (cf.\ A_2)$$

Atunci, $\mathcal{M}, v \models K_i@nonce(m) \wedge K_iK_j@m$, deci $\mathcal{M}, v \models K_iK_j@m$, ceea ce era de arătat.

Regula Jurisdicției

$$R_4.\frac{i \mid\equiv j \Rightarrow m \quad i \mid\equiv j \mid\equiv m}{i \mid\equiv m}$$

Lema 7. *Regula Jurisdicției este corectă.*

Demonstrație. Fie $\mathcal{M} = (R, \sim, V, Exp)$ un model Kripke extins și o stare arbitrară $v \in R$. Avem de arătat că dacă $\mathcal{M}, v \models K_i(K_j@m \rightarrow @m) \wedge K_iK_j@m$ atunci $\mathcal{M}, v \models K_i@m$. Este suficient să aplicăm distributivitatea lui K asupra implicației, $K(\varphi \rightarrow \psi) \rightarrow (K\varphi \rightarrow K\psi)$, cu substituțiile uniforme $\varphi := K_j@m$ și $\psi := @m$. Cum $K_i(K_j@m \rightarrow @m) \rightarrow (K_iK_j@m \rightarrow K_i@m)$ și știind $K_iK_j@m$, aplicând *modus ponens* obținem $K_i@m$.

Regula SC1 (seeing and components)

$$R_5.\frac{i \lhd (m, m')}{i \lhd m}$$

Lema 8. *Regula SC1 este corectă.*

Demonstrație. Fie $\mathcal{M} = (R, \sim, V, Exp)$ un model Kripke extins și o stare arbitrară $v \in R$. Avem de arătat că dacă $\mathcal{M}, v \models [recv_i]@(m, m')$ atunci $[recv_i]@m$. Din inferența peste Exp, pentru orice w astfel încât $(v, w) \in V(recv_i)$, dacă $(m, m') \in Exp(w)$, în particular $m \in Exp(w)$, deci $\mathcal{M}, w \models @m$, de unde $\mathcal{M}, v \models [recv_i]@m$.

Regula SC2 (seeing and components)

$$R_6.\frac{i \mid\equiv i \overset{k}{\leftrightarrow} j \quad i \lhd \{m\}_k}{i \lhd m}$$

Lema 9. *Regula SC2 este corectă.*

Demonstrație. Fie $\mathcal{M} = (R, \sim, V, Exp)$ un model Kripke extins și o stare arbitrară $v \in R$. Avem de arătat că dacă $\mathcal{M}, v \models K_i@key_k(i, j) \wedge [recv_i]@\{m\}_k$ atunci $\mathcal{M}, v \models [recv_i]@m$. Pentru $\mathcal{M}, v \models [recv_i]@\{m\}_k$ inferăm $\mathcal{M}, v \models K_i[recv_i]@\{m\}_k$. Atunci, aplicând (A_3), din $\mathcal{M}, v \models K_i@key_k(i, j)$ obținem $\mathcal{M}, v \models K_i@k$, iar pentru toate lumile $w \in R$, $v \sim_i w$, avem că $\{m\}_k \in Exp(w)$ și $k \in Exp(w)$. Din inferența peste Exp, $m \in Exp$, deci $\mathcal{M}, w \models @m$, iar $\mathcal{M}, v \models K_i@m$. Cum $\mathcal{M}, v \models [recv_i]@m$ implică, din (A_2), că $\mathcal{M}, v \models K_i@m$, obținem concluzia.

Regula NC (nounces concatenation)

$$R_7.\frac{i \mid\equiv \#(m)}{i \mid\equiv \#(m, m')}$$

Lema 10. *Regula NC este corectă.*

Demonstrația este imediată din regulile de inferență din Exp: dacă $nonce(m) \in Exp$, atunci $m \in Exp$, iar pentru orice $m' \in Exp$, avem și ca $(m, m') \in Exp$. Pentru orice model Kripke extins $\mathcal{M} = (R, \sim, V, Exp)$ și pentru o stare arbitrară $v \in R$, în orice lume $w \in R$ pentru care $v \sim_i w$, obținem ca $\mathcal{M}, w \models @nonce(m)$, deci $\mathcal{M}, w \models @m$, de unde $\mathcal{M}, w \models (m, m')$ pentru orice $m' \in Exp(w)$. Atunci $\mathcal{M}, v \models K_i@nonce(m, m')$.

Regula CK (Commutativity of keys)

$$R_9.\frac{i \mid\equiv i \overset{k}{\longleftrightarrow} j}{i \mid\equiv j \overset{k}{\longleftrightarrow} i}$$

Lema 11. *Regula CK este corectă.*

Demonstrația este din inferența peste Exp, dacă $key_k(i, j) \in Exp$, atunci $key_k(j, i) \in Exp$.

Capitolul 3

Implementare în Lean

În acest capitol vom prezenta demonstratorul interactiv *Lean*, vom implementa sintaxa și sistemul deductiv pentru *DELP* și vom verifica și corectitudinea regulilor BAN, utilizând demonstrațiile deja prezentate în capitolul anterior.

3.1 Prezentare generală

Lean este un proiect al *Microsoft Research Redmond*, început în anul 2013, dezvoltat pe baza *teoriei tipurilor dependente*, un sistem suficient de puternic pentru a putea demonstra majoritatea teoremelor convenționale din matematică. Sistemul pe care este bazat *Lean* este *Calculus of Constructions*. [1]

În implementarea oferită în cadrul acestei lucrări, am utilizat *Lean* pentru a demonstra sintactic anumite rezultate. Prezentăm, în continuare, câteva exemple de teoreme demonstrate sintactic, în logica propozițională.

Exemplu. Putem demonstra că $\vdash p \to p$ în mai multe moduri. De exemplu, putem utiliza teoria tipurilor dependente:

```
1  example : Π p : Prop, p → p := λ p (hp : p), hp.
```

Putem, în plus, să scriem mult mai explicit demonstrația:

```
1  example : Π p : Prop, p → p :=
2    assume p : Prop,
3    assume hp : p,
4    show p, from hp.
```

De asemenea, în loc să considerăm p un tip dependent, putem specifica de la început că p este *Prop*.

```
1  example { p : Prop } : p → p :=
2    assume hp : p,
3    show p, from hp.
```

Utilizând această scriere, putem demonstra următoarele teoreme:

- $(p \wedge q) \vdash (p \to q) \vee (q \to p)$

```
example { p q : Prop } : (p ∧ q) → ((p → q) ∨ (q → p)) :=
  assume h : (p ∧ q),
  have hp : p, from and.elim_left h,
  have hq : q, from and.elim_right h,
  have h : p → q, from assume p, hq,
  or.intro_left (q → p) h.
```

- $(p \wedge q) \wedge r \vdash s \vee (q \vee t)$

```
example { p q r s t : Prop } : ((p ∧ q) ∧ r) → (s ∨ (q ∨ t)) :=
  assume h : (p ∧ q) ∧ r,
  have   hpq : (p ∧ q), from and.elim_left h,
  have   hq  : q,       from and.elim_right hpq,
  have   hqt : q ∨ t,   from or.intro_left t hq,
  show   s ∨ (q ∨ t),   from or.intro_right s hqt.
```

3.2 Implementarea DELP

În cadrul acestei secțiuni, vom prezenta implementarea sistemului *DELP* în *Lean*. Componentele principale pentru reprezentare sunt tipurile inductive pentru mesaje, programe și formule, cât și sistemul deductiv. După implementarea acestuia, vom verifica, de data aceasta automat, corectitudinea regulilor BAN. Ultimul capitol este dedicat implementării protocolului *Needham-Schroeder*, cu scopul de a demonstra anumite proprietăți ale acestuia - de exemplu, cunoașterea unui secret comun.

Ideile de implementare respectă structura prezentată în articolul [2], care propune o implementare în *Lean* a sistemului epistemic *S5*.

3.2.1 Limbajul

Pentru a reprezenta sistemul *DELP*, vom defini următoarele tipuri inductive:

```
inductive message (σ : ℕ) : Type
  | null : fin σ → message
  | conc : message → message → message
  | nonc : message → message
  | keys : message → message → message → message
  | encr : message → message → message
  | decr : message → message → message
  | tupl : message → message → message
```

```
inductive program (σ : ℕ) : Type
  | skip : program
```

```
  | secv : program → program → program
  | reun : program → program → program
  | send : message σ → program
  | recv : message σ → program
```

```
inductive form (σ : ℕ) : Type
  | atom : fin σ → form
  | botm : form
  | impl : form → form → form
  | know : message σ → form → form
  | prog : program σ → form → form
  | mesg : message σ → form
  | and : form → form → form
  | or : form → form → form
```

Pentru a păstra o corespondență ușoară între formulele din logică și implementare, vom defini următoarele notații:

```
notation p `→` q       := form.impl p q
notation `ι` μ         := form.mesg μ
notation p `∧` q       := form.and p q
notation p `∨` q       := form.or p q
notation `K` m `,` p   := form.know m p
notation `[` α `]` φ   := form.prog α φ

notation `·`           := {}
notation Γ ` ∪ ` p     := set.insert p Γ

notation m `||` n      := message.tupl m n
notation `{` m `}` k   := message.encr m k
```

3.2.2 Sistemul deductiv

Vom defini un context pentru sistemul deductiv, o mulțime Γ de enunțuri:

```
def ctx (σ : ℕ) : Type := set (form σ)
```

În acest moment, putem defini toate axiomele și regulile de deducție ale sistemului:

```
inductive proof (σ : ℕ) : ctx σ → form σ → Prop
-- propositional logic
  | ax { Γ } { p } (h : p     Γ) : proof Γ p
  | pl1 { Γ } { p q : form σ } : proof Γ (p → (q → p))
-- epystemic logic
  | kand { Γ } { i : message σ } { p q : form σ } : proof Γ (((K i, p) ∧
      (K i, q)) → (K i, (p ∧ q)))
  | ktruth { Γ } { i : message σ } { φ : form σ } : proof Γ ((K i, φ) →
    φ)
```

```
8    | kdist { Γ } { i : message σ } { φ ψ : form σ } : proof Γ ((K i, (φ →
         ψ)) → ((K i, φ) → (K i, ψ)))
9  -- dynamic logic
10   | progrdistr { Γ } { α : program σ } { φ ψ : form σ } : proof Γ ([α](φ
         → ψ) → ([α]ψ → [α]ψ))
11   | pdtruth { Γ } { α : program σ } { φ : form σ } : proof Γ (([α]φ) → φ
         )
12 -- delp axioms
13   | honestyright { Γ } { m k i j : message σ } : proof Γ ((ι (k.keys i j
         )) ∧ (ι ({ m } k)) → ([send j](ι m)))
14   | knowreceive { Γ } { m i : message σ } : proof Γ (([recv i](ι m)) → (
         K i, (ι m)))
15   | knowsend { Γ } { m i : message σ } : proof Γ (([send i](ι m)) → (K i
         , (ι m)))
16   | knowreceivef { Γ } { i : message σ } { φ : form σ } : proof Γ (([
         recv i]φ) → (K i, φ))
17   | knowsendf { Γ } { i : message σ } { φ : form σ } : proof Γ (([send i
         ]φ) → (K i, φ))
18 -- modus ponens and generalization
19   | mp { Γ } { p q : form σ } (hpq : proof Γ (p → q)) (hp : proof Γ p) :
          proof Γ q
20   | kgen { Γ } { φ : form σ } { i : message σ } (h : proof Γ φ) : proof
         Γ (K i, φ)
21   | pdgen { Γ } { φ : form σ } { α : program σ } (h : proof Γ φ) : proof
          Γ ([α]φ)
```

3.3 Verificarea regulilor BAN

Având corespondențele formulelor din logica BAN în sistemul *DELP* și având și sistemul definit în *Lean*, putem verifica formal corectitudinea regulilor de deducție. În această secțiune vom prezenta, complet, toate aceste demonstrații.

Regula MM-SK (message meaning rules for shared key)

$$R_1.\frac{i \mid\equiv j \overset{k}{\leftrightarrow} i \quad i \triangleleft \{m\}_k}{i \mid\equiv j \mid\sim m}$$

```
1  lemma MMSK (σ : ℕ) { m k i j : message σ } { Γ : ctx σ }
2    : (σ-Γ ⊢ ((K i, (ι (k.keys i j))) ∧ ([recv i](ι { m } k)))) → (σ-Γ ⊢ (
         K i, ([send j](ι m)))) :=
3    assume h0 : σ-Γ ⊢ (K i, (ι (k.keys i j))) ∧ ([recv i](ι { m } k)),
4    have h1 : σ-Γ ⊢ K i, (ι (k.keys i j)),
5      from @andleft σ Γ (K i, (ι (k.keys i j))) ([recv i](ι {m}k)) h0,
6    have h2 : σ-Γ ⊢ [recv i](ι { m } k),
7      from @andright σ Γ (K i, ι (k.keys i j)) ([recv i](ι {m}k)) h0,
```

```
  have h3 : σ-Γ ⊢ ([recv i](ι{m}k)) → (K i, ι{m}k),
    from @knowreceive σ Γ ({m}k) i,
  have h4 : σ-Γ ⊢ K i, ι{m}k,
    from @mp σ Γ ([recv i](ι{m}k)) ((K i, ι{m}k)) h3 h2,
  have andk : σ-Γ ⊢ (K i, ι k.keys i j) ∧ (K i, ι{m}k) → (K i, (ι k.keys
      i j) ∧ ι{m}k),
    from @kand σ Γ i (ι (k.keys i j)) (ι{m}k),
  have and1k : σ-Γ ⊢ ((K i, ι k.keys i j) ∧ (K i, ι{m}k)),
    from @andintro σ Γ (K i, ι (k.keys i j)) (K i, ι{m}k) h1 h4,
  have h5 : σ-Γ ⊢ (K i, (ι k.keys i j) ∧ ι{m}k),
    from @mp σ Γ ((K i, ι k.keys i j) ∧ (K i, ι{m}k)) ((K i, (ι k.keys i
        j) ∧ ι{m}k)) andk and1k,
  have h6 : σ-Γ ⊢ (K i, ((ι k.keys i j) ∧ ι{m}k)) → ((ι k.keys i j) ∧ ι{
      m}k),
    from @ktruth σ Γ i ((ι k.keys i j) ∧ ι{m}k),
  have h7 : σ-Γ ⊢ (ι k.keys i j) ∧ ι{m}k,
    from @mp σ Γ (K i, ((ι k.keys i j) ∧ ι{m}k)) ((ι k.keys i j) ∧ ι{m}k
       ) h6 h5,
  have h8 : σ-Γ ⊢ (ι (k.keys i j)) ∧ (ι ({m}k)) → ([send j](ι m)),
    from @honestyright σ Γ m k i j,
  have h9 : σ-Γ ⊢ [send j](ι m),
    from @mp σ Γ ((ι k.keys i j) ∧ ι{m}k) ([send j](ι m)) h8 h7,
  show σ-Γ ⊢ K i, [send j](ι m),
    from @kgen σ Γ ([send j](ι m)) i h9.
```

Regula BC (belief and components)

$$R_2.\frac{i \mid\equiv j \mid\sim (m, m')}{i \mid\equiv j \mid\sim m}$$

```
lemma BC (σ : ℕ) { m m' i j : message σ } { Γ : ctx σ }
  : (σ-Γ ⊢ K i, [send j](ι m||m')) → (σ-Γ ⊢ K i, [send j](ι m)) :=
  assume h0 : σ-Γ ⊢ K i, [send j](ι m||m'),
  have h1 : σ-Γ ⊢ (K i, [send j](ι m||m')) → ([send j](ι m||m')),
    from @ktruth σ Γ i ([send j](ι m||m')),
  have h2 : σ-Γ ⊢ ([send j](ι m||m')),
    from @mp σ Γ (K i, [send j](ι m||m')) ([send j](ι m||m')) h1 h0,
  have h3 : σ-Γ ⊢ ([send j](ι m||m')) → (ι m||m'),
    from @pdtruth σ Γ (send j) (ι m||m'),
  have h4 : σ-Γ ⊢ ι m||m',
    from @mp σ Γ ([send j](ι m||m')) (ι m||m') h3 h2,
  have h5 : σ-Γ ⊢ ι m,
    from @messageleft σ Γ m m' h4,
  have h6 : σ-Γ ⊢ [send j](ι m),
    from @pdgen σ Γ (ι m) (send j) h5,
  show σ-Γ ⊢ K i, [send j](ι m),
    from @kgen σ Γ ([send j](ι m)) i h6.
```

Regula NC (nounce verification)

$$R_3.\frac{i \mid\equiv \#(m) \;\; i \mid\equiv j \mid\sim m}{i \mid\equiv j \mid\equiv m}$$

```
lemma NV (σ : ℕ) { m i j : message σ } { Γ : ctx σ }
  : (σ-Γ ⊢ (K i, ι nonc m) ∧ K i, [send j](ι m)) → (σ-Γ ⊢ K i, (K j, ι m
      )) :=
  assume h : σ-Γ ⊢ (K i, ι nonc m) ∧ K i, [send j](ι m),
  have h0 : σ-Γ ⊢ (K i, ι nonc m),
    from @andleft σ Γ (K i, ι nonc m) (K i, [send j](ι m)) h,
  have h1 : σ-Γ ⊢ K i, [send j](ι m),
    from @andright σ Γ (K i, ι nonc m) (K i, [send j](ι m)) h,
  have h2 : σ-Γ ⊢ (K i, ι m.nonc) → ι m.nonc,
    from @ktruth σ Γ i (ι nonc m),
  have h3 : σ-Γ ⊢ ι m.nonc,
    from @mp σ Γ (K i, ι m.nonc) (ι m.nonc) h2 h0,
  have h4 : σ-Γ ⊢ (K i, [send j](ι m)) → ([send j](ι m)),
    from @ktruth σ Γ i ([send j](ι m)),
  have h5 : σ-Γ ⊢ [send j](ι m),
    from @mp σ Γ (K i, [send j](ι m)) ([send j](ι m)) h4 h1,
  have h6 : σ-Γ ⊢ ([send j](ι m)) → (K j, ι m),
    from @knowsend σ Γ m j,
  have h7 : σ-Γ ⊢ K j, ι m,
    from @mp σ Γ ([send j](ι m)) (K j, ι m) h6 h5,
  show σ-Γ ⊢ K i, (K j, (ι m)),
    from @kgen σ Γ (K j, (ι m)) i h7.
```

Regula Jurisdicției

$$R_4.\frac{i \mid\equiv j \Rightarrow m \;\; i \mid\equiv j \mid\equiv m}{i \mid\equiv m}$$

```
lemma JR (σ : ℕ) { m i j : message σ } { Γ : ctx σ }
  : (σ-Γ ⊢ (K i, (K j, ι m) → (ι m)) ∧ (K i, K j, ι m)) → (σ-Γ ⊢ K i, ι
      m) :=
  assume h0 : σ-Γ ⊢ (K i, ((K j, ι m) → (ι m))) ∧ (K i, (K j, ι m)),
  have h1 : σ-Γ ⊢ K i, ((K j, ι m) → (ι m)),
    from @andleft σ Γ (K i, (K j, ι m) → (ι m)) (K i, K j, ι m) h0,
  have h2 : σ-Γ ⊢ (K i, K j, ι m),
    from @andright σ Γ (K i, ((K j, ι m) → (ι m))) (K i, K j, ι m) h0,
  have h3 : σ-Γ ⊢ (K i, ((K j, ι m) → (ι m))) → ((K i, K j, ι m) → (K i,
       ι m)),
    from @kdist σ Γ i (K j, ι m) (ι m),
  have h4 : σ-Γ ⊢ (K i, K j, ι m) → (K i, ι m),
    from @mp σ Γ (K i, (K j, ι m) → (ι m)) ((K i, K j, ι m) → (K i, ι m)
      ) h3 h1,
```

```
12    show σ-Γ ⊢ K i, ι m,
13      from @mp σ Γ (K i, K j, ι m) (K i, ι m) h4 h2.
```

Regula SC1 (seeing and components)

$$R_5.\frac{i \lhd (m, m')}{i \lhd m}$$

```
1   lemma SC1 (σ : ℕ) { m m' i : message σ } { Γ : ctx σ }
2     : (σ-Γ ⊢ [recv i]ι m||m') → (σ-Γ ⊢ [recv i]ι m) :=
3     assume h0 : σ-Γ ⊢ [recv i]ι m||m',
4     have h1 : σ-Γ ⊢ ([recv i]ι m||m') → (ι m||m'),
5       from @pdtruth σ Γ (recv i) (ι m||m'),
6     have h2 : σ-Γ ⊢ ι m||m',
7       from @mp σ Γ ([recv i]ι m||m') (ι m||m') h1 h0,
8     have h3 : σ-Γ ⊢ ι m,
9       from @messageleft σ Γ m m' h2,
10    show σ-Γ ⊢ [recv i]ι m,
11      from @pdgen σ Γ (ι m) (recv i) h3.
```

Regula SC2 (seeing and components)

$$R_6.\frac{i \mid\equiv i \overset{k}{\leftrightarrow} j \quad i \lhd \{m\}_k}{i \lhd m}$$

```
1   lemma SC2 (σ : ℕ) { m i j k : message σ } { Γ : ctx σ }
2     : (σ-Γ ⊢ (K i, ι k.keys i j) ∧ ([recv i]ι{m}k)) → (σ-Γ ⊢ [recv i]ι m)
        :=
3     assume h0 : σ-Γ ⊢ (K i, ι k.keys i j) ∧ ([recv i]ι{m}k),
4     have h1 : σ-Γ ⊢ (K i, ι k.keys i j),
5       from @andleft σ Γ (K i, ι k.keys i j) ([recv i]ι{m}k) h0,
6     have h2 : σ-Γ ⊢ ([recv i]ι{m}k),
7       from @andright σ Γ (K i, ι k.keys i j) ([recv i]ι{m}k) h0,
8     have h3 : σ-Γ ⊢ (K i, ι k.keys i j) → ι k.keys i j,
9       from @ktruth σ Γ i (ι k.keys i j),
10    have h4 : σ-Γ ⊢ ι k.keys i j,
11      from @mp σ Γ (K i, ι k.keys i j) (ι k.keys i j) h3 h1,
12    have h5 : σ-Γ ⊢ ([recv i]ι{m}k) → ι{m}k,
13      from @pdtruth σ Γ (recv i) (ι{m}k),
14    have h6 : σ-Γ ⊢ ι{m}k,
15      from @mp σ Γ ([recv i]ι{m}k) (ι{m}k) h5 h2,
16    have h7 : σ-Γ ⊢ ((ι k.keys i j) ∧ ι{m}k) → [send j]ι m,
17      from @honestyright σ Γ m k i j,
18    have h8 : σ-Γ ⊢ (ι k.keys i j) ∧ ι{m}k,
19      from @andintro σ Γ (ι k.keys i j) (ι{m}k) h4 h6,
20    have h9 : σ-Γ ⊢ [send j]ι m,
```

```
21      from @mp σ Γ ((ι k.keys i j) ∧ ι{m}k) ([send j]ι m) h7 h8,
22    have h10 : σ-Γ ⊢ ([send j]ι m) → ι m,
23      from @pdtruth σ Γ (send j) ι m,
24    have h11 : σ-Γ ⊢ ι m,
25      from @mp σ Γ ([send j]ι m) (ι m) h10 h9,
26    show σ-Γ ⊢ [recv i]ι m,
27      from @pdgen σ Γ (ι m) (recv i) h11.
```

Capitolul 4

Implementarea protocolului Needham-Schroeder

În acest capitol vom reaminti protocolul *Needham-Schroeder*, prezentat și la începutul acestei lucrări, cu scopul de a implementa specificațiile în limbajul *Lean* și de a verifica toate proprietățile de securitate pe care le putem demonstra în sistemul *DELP*.

4.1 Protocolul Needham-Schroeder

În cadrul acestei secțiuni vom reaminti protocolul Needham-Schroeder, prezentând schimbul de mesaje și modelarea lui din logica BAN. În final, vom oferi o implementare a acestuia în sistemul *DELP*.

4.1.1 Protocolul Needham-Schroeder și explicația schimbului de mesaje

Specificația protocolului este următoarea, pentru agenții participanți A, S și B (Alice, Server și Bob).

$$
\begin{aligned}
&A \to S : A, B, N_a \\
&S \to A : \{N_a, B, K_{ab}, \{K_{ab}, A\}_{K_{bs}}\}_{K_{as}} \\
&A \to B : \{K_{ab}, A\}_{K_{bs}} \\
&B \to A : \{N_b\}_{K_{ab}} \\
&A \to B : \{N_b - 1\}_{K_{ab}}
\end{aligned}
$$

O descriere a protocolului pe pașii prezentați este următoarea:

- Alice inițiază conexiunea cu Serverul, transmițând cine este, cu cine vrea să comunice și un *nonce*;
- Serverul îi transmite, criptat cu cheia comună dintre A și S, *nonce*-ul generat de A, identitatea lui B și cheia de comunicare dintre A și B, la care se adaugă un mesaj pe care îl poate decripta doar B (fiind criptat cu cheia publică dintre B și S), care conține cheia partajată de A și B și identitatea lui A. În acest mod, A nu poate să citească mesajul transmis de server către B, iar B află cu cine trebuie să comunice și cum poate comunica;
- A îi transmite lui B mesajul pe care nu-l putea decripta, primit de la Server;
- B decriptează mesajul, și îi transmite lui A un *nonce*, criptat cu cheia comună dintre A și B;
- A primește mesajul lui B, îl decriptează, și îl retransmite, aplicându-i o funcție simplă - în acest caz, îl decrementează. Această etapă este utilă în două situații: este o primă protecție pe un *reply attack* și, în plus, arată că agenții sunt încă activi în cadrul sesiunii - verifică un *claim* de *alive*.

4.1.2 Modelarea protocolului în logica BAN

Pentru a putea modela protocolul cu ajutorul logicii BAN, trebuie să determinăm care sunt asumpțiile pe care le putem extrage din schimbul de mesaje. În articolul care introduce logica BAN [4], autorii selectează următoarele asumpții:

$$A \mid\equiv A \overset{K_{as}}{\longleftrightarrow} S \tag{4.1}$$

$$S \mid\equiv A \overset{K_{as}}{\longleftrightarrow} S \tag{4.2}$$

$$S \mid\equiv A \overset{K_{ab}}{\longleftrightarrow} B \tag{4.3}$$

$$A \mid\equiv (S \Rightarrow A \overset{K}{\leftrightarrow} B) \tag{4.4}$$

$$A \mid\equiv (S \Rightarrow A \overset{K_{ab}}{\longleftrightarrow} B) \tag{4.5}$$

$$A \mid\equiv N_a \tag{4.6}$$

$$S \mid\equiv A \overset{K_{ab}}{\longleftrightarrow} B \tag{4.7}$$

$$B \mid\equiv B \overset{K_{bs}}{\longleftrightarrow} S \tag{4.8}$$

$$S \mid\equiv B \overset{K_{bs}}{\longleftrightarrow} S \tag{4.9}$$

$$B \mid\equiv (S \Rightarrow A \overset{K}{\leftrightarrow} B) \tag{4.10}$$

$$B \mid\equiv N_b \tag{4.11}$$

$$B \mid\equiv A \overset{K_{ab}}{\longleftrightarrow} B \tag{4.12}$$

Asumpția (4.12) este una dintre problemele identificate ulterior asupra modelării oferite de *BAN*, fiind cea pentru care autorii au fost criticați, ea nefiind naturală și

nefiind corectă întrucât, din schimbul de mesaje, B cunoaște cheia de comunicare cu A abia după ce A primește răspunsul de la S.

Aplicând, succesiv, pașii din schimbul de mesaje conform specificației protocolului, ajungem la următoarele modelări:

1. A recepționează mesajul de la S, ceea ce înseamnă că $A \triangleleft \{N_a, (A \overset{K_{ab}}{\longleftrightarrow} B), A \overset{K_{ab}}{\longleftrightarrow} B, \{A \overset{K_{ab}}{\longleftrightarrow} B\}_{K_{bs}}\}_{K_{as}}$ pe care îl poate decripta, iar cum A știe că Na este proaspăt generat, atunci poate să determine că,

2. $A \mid\equiv S \mid\equiv A \overset{K_{ab}}{\longleftrightarrow} B$ (A crede că S crede în cheia de comunicare dintre A și B) și că $A \mid\equiv S \mid\equiv A \overset{K_{ab}}{\longleftrightarrow} B$ (A crede că S crede că cheia de comunicare este încă *alive*).

3. din *jurisdicție* obținem că $A \mid\equiv A \overset{K_{ab}}{\longleftrightarrow} B$ și că $A \mid\equiv A \overset{K_{ab}}{\longleftrightarrow} B$ (inferență imediată din faptul că S controlează o formulă, iar A crede că S crede formula, atunci A crede formula).

4. A vede și partea din mesaj criptata după cheia de comunicare dintre B și S, doar că nu poate decripta acea parte: $A \triangleleft \{A \overset{K_{ab}}{\longleftrightarrow} B\}_{K_{bs}}$.

5. B va deduce imediat că S i-a spus cheia de comunicare, însă nu poate deduce dacă mesajul transmis este proaspăt generat sau nu (nu are capabilitatea lui A de a decide dacă un mesaj transmis de server este *fresh*), de unde apare și necesitatea asumpției (4.12), pentru a deduce, din $B \mid\equiv S \mid\sim A \overset{K_{ab}}{\longleftrightarrow} B$ că $B \mid\equiv A \overset{K_{ab}}{\longleftrightarrow} B$.

6. în acest moment, A și B cred simultan în cheia lor comună de comunicare, iar din schimbul de mesaje, fiecare crede în faptul că și celălalt crede în cheia de comunicare, însemnând o *autentificare mutuală*.

Cu aceste asumpții și aplicând regulile de deducție, conchidem că protocolul Needham-Schroeder asigură o autentificare mutuală a agenților participanți.

4.1.3 Descrierea protocolului Needham-Schroeder în Lean

În cadrul acestei secțiuni, vom analiza anumite proprietăți ale protocolului *Needham-Schroeder*, folosind reprezentarea în formulele din sistemul *DELP*. Vom formaliza schimbul de mesaje conform următoarelor formule:

Inițializarea protocolului

Cunoștințele inițiale ale agenților se vor reprezenta astfel:

$$K_A(@N_A \wedge @key_{K_{AS}}(A,S)) \tag{4.13}$$
$$K_S(@key_{K_{AS}}(A,S) \wedge @key_{K_{BS}}(B,S) \wedge @key_{K_{AB}}(A,B)) \tag{4.14}$$
$$K_B @key_{K_{BS}}(B,S) \tag{4.15}$$

Echivalentul în *Lean* va fi următorul:

```
axiom NSinit (σ : ℕ) { Γ : ctx σ } { A B S Na Kab Kas Kbs : message σ }
  : σ-Γ ⊢ (K A, ((ι Na) ∧ (ι Kas.keys A S)))
    ∧ (K S, ((ι Kas.keys A S) ∧ (ι Kbs.keys B S) ∧ (ι Kab.keys A B)))
    ∧ (K B, (ι Kbs.keys B S)).
```

Prima rundă: schimbul de mesaje între A și S

După ce agentul A inițializează comunicarea, serverul (S) primește *nonce*-ul generat:

$$[send_A][recv_S]@N_A \tag{4.16}$$

```
axiom NS1AtoS (σ : ℕ) { Γ : ctx σ } { A S Na : message σ }
  : σ-Γ ⊢ [send A][recv S](ι Na).
```

A doua rundă: schimbul de mesaje între S și A

S îi transmite lui A *nonce*-ul criptat cu cheia simetrică dintre A și S și cheia de comunicare cu B, criptată atât doar cu cheia dintre A și S, cât și cu cheia dintre B și S, pentru a fi transmisă sigur de către A în rundele următoare.

$$[send_S][recv_A]\Big(@\{N_A\}_{K_{AS}} \wedge @\{key_{K_{AB}}(A,B)\}_{K_{AS}} \wedge @\{\{key_{K_{AB}}(A,B)\}_{K_{BS}}\}_{K_{AS}}\Big) \tag{4.17}$$

```
axiom NS2StoA (σ : ℕ) { Γ : ctx σ } { A B S Na Kab Kas Kbs : message σ }
  : σ-Γ ⊢ [send S][recv A]((ι {Na}Kas)
    ∧ (ι {(Kab.keys A B)}Kas)
    ∧ (ι {{(Kab.keys A B)}Kbs}Kas)).
```

A treia rundă: schimbul de mesaje între A și B

A îi transmite lui B cheia de comunicare, criptată cu cheia de comunicare dintre B și S:

$$[send_A][recv_B]@\{key_{K_{AB}}(A,B)\}_{K_{BS}} \tag{4.18}$$

```
axiom NS3AtoB (σ : ℕ) { Γ : ctx σ } { A B S Kab Kbs : message σ }
  : σ-Γ ⊢ [send A][recv B]ι {(Kab.keys A B)}Kbs.
```

Ultima rundă: schimbul bidirecțional între A și B

În cadrul acestei ultime etape, agenții aplică o funcție de control asupra *nonce*-ului. Această parte nu va fi momentan modelată, întrucât putem obține verificarea anumitor proprietăți de securitate doar din cunoștințele inițiale și primele trei schimburi de mesaje.

4.1.4 Verificarea proprietăților de securitate ale protocolului Needham-Schroeder în Lean

Pentru a putea demonstra proprietățile de securitate, vom demonstra, inițial, următoarele leme.

Lema 12. *Dacă* Γ *este o mulțime de enunțuri,* i *și* j *sunt doi agenți, iar* φ *este o formulă, atunci* $\Gamma \vdash [send_i][recv_j]\varphi$ *implică* $\Gamma \vdash K_j\varphi$.

Demonstrație.

```
lemma secv_imp_knowledge (σ : ℕ) { Γ : ctx σ } { i j : message σ } { φ :
    form σ }
  : (σ-Γ ⊢ [send i][recv j]φ) → (σ-Γ ⊢ K j, φ) :=
  assume h0 : σ-Γ ⊢ [send i][recv j]φ,
  have h1 : σ-Γ ⊢ ([send i]([recv j]φ)) → K i, [recv j]φ,
    from @knowsendf σ Γ i ([recv j]φ),
  have h2 : σ-Γ ⊢ K i, [recv j]φ,
    from @mp σ Γ ([send i]([recv j]φ)) (K i, [recv j]φ) h1 h0,
  have h3 : σ-Γ ⊢ (K i, [recv j]φ) → [recv j]φ,
    from @ktruth σ Γ i ([recv j]φ),
  have h4 : σ-Γ ⊢ [recv j]φ,
    from @mp σ Γ (K i, [recv j]φ) ([recv j]φ) h3 h2,
  have h5 : σ-Γ ⊢ ([recv j]φ) → (K j, φ),
    from @knowreceivef σ Γ j φ,
  show σ-Γ ⊢ K j, φ,
    from @mp σ Γ ([recv j]φ) (K j, φ) h5 h4.
```

Lema 13. *În cadrul protocolului Needham-Schroeder, agentul* A *știe cheia de criptare* K_{AS}.

Demonstrație.

```
lemma A_knows_Kas (σ : ℕ) { Γ : ctx σ } { A B S Na Kab Kas Kbs : message
    σ }
  : σ-Γ ⊢ K A, ι(Kas.keys A S) :=
  have h0 : σ-Γ ⊢ (K A, ((ι Na) ∧ (ι Kas.keys A S)))
    ∧ (K S, ((ι Kas.keys A S) ∧ (ι Kbs.keys B S) ∧ (ι Kab.keys A B)))
    ∧ (K B, (ι Kbs.keys B S)),
```

```
    from @NSinit σ Γ A B S Na Kab Kas Kbs,
  have h1 : σ-Γ ⊢ K A,(ι Na) ∧ ι Kas.keys A S,
    from @andleft σ Γ (K A, ((ι Na) ∧ (ι Kas.keys A S)))
    ((K S, ((ι Kas.keys A S) ∧ (ι Kbs.keys B S) ∧ (ι Kab.keys A B)))
    ∧ (K B, (ι Kbs.keys B S))) h0,
  have h2 : σ-Γ ⊢ (K A,(ι Na) ∧ ι Kas.keys A S) → ((ι Na) ∧ (ι Kas.keys
      A S)),
    from @ktruth σ Γ A ((ι Na) ∧ (ι Kas.keys A S)),
  have h3 : σ-Γ ⊢ ((ι Na) ∧ (ι Kas.keys A S)),
    from @mp σ Γ (K A,(ι Na) ∧ ι Kas.keys A S) ((ι Na) ∧ (ι Kas.keys A S
       )) h2 h1,
  have h4 : σ-Γ ⊢ (ι Kas.keys A S),
    from @andright σ Γ (ι Na) (ι Kas.keys A S) h3,
  show σ-Γ ⊢ K A, ι Kas.keys A S,
    from @kgen σ Γ (ι Kas.keys A S) A h4.
```

Lema 14. *În cadrul protocolului Needham-Schroeder, după ce recepționează mesajul de la S, agentul A știe cheia de comunicare între A și B, criptată cu cheia dintre A și S.*

Demonstrație.

```
lemma A_knows_Kab_encrypted_Kas (σ : ℕ) { Γ : ctx σ } { A B S Na Kab Kas
     Kbs : message σ }
  : σ-Γ ⊢ K A, ι{Kab.keys A B}Kas :=
  have h0 : σ-Γ ⊢ [send S][recv A]((ι {Na}Kas)
    ∧ (ι {(Kab.keys A B)}Kas)
    ∧ (ι {{(Kab.keys A B)}Kbs}Kas)),
    from @NS2StoA σ Γ A B S Na Kab Kas Kbs,
  have h1 : (σ-Γ ⊢ [send S][recv A]((ι {Na}Kas)
    ∧ (ι {(Kab.keys A B)}Kas)
    ∧ (ι {{(Kab.keys A B)}Kbs}Kas)))
    → (σ-Γ ⊢ K A, ((ι {Na}Kas)
    ∧ (ι {(Kab.keys A B)}Kas)
    ∧ (ι {{(Kab.keys A B)}Kbs}Kas))),
    from @secv_imp_knowledge σ Γ S A ((ι {Na}Kas)
    ∧ (ι {(Kab.keys A B)}Kas)
    ∧ (ι {{(Kab.keys A B)}Kbs}Kas)),
  have h2 : σ-Γ ⊢ K A, ((ι {Na}Kas)
    ∧ (ι {(Kab.keys A B)}Kas)
    ∧ (ι {{(Kab.keys A B)}Kbs}Kas)),
    from h1 h0,
  have h3 : σ-Γ ⊢ (K A, ((ι {Na}Kas)
    ∧ (ι {(Kab.keys A B)}Kas)
    ∧ (ι {{(Kab.keys A B)}Kbs}Kas)))
    → ((ι {Na}Kas)
    ∧ (ι {(Kab.keys A B)}Kas)
    ∧ (ι {{(Kab.keys A B)}Kbs}Kas)),
    from @ktruth σ Γ A ((ι {Na}Kas)
    ∧ (ι {(Kab.keys A B)}Kas)
```

```
28      ∧ (ι {{(Kab.keys A B)}Kbs}Kas)),
29    have h4 : σ-Γ ⊢ (ι {Na}Kas)
30      ∧ (ι {(Kab.keys A B)}Kas)
31      ∧ (ι {{(Kab.keys A B)}Kbs}Kas),
32      from @mp σ Γ  (K A, ((ι {Na}Kas)
33      ∧ (ι {(Kab.keys A B)}Kas)
34      ∧ (ι {{(Kab.keys A B)}Kbs}Kas)))
35      ((ι {Na}Kas)
36      ∧ (ι {(Kab.keys A B)}Kas)
37      ∧ (ι {{(Kab.keys A B)}Kbs}Kas)) h3 h2,
38    have h5 : σ-Γ ⊢ (ι {(Kab.keys A B)}Kas) ∧ (ι {{(Kab.keys A B)}Kbs}Kas)
        ,
39      from @andright σ Γ (ι {Na}Kas) ((ι {(Kab.keys A B)}Kas) ∧ (ι {{(Kab.
          keys A B)}Kbs}Kas)) h4,
40    have h6 : σ-Γ ⊢ ι {(Kab.keys A B)}Kas,
41      from @andleft σ Γ (ι {(Kab.keys A B)}Kas) (ι {{(Kab.keys A B)}Kbs}
          Kas) h5,
42    show σ-Γ ⊢ K A, ι {(Kab.keys A B)}Kas,
43      from @kgen σ Γ (ι {(Kab.keys A B)}Kas) A h6.
```

În acest moment, putem demonstra următoarea teoremă, și anume:

Teorema 3. *Agentul A știe cheia de comunicare dintre A și B (i.e., A știe secretul comun dintre A și B).*

Demonstrație.

```
1   theorem A_knows_Kab (σ : ℕ) { Γ : ctx σ } { A B S Na Kab Kas Kbs :
        message σ }
2     : σ-Γ ⊢ K A, ι(Kab.keys A B) :=
3     have h0 : σ-Γ ⊢ K A, ι{Kab.keys A B}Kas,
4       from @A_knows_Kab_encrypted_Kas σ Γ A B S Na Kab Kas Kbs,
5     have h1 : σ-Γ ⊢ K A, ι(Kas.keys A S),
6       from @A_knows_Kas σ Γ A B S Na Kab Kas Kbs,
7     have h2 : σ-Γ ⊢ (K A, ι(Kas.keys A S)) → ι(Kas.keys A S),
8       from @ktruth σ Γ A ι(Kas.keys A S),
9     have h3 : σ-Γ ⊢ ι(Kas.keys A S),
10      from @mp σ Γ (K A, ι(Kas.keys A S)) (ι(Kas.keys A S)) h2 h1,
11    have h4 : σ-Γ ⊢ (K A, ι{Kab.keys A B}Kas) → ι{Kab.keys A B}Kas,
12      from @ktruth σ Γ A ι{Kab.keys A B}Kas,
13    have h5 : σ-Γ ⊢ ι{Kab.keys A B}Kas,
14      from @mp σ Γ (K A, ι{Kab.keys A B}Kas) (ι{Kab.keys A B}Kas) h4 h0,
15    have h6 : σ-Γ ⊢ ((ι(Kas.keys A S)) ∧ (ι{Kab.keys A B}Kas)) → ([send S
        ](ι (Kab.keys A B))),
16      from @honestyright σ Γ (Kab.keys A B) Kas A S,
17    have h7 : σ-Γ ⊢ (ι(Kas.keys A S)) ∧ (ι{Kab.keys A B}Kas),
18      from @andintro σ Γ (ι(Kas.keys A S)) (ι{Kab.keys A B}Kas) h3 h5,
19    have h8 : σ-Γ ⊢ [send S](ι (Kab.keys A B)),
20      from @mp σ Γ ((ι(Kas.keys A S)) ∧ (ι{Kab.keys A B}Kas)) ([send S](ι
          (Kab.keys A B))) h6 h7,
```

```
21    have h9 : σ-Γ ⊢ ([send S](ι Kab.keys A B)) → (ι Kab.keys A B),
22      from @pdtruth σ Γ (send S) (ι Kab.keys A B),
23    have h10 : σ-Γ ⊢ (ι Kab.keys A B),
24      from @mp σ Γ ([send S](ι Kab.keys A B)) (ι Kab.keys A B) h9 h8,
25    show σ-Γ ⊢ K A, ι Kab.keys A B,
26      from @kgen σ Γ (ι Kab.keys A B) A h10.
```

Lema 15. *În cadrul protocolului Needham-Schroeder, agentul B cunoaște cheia de criptare dintre el și S.*

Demonstrație.

```
1  lemma B_knows_Kbs (σ : ℕ) { Γ : ctx σ } { A B S Na Kab Kas Kbs : message
       σ }
2    : σ-Γ ⊢ K B, (ι Kbs.keys B S) :=
3    have h0 : σ-Γ ⊢ (K A, ((ι Na) ∧ (ι Kas.keys A S)))
4      ∧ (K S, ((ι Kas.keys A S) ∧ (ι Kbs.keys B S) ∧ (ι Kab.keys A B)))
5      ∧ (K B, (ι Kbs.keys B S)),
6      from @NSinit σ Γ A B S Na Kab Kas Kbs,
7    have h1 : σ-Γ ⊢ (K S, ((ι Kas.keys A S) ∧ (ι Kbs.keys B S) ∧ (ι Kab.
         keys A B)))
8      ∧ (K B, (ι Kbs.keys B S)),
9      from @andright σ Γ (K A, ((ι Na) ∧ (ι Kas.keys A S)))
10     ((K S, ((ι Kas.keys A S) ∧ (ι Kbs.keys B S) ∧ (ι Kab.keys A B)))
11     ∧ (K B, (ι Kbs.keys B S))) h0,
12   show σ-Γ ⊢ K B, (ι Kbs.keys B S),
13     from @andright σ Γ (K S, ((ι Kas.keys A S) ∧ (ι Kbs.keys B S) ∧ (ι
           Kab.keys A B)))
14     (K B, (ι Kbs.keys B S)) h1.
```

Având toate aceste rezultate, putem demonstra că agentul B cunoaște și el cheia de comunicare cu A, astfel că partajează un secret comun.

Teorema 4. *Agentul B știe cheia de comunicare dintre A și B.*

Demonstrație.

```
1  theorem B_knows_Kab (σ : ℕ) { Γ : ctx σ } { A B S Na Kab Kas Kbs :
       message σ }
2    : σ-Γ ⊢ K B, ι(Kab.keys A B) :=
3    have h0 : σ-Γ ⊢ K B, (ι Kbs.keys B S),
4      from @B_knows_Kbs σ Γ A B S Na Kab Kas Kbs,
5    have h1 : σ-Γ ⊢ [send A][recv B]ι {(Kab.keys A B)}Kbs,
6      from @NS3AtoB σ Γ A B S Kab Kbs,
7    have h2 : (σ-Γ ⊢ [send A][recv B]ι {(Kab.keys A B)}Kbs) → (σ-Γ ⊢ K B,
         ι {(Kab.keys A B)}Kbs),
8      from @secv_imp_knowledge σ Γ A B (ι {(Kab.keys A B)}Kbs),
9    have h3 : σ-Γ ⊢ K B, ι {(Kab.keys A B)}Kbs,
10     from h2 h1,
11   have h4 : σ-Γ ⊢ (K B, (ι Kbs.keys B S)) → (ι Kbs.keys B S),
```

```
12      from @ktruth σ Γ B (ι Kbs.keys B S),
13    have h5 : σ-Γ ⊢ ι Kbs.keys B S,
14      from @mp σ Γ (K B, (ι Kbs.keys B S)) (ι Kbs.keys B S) h4 h0,
15    have h6 : σ-Γ ⊢ (K B, ι {(Kab.keys A B)}Kbs) → ι {(Kab.keys A B)}Kbs,
16      from @ktruth σ Γ B (ι {(Kab.keys A B)}Kbs),
17    have h7 : σ-Γ ⊢ ι {(Kab.keys A B)}Kbs,
18      from @mp σ Γ (K B, ι {(Kab.keys A B)}Kbs) (ι {(Kab.keys A B)}Kbs) h6
          h3,
19    have h8 : σ-Γ ⊢ (ι Kbs.keys B S) ∧ (ι {(Kab.keys A B)}Kbs),
20      from @andintro σ Γ (ι Kbs.keys B S) (ι {(Kab.keys A B)}Kbs) h5 h7,
21    have h9 : σ-Γ ⊢ (ι Kbs.keys B S) ∧ (ι {(Kab.keys A B)}Kbs) → [send S]ι
        (Kab.keys A B),
22      from @honestyright σ Γ (Kab.keys A B) Kbs B S,
23    have h10 : σ-Γ ⊢ [send S]ι (Kab.keys A B),
24      from @mp σ Γ ((ι Kbs.keys B S) ∧ (ι {(Kab.keys A B)}Kbs)) ([send S]ι
          (Kab.keys A B)) h9 h8,
25    have h11 : σ-Γ ⊢ ([send S]ι (Kab.keys A B)) → (ι Kab.keys A B),
26      from @pdtruth σ Γ (send S) (ι Kab.keys A B),
27    have h12 : σ-Γ ⊢ (ι Kab.keys A B),
28      from @mp σ Γ ([send S]ι (Kab.keys A B)) (ι Kab.keys A B) h11 h10,
29    show σ-Γ ⊢ K B, (ι Kab.keys A B),
30      from @kgen σ Γ (ι Kab.keys A B) B h12.
```

Concluzii și direcții viitoare

În această lucrare am definit *DELP*, un sistem dinamic epistemic pentru modelarea și analizarea protocoalelor de securitate. Am prezentat trei abordări recente în această direcție, principalele logici clasice pe care se bazează aspectele teoretice, am demonstrat completitudinea pentru *DELP* și am verificat corectitudinea logicii *BAN* în acest sistem, utilizând și o implementare a limbajului și a regulilor de deducție *BAN* în *Lean*.

Direcțiile viitoare includ rafinări ale sistemului pentru a putea modela proprietăți de securitate mai fine și, în plus, cuprind includerea unor noi abordări, pentru a îmbunătăți expresivitatea acestei logici, în limita rezultatelor teoretice.

O primă direcție viitoare constă în modelarea încrederii (în locul cunoașterii) pentru anumite proprietăți de securitate, ceea ce s-ar putea modela prin intermediul unui operator modal din sistemul *K*. Cunoașterea prezentată în această lucrare face parte din sistemul modal *S5* (sistemul *KT45*).

O a doua direcție constă în schimbarea mesajelor *nonce* și *key* în predicate peste mulțimea mesajelor, pentru a putea exprima o proprietate a acestora și pentru a facilita reprezentarea cunoștințelor adversarului. În modelarea curentă, mulțimea *Exp* a cunoștințelor reprezintă cunoștințele generale ce pot fi văzute din exterior, în analiză, și nu cunoștințele unui adversar, inclus în sesiunile de comunicare.

În plus, considerăm important pentru studiile următoare să considerăm și adăugarea unui comportament temporal, pentru a putea modela proprietatea de *proaspăt generat* (*freshness*); momentan, utilizăm o variantă mai slabă, și anume unicitatea pe sistem (*nonce*).

Nu în ultimul rând, considerăm și varianta de a adăuga interpretarea probabilistică, prezentată inițial în articolul [5].

Pe partea de implementare în *Lean* vom adăuga demonstrația pentru teorema de completitudine și vom ține toate rezultatele teoretice grupate și verificate automat pentru orice modificare ulterioară.

Bibliografie

[1] Jeremy Avigad, Leonardo de Moura, and Soonho Kong. Theorem proving in lean, 2021.

[2] Bruno Bentzen. A henkin-style completeness proof for the modal logic s5. *arXiv preprint arXiv:1910.01697*, 2019.

[3] Patrick Blackburn, Maarten De Rijke, and Yde Venema. *Modal logic: graph. Darst*, volume 53. Cambridge University Press, 2002.

[4] Michael Burrows, Martin Abadi, and Roger Michael Needham. A logic of authentication. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 426(1871):233–271, 1989.

[5] Joseph Y Halpern, Ron van der Meyden, and Riccardo Pucella. An epistemic foundation for authentication logics. *arXiv preprint arXiv:1707.08750*, 2017.

[6] David Harel, Dexter Kozen, and Jerzy Tiuryn. Dynamic logic. In *Handbook of philosophical logic*, pages 99–217. Springer, 2001.

[7] G Hollestelle, S Mauw, and CJF Cremers. Systematic analysis of attacks on security protocols. *Master's Thesis, Technical University of Eindhover, Department of Mathematics and Computer Science*, 2005.

[8] Hans Van Ditmarsch, Sujata Ghosh, Rineke Verbrugge, and Yanjing Wang. Hidden protocols: Modifying our expectations in an evolving world. *Artificial Intelligence*, 208:18–40, 2014.

[9] Hans Van Ditmarsch, Wiebe van Der Hoek, and Barteld Kooi. *Dynamic epistemic logic*, volume 337. Springer Science & Business Media, 2007.